2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Search-based Strategy for Spatio-Temporal Environmental Property Restoration

Amel Nestor Docena and Alberto Quattrini Li

***Abstract*— This paper addresses the *spatio-temporal areas restoration problem*: a robot, with limited battery life, deployed in a known environment, needs to persistently plan a schedule to visit areas of interest and charge its battery as needed. The goal is to restore the areas' properties that temporally decay—such as air quality—so that the time the measured property values are below a certain threshold is minimized. This problem is different from typical problems solved in the area of monitoring a spatio-temporal environment. A related problem is the orienteering problem, where a robot visits nodes to maximize the profit collected at each visited node within a time budget frame. That problem is NP-hard. The typical formulation considers static profit, while we consider a time-varying one. Given look-ahead time window or schedule length, we formulate the problem as an optimization search problem with a temporal objective, and devise a heuristic function that enables finding solutions in polynomial time. The heuristic evaluates the discounted opportunity costs of a visit—a concept borrowed from economics. We then develop a greedy algorithm that takes the immediate feasible visit that minimizes this heuristic. This strategy addresses a primary limitation of a recent approach in applications where being able to revisit highly urgent areas within the time window of the schedule is critical. We provide a theoretical analysis on lower and upper bounds for the problem. Extensive experimental results with a robotic simulator show that our method is able to keep the areas in the environment above the threshold better than other methods and closer to the optimal. This work can enable high-impact applications, such as environmental preservation.**

I. Introduction

In this paper, we propose an optimization strategy for the open problem of *spatio-temporal areas restoration* (STAR): a robot, with limited battery that can be charged at a charging station, needs to persistently visit locations and restore their properties (e.g., air quality). Such properties decay over time, so the goal is to find a visit schedule that minimizes the time the measured property values are below a certain threshold (see Figure 1 for an illustration of the problem).

Addressing this problem is significant in many important applications. For example, in agriculture, crops are taken care of according to a farming cycle; water bodies (e.g., bays and lakes) are cleaned from periodic waste and algal bloom; public spaces, like offices, are cleaned and air-purified regularly. Although in some of the applications listed above, a network of static devices could be used, using robots doesn't require any change to the infrastructure in the environment, reducing the operational and maintenance costs. Only a few robots are needed to cover large areas. As

The authors are with Department of Computer Science, Dartmouth College, USA {amel.nestor.docena.gr, alberto.quattrini.li}@dartmouth.edu

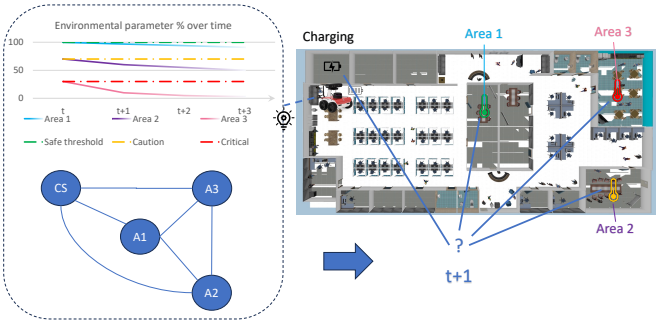


Fig. 1: Given the current state of the environment and the robot, we search for an optimal schedule the current battery can cover that minimizes the time that temporal property in areas fall below a critical threshold. This schedule deploys the robot on which areas to restore and when to charge in the next k visits.

robot costs have markedly decreased in the last few years, they have become an accessible technology even for small companies or individuals. Furthermore, their use may be the best alternative especially in scenarios where the task may be unreachable, risky, or hazardous for manned preservation [1].

The problem of persistently restoring the temporal properties of a number of areas spatially distributed in an environment is still open, because of its combinatorial nature. In the literature, there are methods not addressing the same problem as this paper, but solving some related problems. There are strategies for coverage, exploration, and/or monitoring [2]–[4] with the goal of accurately modeling a spatial phenomenon of an environment while minimizing the overall effort. There are methods for solving the orienteering problem, where the robot needs to visit a set of nodes to collect reward that do not change over time [5]. Recently, [6] proposed a dynamic programming based method for solving the orienteering problem with time-varying profits. The problem they addressed is different than what is considered in this paper: in our case, committing to specific choices might result in continued losses. In addition, the revisit of areas is typically not considered in the literature.

This paper solves STAR, providing the following key contributions:

1) A way to model this problem, including an abstraction of the state and temporal implications of each visit, borrowing the concept of 'opportunity cost of a decision' from economics. We formulate the objective as minimizing the opportunity cost of a schedule of visits.
2) To solve the objective for a single robot with a limit on battery charge, we propose a tree-based search optimization strategy that allows for revisit of areas,

differently from existing approaches, and especially suitable in applications where this revisit is critical.
3) Since the full tree search strategy—deemed as an optimal solution—is exponential, we propose a heuristic to find solutions within polynomial runtime.
4) Experimental analysis from robot simulations showing that our proposed strategy performs close to the optimal solution and better than other methods.

While some extensions (e.g., considering uncertainty and adding multiple robots) will be part of future work, this paper serves as foundation in persistent monitoring and restoration in a spatio-temporal environment, going beyond the common focus on monitoring, with potential for many applications.

II. Related Work

While the problem of persistently restoring environmental properties that change over time is still an open problem, there is research in related sub-problems.

There has been work in developing strategies for exploring and/or monitoring an environment, without focusing on restoration. The problem of exploring/monitoring an environment has different names in the literature: exploration [7]–[9], coverage [10], [11], information gathering [12]–[14], adaptive sampling [15], [16], surveillance/patrolling [17], [18]. The techniques that are used to solve such problems can be categorized in two main categories, *non-adaptive* and *adaptive*. Non-adaptive approaches typically assume that the environment is known and are based on a geometric formulation, where an environment is decomposed in cells, abstracted with a graph, and an order of visits is found to minimize the overall cost [2]. We also use a graph representation of the environment, but, because of the temporal changes, planning needs to consider such temporal variations and replanning might be necessary. Adaptive approaches instead model real-time the phenomenon to monitor and use information-theoretic techniques to identify areas that should be sampled next [4]. The majority of the work in the literature focused on monitoring static phenomena. More recently, there have been some work modeling temporal phenomena [19]. Note that some of these techniques have been used for restoring properties, but, differently from our paper, it is typically assumed that once an area is visited, that area is restored, such as in the case of oil spill cleaning [20]–[22].

Some work also looked at repeatedly covering the same area [23]–[25]. This perspective is similar to our problem as the robot needs to revisit areas, but in our paper, differently from the literature, the properties are being persistently restored given their continual decay.

There have been techniques for planning with temporal-logic specifications [26], mostly for motion planning [27]–[29] with specific task execution guarantees. Some work also looked at temporal task planning, e.g., for cleaning while identifying human actions [30], or for identifying optimal temporal task allocation to robots under resource constraints [31]. In our work, we model the problem as an optimization problem, given that it might not be possible with the available resources to always satisfy the areas being above a critical level.

A number of works also included battery constraints with potential need of going back to the charging station [32]–[35]. Our work explicitly includes battery constraint in the optimization problem and considers the action to go back and recharge in the planning.

A closely related problem to STAR is the orienteering problem (OP) [5], where nodes with profits need to be visited to maximize the overall collected profits. OP is known to be NP hard. Most approaches solving OP consider static profit. Instead, our problem involves a temporal objective function. There is a recent work that offers an approach to solve OP on time-varying profits with a single robot [6]. They formulated the problem as a mixed-integer program, wherein the objective is to maximize the time-varying profits among depots with a primary constraint being that areas are visited at most once within the time window. This problem formulation aligns with one of their application example, taxi routing. Although their approach could be applied to our problem by maximizing the inverse of the losses, the approach wouldn't generally optimally solve our problem, as also shown in the experiments: first, the areas need to be visited more than once and, second, committing to a solution would result in potential continual loss in what we have forgone.

Zhang and Vorobeychik [36] proposed the Generalized Cost-Benefit Algorithm to solve the single-robot OP with static profit, which greedily makes the decision that has the optimal marginal benefit to marginal cost ratio. This has theoretical guarantee but assuming a submodular objective function with a fixed budget constraint. In general, in our problem, we do not have a fixed budget constraint as it depends on the current battery level nor do we assume a submodular objective function. But that algorithm inspires the polynomial solution we propose.

Overall, there is a need to solve the open problem of STAR, tackled in this paper.

III. Problem Statement

An autonomous mobile robot r is deployed in an arbitrary known environment $\mathcal{W} \subset \mathbb{R}^2$, potentially with obstacles $\mathcal{W}_o \subset \mathcal{W}$. The robot operates in the free space $\mathcal{W}_f \subset \mathcal{W}$, wherein the robot's footprint is accounted for by the Minkowski sum that expands the obstacles accordingly, as commonly done in path planning [37]. The robot has a finite-range laser sensor and an odometer which allows it to localize with a probabilistic state estimator. The robot runs also path planner and low-level controller allowing the robot to reach goal locations. Moreover, it is equipped with a monitoring sensor that measures an environmental property of interest F (e.g., air quality) and a device to restore such (e.g., air purifier). The robot has limited battery, but can be recharged at a charging station, denoted as $0 \in \mathcal{W}_f$.

Within the environment, there are areas of interest $J = \{1, 2, ..., j\} \subset \mathcal{W}_f$, each with an environmental property to monitor and restore. The property of an area j, denoted as

F_j, changes over time (e.g., air quality drops due to the presence of people in the area). F_j is modeled by a function $F_j = g(t, \delta_j) \in \mathbb{R}$. We assume that $g(\cdot)$ is monotonic and described with a corresponding decay rate $\delta_j > 0$, as modeled in some environmental studies [38]–[41]. While the information gathering is also an interesting research direction, here we assume that there is prior knowledge about $g(\cdot)$, for example through prior survey of the areas, and δ_j stays fixed. Without loss of generality, in this paper, we consider the case where F decays over time; the case where F grows can be formulated as the inverse.

To provide some theoretical guarantees, we assume that once r decides to visit a location from its current location it will certainly arrive there. When the robot restores F, F will be restored back to its maximum level. The same happens when recharging the battery. Note that both restoration and charge operations take some time; moreover, noise can happen in both travel and restoration—these are accounted for in our formulation but assumed to stay fixed.

Thus, the STAR problem is: given the known environment $\mathcal{W}$, the charging station and J areas of interest, the robot r, its current location and battery level b, and the time elapsed since last restoration for each decaying area, the goal is to find a *schedule of visits* D^* that minimizes the time that F of areas is below a certain critical threshold $z \in \mathbb{R}$ and such that the battery level does not fully deplete while on mission.

IV. Search-based Strategy for Temporal Area Restoration

To solve STAR, we formulate a temporal optimization search problem given a schedule length. We first introduce the modeling of the problem; we then describe the general algorithm to solve it; then present the heuristic to make the approach scalable.

A. Modeling

Environment: We represent the charging station and areas of interest in $\mathcal{W}_f$ with a graph $G = (V, E)$, where the vertices $V = \{0\} \cup J$ and the edges E represent the connectivity between vertices, and information on distance can be stored in a matrix $\mathbf{H}$ as determined by a path planner—we use A^* on an inflated map to account for the robot footprint.

Vertices. In each area ($j \in V$) starting at $t = 0$, F_j decays with respect to t at a rate δ_j and will hit the threshold z at some point and may decay further. We establish that we incur a *loss* in an area only if F_j starts to decay and continues to grow increasingly as F_j decays toward z and beyond. We thus associate each area with a loss function $L(t, \delta_j) \in \mathbb{R}$ with two key properties: (1) inversely related to $g(\cdot)$, and (2) the marginal loss is increasing w.r.t. t. Accordingly, because of how L behaves and the monotonicity assumption of g, for any time elapsed $t < t'$, $g(t, \delta_j) > g(t', \delta_j)$ and $L(t, \delta_j) < L(t', \delta_j)$. In addition, L is differentiable throughout $t \geq 0$ such that $\frac{\partial L(t,\delta_j)}{\partial t^2} > 0$ at any t.

Edges. An edge exists between two distinct vertices l and j if the robot can reach j starting from l. The distance is $\mathbf{H}[l, j] > 0$.

State representation: We represent a state of the environment for a given time stamp T as $(T, l, b, \mathbf{t}, \delta)$, where l is the current location of the robot; $\mathbf{t} = \{t_1, ..., t_j\}$ is the set of time elapsed since last restoration for all areas, $(\forall t \in \mathbf{t} : t \geq 0)$; and, $\delta = \{\delta_1, ..., \delta_j\}$ is the set of their respective decay rates.

Visits: From a state s, the possible visits (or actions) d for r are either to monitor/restore areas ($d = j$) or to charge at the charging station ($d = 0$).

Duration. We can measure the time to travel across vertices by $\mathbf{T} = \mathbf{H}/v$, where v is the robot's velocity. We thus model the duration of a visit taken at l as $\tau = (\tau_x + \tau_y)(1+\varepsilon)$, where $\tau_x = \mathbf{T}[l, d]$ is the travel duration; τ_y is the duration to charge or restore an area estimated by assuming a known charging/restoration rate; and ε is the average discrepancy rate sampled from a history of actual runs to capture the noise that can occur during mission (e.g., moving obstacles, hardware malfunction), potentially extending the assumed duration.

Resource consumption. We model the battery life as a linear function to the current consumption over each edge [35] and in restoring F. The consumption function $c(d)$ accounts for the travel and restore times of the visit:

$$c(d) = \rho_x(\tau_x) + \mathbf{1}_d \rho_y(\tau_y) \in \mathbb{R}, \tag{1}$$

where ρ_x and ρ_y are the unit battery spent in travelling and in restoring F, respectively; $\mathbf{1}_d$ to indicate, (i.e., $\mathbf{1}_d = 1$), if the decision is to preserve a temporal area, 0 if to charge.

Feasibility. A visit is accordingly *feasible* only if:

$$b > c(d) + c(d = 0); \tag{2}$$

that is, the current battery can strictly cover the consumption, and if the decision is to restore, an additional consumption to further allow r to head back to the charging station.

Transition model: We define a piece-wise transition function $w(s_{i-1}, d_i) = (T_i, l_i, b_i, \mathbf{t}_i, \delta)$ that yields a subsequent state s_i from s_{i-1} depending on d_i. In either case of d_i: $T_i = T_{i-1} + \tau$, while δ stay fixed. If $d_i = j$, the values for the state elements that have temporal implications, $(l_i, b_i, \mathbf{t}_i)$, become: $l_i = j$, $b_i = b_{i-1} - c(d)$, $\mathbf{t}_i = \{t_{(j,i)} = 0\} \cup \{\forall m \neq j : t_{(m,i)} = t_{(m,i-1)} + \tau\}$. That is: the new location will be the restored area; the battery reduced for the consequent consumption; and the time elapsed for that area is reset while that in other areas continually lapses for the duration of this visit. But if $d_i = 0$: $l_i = 0$, $b_i = \tilde{b}$, $\mathbf{t}_i = \{\forall j \in J : t_{(j,i)} = t_{(j,i-1)} + \tau\}$. For this case, the new location is the charging station; the battery is recharged to max level $\tilde{b} \in \mathbb{R}^+$; but, the time in all areas lapses for the charge duration.

Opportunity cost: Each visit, therefore, has an *opportunity cost* amounting to the losses incurred in decaying areas that have been forgone by committing to the visit. We define the opportunity cost function of d_i given s_{i-1} as:

$$q(s_{i-1}, d_i) = \sum_{j \in J} L(t_{(j,i)}, \delta_j) \in \mathbb{R}. \tag{3}$$

where each $t_{(j,i)} \in w(s_{i-1}, d_i)$. Note that if $d_i = j$, we are just summing up the losses of those areas that have not been visited, since no loss is incurred in a newly restored area by definition.

Schedule of visits. We measure the opportunity cost of a schedule, $D = \{d_1, ..., d_k\}$, if deployed starting from state s_0 as:

$$Q_D = \sum_{i=1}^{k} \gamma^{i-1} q(s_{i-1}, d_i) \in \mathbb{R}, \tag{4}$$

where $\gamma \in (0, 1]$ is a discount factor for future losses.

Objective function: Let $\mathbf{D}$ be the set of possible schedules of length k. Our strategy will choose the optimal schedule $D^* \in \mathbf{D}$ that has the minimum opportunity cost, while ensuring feasibility:

$$\begin{aligned} D^* = \arg\min_{D} \quad & Q_D \\ \text{s.t.} \quad & c(d_i) \leq b_i, (\forall i = 1, ..., k) \\ & d_i \in D. \end{aligned} \tag{5}$$

Note that D^* may not be unique. Among $\mathbf{D}^* \subseteq \mathbf{D}$ that contains all optimal schedules, we establish a further condition that ensures resource efficiency to refine our choice of D^*: $(\forall D \in \mathbf{D}^*)(\forall D' \in \mathbf{D}^*)$ where $D \neq D'$,

$$(D = D^*) \implies \sum_{d \in D} c(d) < \sum_{d' \in D'} c(d'). \tag{6}$$

B. Tree-based Search Algorithm for Optimal Deployment

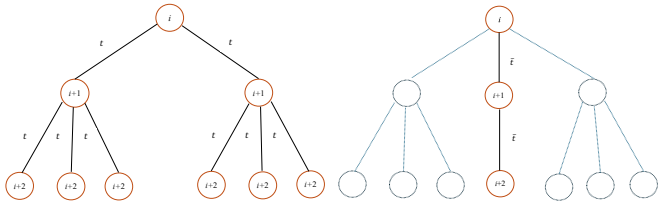
Starting from current location l, an algorithm that guarantees optimality builds a tree V^k, considering the different visits $d \in V$ up to depth k, while keeping track of the opportunity cost and resource consumption of the current solution. Since the losses in the areas are calculated based on the opportunity cost previously defined, the computational complexity is $O(|J||V|^k)$. To solve practical problems, there are a number of speedup techniques that can be used.

Reducing the schedule length: When $k = 1$, the schedule is *myopic*. This solution resembles that of a UCB-style as in [36]. When instead $k > 1$, it becomes *non-myopic*. Obviously, reducing k drastically reduces the computation, although this may affect performance as varying k has varying overall effectiveness on a given problem as will be shown in the experiments.

Heuristic: Instead of building the complete search tree with depth k, we provide a greedy strategy that evaluates the *discounted* opportunity costs of each next feasible visit d from s_0 by using a heuristic (Fig. 2):

$$\hat{Q}_d = \sum_{j \in J} L(t_{(j,1)}, \delta_j) + \sum_{i=2}^{k} [\gamma^{i-1} \sum_{j \in J} L(\hat{t}_{(j,i)}, \delta_j)] \in \mathbb{R} \tag{7}$$

where the first summand is the immediate opportunity cost of d; while, the second summand is its forecasted opportunity costs supposing r takes $k-1$ visits afterwards, discounted by γ. The time elapsed for each area j that is decaying for $i \geq 2$ is forecasted by Eq. (8):



(a) *Actual* time elapsed F_j is decaying in subsequent visits after visit in i. (b) *Average* time elapsed F_j is decaying for each subsequent visit after visit in i.

Fig. 2: Heuristic function, Eq. (7), forecasts future opportunity costs by how much areas are decaying in subsequent visits, on average (right), assuming an area gets restored once it hits the critical threshold and r charges only when no more feasible visit, allowing for not expanding all the branches of the tree as for the tree-based search strategy (left).

$$\hat{t}_{(j,i)} = \begin{cases} \hat{t}_{(j,i-1)} + \bar{\tau}_j \text{ if } g(\hat{t}_{(j,i-1)}, \delta_j) \geq z \\ 0 \text{ otherwise,} \end{cases} \tag{8}$$

where $\bar{\tau}_j$ is the average duration that area j decays. This estimation is based on the assumption (A1) that in future visits $2 \leq i \leq k$, areas would be restored once they fall below the critical threshold, at least. In addition, it is assumed (A2) that r will charge only when there is no feasible visit.

Measuring $\bar{\tau}_j$. Suppose that the duration of visits is based primarily on $\mathbf{T}$, wherein every j-th column is the duration it would take from all vertices to reach vertex j. Suppose r commits to a visit other than restoring area j in the next visit $i+1$, and so j keeps decaying. We can measure $\bar{\tau}_j$ by taking the average of all the columns in the matrix but column j.

Algorithm 1 uses such a heuristic function: we pick the visit d^* that minimizes $\hat{Q}_d$ and has maximum remaining feasible battery (i.e., after accounting for the resource consumption of the visit plus the consumption to head back to the charging station, as safety measure). We will show in the next section the effect of different values for k and γ. This strategy allows for solving problems in *polynomial time*, as now the computational complexity is $O(|J|^2 k)$.

By assuming (A1) and (A2), we are introducing suboptimality, but we are trading for significant computational efficiency. Our experimental results show that the heuristic algorithm performs close to the optimal algorithm.

C. Discussion: Strict lower and upper bounds of Q_D for any algorithm

Suppose from s_0 we employ an algorithm to come up with a schedule of length k. Suppose also that the duration of each visit is based primarily on travel as measured by $\mathbf{T}$. For any visit $d_{i \leq k}$, the shortest time elapsed possible that an area $j \in J$ is decaying is $\min(\mathbf{t}_{i-1}) + \min(\mathbf{T})$. Consequently, the loss in an area $L(t_{(j,i)}, \delta_j)$, where $t_{(j,i)} \in w(s_{i-1}, d_i)$, is lower bounded by $L([\min(\mathbf{t}_{i-1}) + \min(\mathbf{T})], \min(\delta))$. By multiplying with $|J| - 1$ areas, we can likewise bound $q(s_{i-1}, d_i)$ below. And so for the entire schedule, a strict lower bound is:

$$(|J| - 1) \sum_{i=1}^{k} \gamma^{i-1} L([\min(\mathbf{t}_{i-1}) + \min(\mathbf{T})], \min(\delta)) < Q_D.$$

Algorithm 1 Best-forecasted Visit Greedy Deployment

Input: current state s_0, distance matrix $\mathbf{H}$, robot velocity v, loss $L(\cdot)$, schedule length k, discount γ
Output: At every iteration, while r is operating, an allocation d^*

```
Set Γ = [γ^0, γ^1, ..., γ^{k-1}].
Measure τ̄_j for each j ∈ J based on T = H/v.
while operation do
    Initialize d as array, L as |J| × k matrix.
    for j ∈ J where feasible do
        for i in {1, ..., k} do
            for j ∈ J do
                Forecast t̂_(j,i), wherein for i ≥ 2 use Eq. (8)
                L_[j,i] = L(t̂_(j,i), δ_j)
        L = sum L by row
        Q̂_d = Γ · L
        b_d = b − [c(d = j) + c(d = 0)]
        Store (d, Q̂_d, b_d) in d
    d* = 0
    if d is not ∅ then
        Sort d ascendingly by Q̂_d then descendingly by b_d
        d* = first element of d
    Allocate d*
```

Similarly for the upper bound: if we take d_1 from s_0, the largest loss from an area is upper bounded by $L([\max(\mathbf{t}_0) + \max(\mathbf{T})], \max(\delta))$; moreover, the marginal loss in the latter is greater. Note that the largest loss that can be incurred in each subsequent visit comes from foregoing restoring an area that has both the largest loss and marginal loss. And so any visit taken at $i \leq k$, the loss from an area is bounded above by $L([\max(\mathbf{t}_0) + i \max(\mathbf{T})], \max(\delta))$. Multiplying by $|J|-1$ areas, we get an upper bound for $q(s_{i-1}, d_i)$. Thus for the entire schedule, a strict upper bound is:

$$Q_D < (|J|-1) \sum_{i=1}^{k} \gamma^{i-1} L([\max(\mathbf{t}_0) + i \max(\mathbf{T})], \max(\delta)).$$

V. Experimental Results

To validate the performance of the proposed strategy, we ran experiments with a 2D robotic simulator, Stage [42], which includes motion and sensor noise. We implemented our strategy in Python within the Robot Operating System (ROS) and made it publicly available[1].

A. Setup

We considered three realistic environments that present different characteristics in terms of connectivity and openness. In particular, we took Office and Open from the Radish repository [43], which are called "sdr site b" and "acapulco convention center", respectively. Cluttered is from the robotic simulator, MRESim, repository [44], called "grass". The map sizes are $80\,\mathrm{m}$ by $60\,\mathrm{m}$.

The differential-drive robot (with LiDAR, $v = 1m/s$, and $\tilde{b} = 100$), aims to preserve each F_j, (with $F_j = 100$ at $t_j = 0$ and follows an exponential decay function), from falling below $z = 50$. The restoration rates for an area and charging up are $\rho_x = 0.10$ and $\rho_y = 0.10$, respectively. Sampling from initial runs, we accounted for $\varepsilon = 0.25$.

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

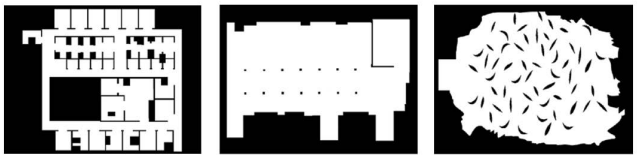


Fig. 3: Environments for the experiments: Office (left), Open (middle), Cluttered (right).

Scenario 1 (S1): Office and Cluttered. We designed scenarios to highlight obvious consequences of committed visits: we divided the more challenging maps, in terms of obstacles, into quadrants that have increasingly faster decays, wherein each quadrant we randomly picked a location for preservation. For Q1 (top-right) the $\delta_1 \in (0.00100, 0.00125]$; Q2 (top-left), $\delta_2 \in (0.00150, 0.00175]$; Q3 (bottom-L), $\delta_3 \in (0.00200, 0.00225]$; and Q4 (bottom-R), $\delta_4 \in (0.00230, 0.00255)$. The schedule length is $k = 4$, which allows for full coverage of the 4 areas.

Scenario 2 (S2): All maps. For evaluate the generalizability of the results, we randomly placed 9 areas, wherein one-third have $\delta \in (0.00100, 0.00125]$; another one-third have $\delta \in (0.00150, 0.00175]$; and the other third have $\delta \in (0.00200, 0.00225]$. The schedule is only capable of partial coverage with $k = 3$.

With these decay rates the restoration task is non-trivial, an area will hit the critical threshold from max measure without intervention about three to eight times within the time frame of our experiments–35 minutes each.

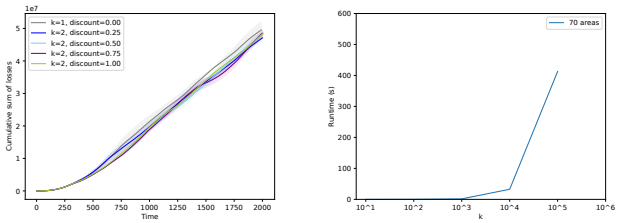
We randomized 5 placements of areas. We repeated each experiment three times—results over repetitions showed similar trend with small variations. We ran the experiments on a Intel i7-10750H CPU at 2.96GHz and 16GB RAM computer.

We implemented the following informed strategies: tree search (indexed T), deemed as optimal baseline, presented in Section IV-B; our proposed heuristic method, Algorithm 1 (H); and a recent proposed method solving a spatio-temporal orienteering problem [6] (DP); (k is suffixed next to the method index). We also ran an uninformed baseline, a random strategy (R), which selects a random area to visit at each decision step.

In running our proposed method, we observed the computed opportunity cost for every decision made to be within the strict bounds discussed previously.

B. Tuning k and γ for Algorithm 1

Before presenting the full results, we discuss the effects of different values for k and γ. We found that the myopic approach ($k = 1$) can already be effective. But by considering a non-myopic approach, the performance may be improved, (see Fig. 4a). The tuned lengths: $k = 4$ for both S1 Office and Cluttered; $k = 9, 12, 12$ for S2 Office, Cluttered, and Open, respectively. While, the tuned $\gamma = 0.25$. We found this conservative discount as the most stable, as we are just



(a) Tuning γ for non-myopic k: S1 Office (b) Scalability

Fig. 4: (a) Cumulative sum of losses of (H) for myopic and non-myopic approach over several trials and (b) scalability of Algo. 1 for a max number of areas the simulator could load.

assuming a fixed ε in our non-myopic computations when in actual runs the noise would vary.

C. Discussion of results

Routing behavior: We observed that the informed strategies would directly visit the fast decaying areas, given the optimization is based on loss. Notably, (T) exhibits prioritizing fast decaying areas over lesser urgent areas, revisiting the former and only visits the latter when their decays have become significant (e.g., a schedule [4, 3, 2, 4] employed in S1 Office). (H) can mimic this behavior. (DP) however, due to its constraint of visiting an area at most once, goes through the lesser urgent areas despite the urgency in the fast decaying ones (i.e., [4, 1, 3, 2]). Furthermore, the difference in objective between (H) and (DP), thereby in decision-making, contributes to the former outperforming the latter, as will be highlighted in the next subsection. We also observed that (T) sometimes charges the battery early, foregoing restoring a decaying area, but leads to a better management of the areas in the long run.

Performance: To evaluate performance, we measure the total losses incurred from all areas for the duration of the experiment. We construct a histogram of relative ratios of total losses between a method and (T), wherein the corresponding dashed horizontal line marks the average ratio. A relative ratio that is above 1 means that the total losses of the method is greater than (T), and the higher the ratio the greater the magnitude. If one method's relative ratio is greater than that of another, it follows that the latter performs better. Moreover, we show the average F_j, with 95% confidence intervals, whether an area is being preserved from z, on average (Figs. 5 and 6). In addition, we tabulate the average duration an area is decaying from z before it gets restored, (Table I). We notice that in some entries the standard deviation is wide, as the average is across the areas. The s.d. can be interpreted as an upper range of the number of visits to other areas that were made before that area is restored. For example, in S1 Office it takes around 50 secs to visit one area from another, on average. And so, for T4 with average 45 secs and s.d. of 52 secs suggests that it takes about 1 to 2 other visits before a robot restores an area. Also, the number of times areas are below the threshold is consistently higher for (DP).

TABLE I: Average duration (secs) an area is decaying from z prior to being restored. H9 is the tuned k for S2 Office; H12 for S2 Cluttered and Open. On average, the duration that F_j stays below z is shorter in (H4) than in (DP4). By tuning k for (H), this duration can be further improved.

S1	**T4**	**H4**	**DP4**
Office	45 (sd=52)	51 (45)	112 (43)
Cluttered	20 (17)	27 (15)	84 (68)

S2	**T4**	**H4**	**H9 (or H12)**	**DP4**
Office	72 (sd=51)	153 (88)	80 (72)	223 (174)
Cluttered	65 (41)	81 (70)	49 (34)	266 (144)
Open	0 (0)	0 (0)	0 (0)	17 (0)

As we would expect, (R) would perform poorly compared to the informed strategies in preserving the areas. Also, the results show that (H) performs closely well relative to (T) and it performs better than (DP) overall at the end of the mission in all of the cases. The informed strategies are able to preserve F_j from z, on average; notably, (H) preserves it by a higher margin than (DP), and the duration of an area's property being below z before it gets restored is shorter in (H) than in (DP). Furthermore, by tuning k the performance of (H) can be improved.

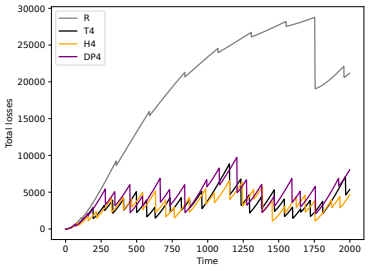
Further experiments: We conducted more experiments modifying key settings: 1) we ran in S1 increasing the number of areas to 12, still ensuring areas are spread across the map; 2) we then slowed down the decay rates of all areas to be within $(0.00100, 0.00125]$; we ran in S1 with 4 areas, then with 12 areas. In all of these settings, we observed similar trend of (H) outperforming (DP).

Runtime: Computationally, (T) is not practical: even for 4 areas with $k = 8$ or 9 areas with $k = 6$, the algorithm runs beyond 30 minutes. To test the computational limit of (H), we timed the decision it makes on the max number of J that could be loaded on the computer while increasing the forecast length k (see Fig. 4b). Even for $k = 10^5$, a decision can be made in about 7 mins. Thus, the algorithm scales well. We observed however as J increases, the robot becomes less effective in preserving the areas, as the task has become more demanding for a single robot.

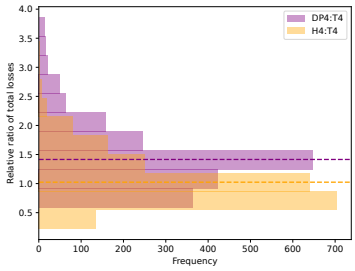
VI. Conclusion and Future Work

We modeled the problem of persistently restoring a temporal environmental property as a search problem, presenting an optimal strategy that solves it, and then a practical heuristic algorithm that evaluates the discounted opportunity costs of the next immediate visits. Our experiments showed the applicability of the proposed method in preserving the temporal properties from falling below the critical threshold, performing better than other existing approaches and closer to the optimal.

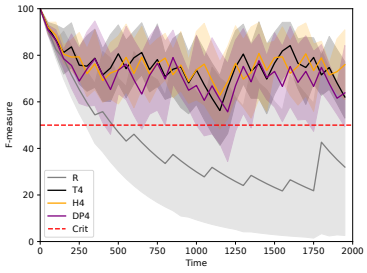
Our future work will extend the problem to include uncertainty in the formulation and the algorithm to multiple robots, as well as connecting to other planning strategies, such as Monte Carlo Tree Search. An effective and efficient strategy in preserving environmental properties presents positive benefits in many important and pressing societal applications, especially pollution management.



(a) Total losses: S1 Office

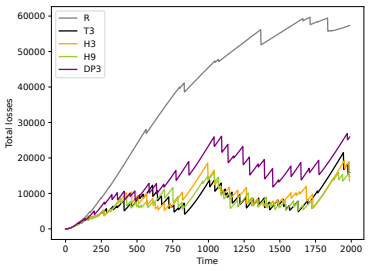


(b) Relative ratio: S1 Office

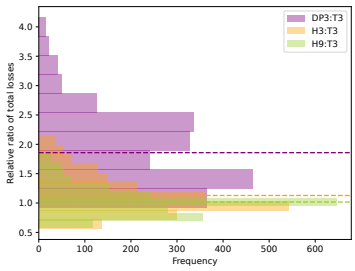


(c) Average F_j: S1 Office

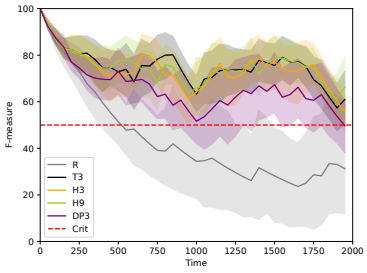
Fig. 5: S1 results: (a)-(b) (H4) performs closely well relative to (T4) and better than (DP4) overall at the end of mission. Relative ratios greater than 1, especially the larger ratios, are more frequent in (DP4) than in (H4). And that, the average ratio of (DP4), 1.41 (sd=0.55), is greater than that of (H4), 1.02 (sd=0.41). (c) On average, (H4) preserves F_j from z by a higher margin than (DP4). Results for S1 Cluttered are similar.



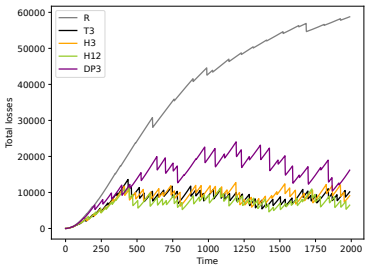
(a) Total losses: S2 Office



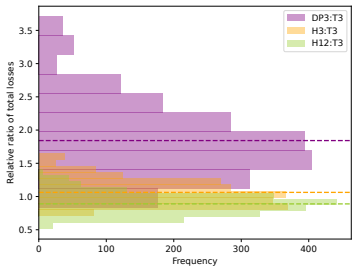
(b) Relative ratio: S2 Office



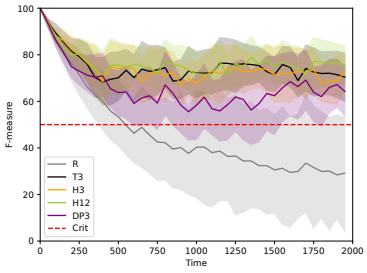
(c) Average F_j: S2 Office



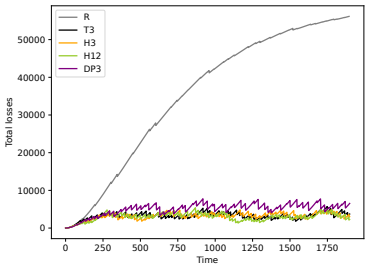
(d) S2 Cluttered



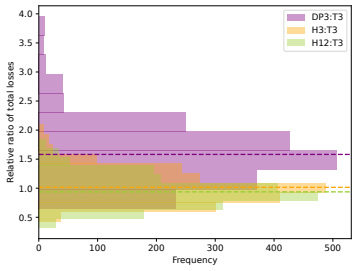
(e) S2 Cluttered



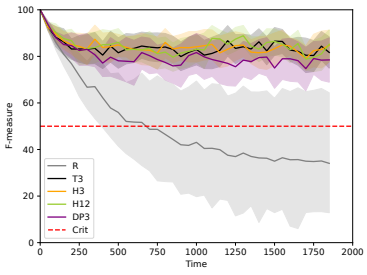
(f) S2 Cluttered



(g) S2 Open



(h) S2 Open



(i) S2 Open

Fig. 6: S2 results: (a)-(b), (d)-(e), (g)-(h) (H) performs better than (DP) and closer to (T) overall at the end of mission. (c), (f), (i) On average, (H) preserves an area better than (DP) by a higher margin. By tuning k, performance of (H) can be improved.

ACKNOWLEDGEMENT

This work is supported in part by the Burke Research Initiation Award and NSF CNS-1919647, 2144624, OIA-1923004. Acknowledgements to Kizito Masaba and Ankita Sarkar for their listening ear in discussing ideas as the sections of this paper were being refined.

REFERENCES

[1] H. Christensen, N. Amato, H. Yanco, *et al.*, "A roadmap for us robotics–from internet to robotics 2020 edition", *Foundations and Trends® in Robotics*, vol. 8, no. 4, pp. 307–424, 2021.

[2] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics", *Robotics and Autonomous systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[3] A. Quattrini Li, "Exploration and mapping with groups of robots: Recent trends", *Current Robotics Reports*, vol. 1, pp. 227–237, 2020.

[4] S. Bai, T. Shan, F. Chen, L. Liu, and B. Englot, "Information-driven path planning", *Current Robotics Reports*, vol. 2, no. 2, pp. 177–188, 2021.

[5] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey", *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.

[6] Z. Ma, K. Yin, L. Liu, and G. S. Sukhatme, "A spatio-temporal representation for the orienteering problem with time-varying profits", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 6785–6792.

[7] B. Yamauchi, "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*, 1997, pp. 146–151.

[8] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration", *IEEE Transactions on robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[9] N. Basilico and F. Amigoni, "Exploration strategies based on multi-criteria decision making for searching environments in rescue operations", *Autonomous Robots*, vol. 31, pp. 401–417, 2011.

[10] H. Choset, "Coverage of known spaces: The boustrophedon cellular decomposition", *Autonomous Robots*, vol. 9, pp. 247–253, 2000.

[11] A. Xu, C. Viriyasuthee, and I. Rekleitis, "Efficient complete coverage of a known arbitrary environment with applications to aerial operations", *Autonomous Robots*, vol. 36, pp. 365–381, 2014.

[12] M. Schwager, P. Dames, D. Rus, and V. Kumar, "A multi-robot control policy for information gathering in the presence of unknown hazards", in *Robotics Research: The 15th International Symposium ISRR*, Springer, 2017, pp. 455–472.

[13] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms", *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1271–1287, 2014.

[14] Y. Kantaros, B. Schlotfeldt, N. Atanasov, and G. J. Pappas, "Asymptotically optimal planning for non-myopic multi-robot information gathering.", in *Robotics: Science and Systems*, 2019, pp. 22–26.

[15] R. N. Smith, M. Schwager, S. L. Smith, B. H. Jones, D. Rus, and G. S. Sukhatme, "Persistent ocean monitoring with underwater gliders: Adapting sampling resolution", *Journal of Field Robotics*, vol. 28, no. 5, pp. 714–741, 2011.

[16] N. R. Lawrance, J. J. Chung, and G. A. Hollinger, "Fast marching adaptive sampling", *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 696–703, 2017.

[17] N. Basilico and S. Carpin, "Balancing unpredictability and coverage in adversarial patrolling settings", in *Algorithmic Foundations of Robotics XIII: Proceedings of the 13th Workshop on the Algorithmic Foundations of Robotics 13*, Springer, 2020, pp. 762–777.

[18] A. Kolling and S. Carpin, "Multi-robot surveillance: An improved algorithm for the graph-clear problem", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 2360–2365.

[19] J. Liu, M. Rangwala, K. S. Ahluwalia, *et al.*, "Intermittent deployment for large-scale multi-robot forage perception: Data synthesis, prediction, and planning", *IEEE Transactions on Automation Science and Engineering*, pp. 1–21, 2022.

[20] J. Song, S. Gupta, J. Hare, and S. Zhou, "Adaptive cleaning of oil spills by autonomous vehicles under partial information", in *MTS/IEEE OCEANS-San Diego*, 2013, pp. 1–5.

[21] X. Jin and A. Ray, "Navigation of autonomous vehicles for oil spill cleaning in dynamic and uncertain environments", *International Journal of Control*, vol. 87, no. 4, pp. 787–801, 2014.

[22] B. Li, B. Moridian, and N. Mahmoudian, "Autonomous oil spill detection: Mission planning for asvs and auvs with static recharging", in *OCEANS 2018 MTS/IEEE Charleston*, 2018, pp. 1–5.

[23] S. S. Ge and C.-H. Fua, "Complete multi-robot coverage of unknown environments with minimum repeated coverage", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2005, pp. 715–720.

[24] E. Stump and N. Michael, "Multi-robot persistent surveillance planning as a vehicle routing problem", in *IEEE International Conference on Automation Science and Engineering (CASE)*, 2011, pp. 569–575.

[25] P. Fazli, A. Davoodi, and A. K. Mackworth, "Multi-robot repeated area coverage", *Autonomous robots*, vol. 34, pp. 251–276, 2013.

[26] E. Plaku and S. Karaman, "Motion planning with temporal-logic specifications: Progress and challenges", *AI communications*, vol. 29, no. 1, pp. 151–162, 2016.

[27] M. Lahijanian, M. R. Maly, D. Fried, L. E. Kavraki, H. Kress-Gazit, and M. Y. Vardi, "Iterative temporal planning in uncertain environments with partial satisfaction guarantees", *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 583–599, 2016.

[28] H. Kress-Gazit, G. E. Fainekos, and G. J. Pappas, "Temporal-logic-based reactive mission and motion planning", *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.

[29] A. Ulusoy, S. L. Smith, X. C. Ding, and C. Belta, "Robust multi-robot optimal path planning with temporal logic constraints", in *IEEE International Conference on Robotics and Automation (CASE)*, 2012, pp. 4693–4698.

[30] M. Cirillo, L. Karlsson, and A. Saffiotti, "Human-aware task planning: An application to mobile robots", *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 1, no. 2, pp. 1–26, 2010.

[31] P. Schillinger, M. Bürger, and D. V. Dimarogonas, "Simultaneous task allocation and planning for temporal logic goals in heterogeneous multi-robot systems", *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 818–838, 2018.

[32] G. P. Strimel and M. M. Veloso, "Coverage planning with finite resources", in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 2950–2956.

[33] M. Wei and V. Isler, "Coverage path planning under the energy constraint", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 368–373.

[34] S. Mishra, S. Rodriguez, M. Morales, and N. M. Amato, "Battery-constrained coverage", in *IEEE International Conference on Automation Science and Engineering (CASE)*, 2016, pp. 695–700.

[35] B. Araki, J. Strang, S. Pohorecky, C. Qiu, T. Naegeli, and D. Rus, "Multi-robot path planning for a swarm of robots that can both fly and drive", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5575–5582.

[36] H. Zhang and Y. Vorobeychik, "Submodular optimization with routing constraints", *AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[37] S. M. LaValle, *Planning algorithms*. Cambridge University Press, 2006.

[38] R. G. Lamb and J. H. Seinfeld, "Mathematical modeling of urban air pollution. general theory", *Environmental Science & Technology*, vol. 7, no. 3, pp. 253–261, 1973.

[39] D. Brown, J. Bridgeman, and J. R. West, "Predicting chlorine decay and thm formation in water supply systems", *Reviews in Environmental Science and Bio/Technology*, vol. 10, pp. 79–99, 2011.

[40] S. V. Liu, F.-l. Chen, and J. Xue, "A meta-analysis of selected near-road air pollutants based on concentration decay rates", *Heliyon*, vol. 5, no. 8, 2019.

[41] J. L. Repace, W. R. Ott, and N. E. Klepeis, "Indoor air pollution from cigar smoke", *Smoking and Tobacco Control Monograph 9. Cigars—Health Effects and Trends*, pp. 161–179, 1998.

[42] R. Vaughan, "Massively multi-robot simulation in stage", *Swarm intelligence*, vol. 2, pp. 189–208, 2008.

[43] A. Howard, "The robotics data set repository (radish)", *http://radish. sourceforge. net/*, 2003.

[44] V. Spirin, S. Cameron, and J. De Hoog, "Time preference for information in multi-agent exploration with limited communication", in *Towards Autonomous Robotic Systems: 14th Annual Conference (TAROS)*, Springer, 2014, pp. 34–45.