Persistent Monitoring of Events With Stochastic Arrivals at Multiple Stations

Jingjin Yu, *Member, IEEE*, Sertac Karaman, *Member, IEEE*, and Daniela Rus, *Fellow, IEEE*

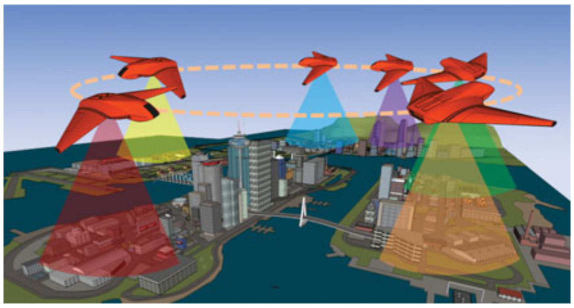
***Abstract*—This paper introduces a new mobile sensor scheduling problem involving a single robot tasked to monitor several events of interest that are occurring at different locations (stations). Of particular interest is the monitoring of transient events of a stochastic nature, with applications ranging from natural phenomena (e.g., monitoring abnormal seismic activity around a volcano using a ground robot) to urban activities (e.g., monitoring early formations of traffic congestion using an aerial robot). Motivated by examples like these, this paper focuses on problems in which the precise occurrence times of the events are unknown *a priori*, but statistics for their interarrival times are available. In monitoring such events, the robot seeks to: 1) maximize the number of events observed and 2) minimize the delay between two consecutive observations of events occurring at the same location. This paper considers the case when a robot is tasked with optimizing the event observations in a balanced manner, following a cyclic patrolling route. To tackle this problem, first, assuming that the cyclic ordering of stations is known, we prove the existence and uniqueness of the optimal solution and show that the solution has desirable convergence rate and robustness. Our constructive proof also yields an efficient algorithm for computing the unique optimal solution with $O(n)$ time complexity, in which n is the number of stations, with $O(\log n)$ time complexity for incrementally adding or removing stations. Except for the algorithm, our analysis remains valid when the cyclic order is unknown. We then provide a polynomial-time approximation scheme that computes for any $\epsilon > 0$ a $(1 + \epsilon)$-optimal solution for this more general, NP-hard problem.**

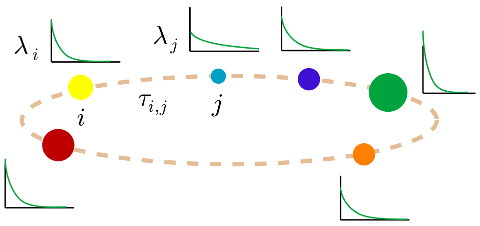
***Index Terms*—Optimization, persistent monitoring, Poisson process, surveillance, stochastic events.**

I. Introduction

An avid documentary maker would like to observe several species of birds. Each species can be seen only at a particular location. Unfortunately, it is impossible to predict when exactly a bird will be seen at a sighting location. Hence, the documentary maker must wait in a hiding spot for the birds to appear. To the advantage of our documentary maker, past experience has furnished her with statistics of sighting times for each location. Given this information, the documentary maker would like to split her time between the locations, waiting to capture photos of bird sightings. While splitting her time, the documentary maker has two objectives. First, she would like to maximize the number of sightings. Second, she would like to minimize the delay between two consecutive sightings of the same species. Most importantly, our documentary maker is committed to striking a balance among the species. In other words, she would like to maximize the number of sightings and minimize the delay between two consecutive sightings, for all species all at the same time.

The bird documentary maker example captures the essential elements of the problem studied in this paper. More formally, consider a single robotic vehicle tasked with monitoring stochastic and transient events that occur at multiple locations (see, e.g., Fig. 1). Unable to predict exactly when an event happens, the robot must travel to a particular location and wait for the event to occur. Limited by a single robot, its schedule[1] must be


(a)



(b)

Fig. 1. (a) One of many potential applications of our persistent monitoring formulation, in which an UAV (robot) is given the task of continuously observing randomly occurring events at a set of fixed locations (the surface areas under the cones). The sizes of the discs represent the relative arrival rates of stochastic events at the locations. (b) Illustration of the underlying geometric problem setting. At each point of interest, say location (station) i, events arrive following a Poisson process with intensity λ_i. It takes a robot $\tau_{i,j}$ time to move from station i to station j, during which no observation can be made. The associated plots roughly capture the (exponential) distributions of event arrivals associated with the stations.

Manuscript received September 4, 2014; accepted February 28, 2015. Date of publication March 26, 2015; date of current version June 3, 2015. This paper was recommended for publication by Associate Editor V. Isler and Editor D. Fox upon evaluation of the reviewers' comments. This work was supported by ONR projects N00014-12-1-1000 and N00014-09-1-1051, the Singapore–MIT Alliance on Research and Technology (SMART) Future of Urban Mobility project, and the AFOSR Dynamic Data-Driven Application Systems Program under Grant FA9550-14-1-0399.

J. Yu and D. Rus are with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: jingjin@csail.mit.edu; rus@csail.mit.edu).

S. Karaman is with the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: sertac@mit.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2015.2409453

[1]In this paper, *schedule* and *policy* are used interchangeably.

optimized to ensure that all locations are observed equally well as best as possible, i.e., in a balanced manner, according to the following objectives: 1) ensure that a large number of events are observed at each location; and 2) ensure that the delay between two observations of events at any given location is minimized. Optimizing these objectives in a balanced manner gives rise to a multiobjective mobile sensor scheduling problem. This paper is concerned with the mathematical analyses and algorithmic approaches for this complex multiobjective optimization problem.

The problem we study is applicable to a broad set of practical scenarios, including surveillance and reconnaissance, and scientific monitoring. The events of interest include natural phenomena (e.g., volcanic eruptions and early formations of blizzards, hailstorms, and tsunamis), biological disasters (e.g., early formations of epidemic diseases on animal or plant populations), military operations (e.g., terrorist attacks), among others. The key common characteristic of these events is that their precise time of occurrence cannot be easily forecast, although the statistics regarding how often they occur may be available, for example, from past experience. Hence, the data-collecting robot must wait at the location of interest to capture the event once it occurs. Then, the fundamental scheduling problem is to decide how much time the robot should spend at each location to achieve various objectives, such as those described above. Our main theoretical result is that this complex multiobjective mobile sensor scheduling problem can be reduced to a quasi-convex optimization problem, which implies efficient algorithms for computing optimal solutions. In particular, globally optimal or near-optimal solutions can be computed in time polynomial in the number of locations.

Related work: Broadly speaking, persistent monitoring problems appear naturally whenever only limited resources are available for serving a set of spatially dispersed tasks. Motivated by a variety of potential applications, such as aerial [1] and underwater [2] data collection, many researchers have studied persistent monitoring problems [3]–[11]. In [3], the authors consider a weighted latency measure as a robot continuously traverses a graph, in which the vertices represent the regions of interest and the edges between the vertices are labeled with the travel time. They present an $O(\log n)$-approximation algorithm for the proposed problem. In [4], a memoryless control policy is designed to guide robots modeled as controllable Markov chains to maximize their monitoring area while avoiding hazardous areas. In [5], the authors consider a persistent monitoring problem for a group of agents in a 1-D mission space. They show that this problem can be solved by parametrically optimizing a sequence of switching locations for the agents.

The coordination and surveillance problem for multiple unmanned aerial vehicles is addressed in [6] and [9]. Coordination among aerial and ground vehicles is further explored in [7]. A random sampling method that generates optimal cyclic trajectories for monitoring Gaussian random field is presented in [8]. The problem of generating speed profiles for robots along predetermined closed paths for keeping bounded a varying field is addressed in [10]. The authors characterized policies for both single and multiple robots. In [11], decentralized adaptive controllers were designed to morph the initial closed paths of robots to focus on regions of high importance.

In contrast with the references cited above, the problem studied in this paper focuses on transient events at discrete locations, emphasizing unknown arrival times (but known statistics). Since an event is only observable at discrete locations, the event arrival times being unknown forces the robot to wait at each station to observe the events of interest. Waiting at a station then introduces delay at other stations. This stochastic event model links our work to stochastic vehicle routing problems such as the dynamic traveling repairman problem studied in [12] and [13], among others. Whereas the problem we consider can be viewed as a vehicle routing problem, our main focus is not on capturing all events, but rather collecting a reasonable amount of events across the locations in a balanced manner, penalizing large gaps between observations at the same location.

Persistent surveillance problems are intimately connected to coverage problems. Coverage of a 2-D region has been extensively studied in robotics [14]–[16], as well as in purely geometric settings. For example, in [17], the proposed algorithms compute the shortest closed routes for continuous coverage of polygonal interiors under an infinite visibility sensing model. Coverage with limited sensing range was also addressed [18], [19]. If the environment to be monitored has a 1-D structure, discrete optimization problems, such as the traveling salesmen problem (TSP), often arise [3]. In most coverage problems, including those cited above, the objective is to place sensors in order to maximize, for example, the area that is within their sensing region. The persistent surveillance problem we study in this paper is a special case, in which limited number of sensors do not allow extensive coverage; hence, we resort to mobility in order to optimize the aforementioned performance metrics.

Persistent monitoring problems are also related to (static) *sensor scheduling* problems (see, e.g., [20]–[22]), which are usually concerned with scheduling the activation times of sensors in order to maximize the information collected over a time-varying process. The problem considered in this paper involves a *mobile sensor* that can travel to each of the locations, in which the additional time required to travel between stations is nonzero. The mobile sensor scheduling literature is also rich. For instance, in [23], the authors study the control of a robotic vehicle in order to maximize data rate, while collecting data stochastically arriving at two locations. The problem studied in this paper is a novel mobile sensor scheduling problem involving several locations and a multiobjective performance metric that includes both the data rate and the delay between consecutive observations.

Contributions: The contributions of this paper can be summarized as follows. First, we propose a novel persistent monitoring and data collection problem, with the unique feature that the precise arrival times of events are unknown *a priori*, but their statistics are available. Combined with the assumption that the events are generated at distributed discrete locations, the stochastic event model allows our formulation to encompass many practical applications in which the precise occurrence times of the events of interest cannot be forecast easily. Second, we prove that this fairly complex multiobjective mobile sensor

TABLE I
LIST OF FREQUENTLY USED SYMBOLS AND THEIR INTERPRETATIONS

λ_i	Arrival rate of the Poisson process at station i
$\tau_{i,j}$	Travel time from station i to station j
π	Cyclic policy of the form $((k_1, t_1), \ldots, (k_n, t_n))$, in which t_i is the time spent by the robot at station k_i, $1 \leq k_i \leq n$, in one policy cycle, or of the form $(t_1, \ldots, t_n)$ when $k_i = i$
$J_i(\pi)$	An objective function to be optimized
T	Total time incurred by a policy cycle
T_{tr}	Total travel time per policy cycle
T_{obs}	$T - T_{\mathrm{tr}}$, total observation time per policy cycle
σ	$1/(\sum_{i=1}^{n}(1/\lambda_i))$, the harmonic sum of λ_i's
γ_i	$\sigma/\lambda_i = 1/(\lambda_i \sum_{j=1}^{n}(1/\lambda_j))$
$N_i(\pi)$	The number of events collected at station i in one period of the policy π
$T_i(\pi)$	The time between two consecutive event observations at station i containing travel to other stations, for the policy π
$Pr(e)$	Probability of an event e
$\mathbb{E}[X]$	Expected value of a random variable X
$\alpha_i(\pi)$	$\mathbb{E}[N_i(\pi)]/\sum_{j=1}^{n}\mathbb{E}[N_j(\pi)]$
Π	$\arg\max_\pi \min_i \alpha_i(\pi)$

scheduling problem admits a unique globally optimal solution in all but rare degenerate cases. The optimal solution is also shown to have desirable convergence property and robustness. Moreover, the unique policy can be computed extremely efficiently when the station visiting order is predetermined. The policy can also efficiently computed to be $(1+\epsilon)$-optimal, for arbitrarily small $\epsilon > 0$, when the visiting order is not given *a priori*. At the core of our analysis is a key intermediate result that reduces the mobile sensor scheduling problem to a quasi-convex optimization problem in one variable, which is of independent interest.

This paper builds on [24] and significantly generalizes it in the following aspects: 1) In addition to existence, solution uniqueness is now established; 2) convergence and robustness results are introduced and thoroughly discussed to render the study more complete; 3) a polynomial-time approximation algorithm is provided that solves the more general problem when the cyclic ordering of stations is unknown *a priori*; and 4) extensive computational experiments are added to confirm our theoretical development as well as to provide insights into the structure of the proposed optimization problem.

The rest of this paper is organized as follows. In Section II, we provide a precise definition of the multiobjective persistent monitoring problem that we study. Starting with the assumption that the stations' cyclic order is known, we prove existence and uniqueness of optimal solutions to this slightly restricted problem in Section III. We further explore the convergence and robustness properties of optimal solutions in Section IV. In Section V, we deliver algorithmic solutions for the multiobjective optimization problem with and without a predetermined station visiting order and characterize their computational complexity. We present and discuss computational experiments in Section VI and conclude the paper in Section VII. Frequently used symbols are listed in Table I.

II. Problem Statement

Consider a network of n *stations* or *sites* that are spatially distributed in $\mathbb{R}^2$. At each station, interesting but *transient* events may occur at unpredictable time instances. The arrival times of events at a station i, $1 \leq i \leq n$, are assumed to follow a Poisson process with a known (mean) *arrival rate* or *intensity* λ_i, with a unit of number of events per hour. The event arrival processes are assumed to be independent between two different stations. Let there be a mobile robot that travels from station to station. The robot is equipped with on-board sensors, such as cameras, that allow the robot to record data containing the stochastic events occurring at the stations. Let $\tau_{i,j}$ denote the time it takes the robot to travel from station i to station j. We assume that $\tau_{i,j}$ is proportional to the Euclidean distances between stations i and j.

We want to design *cyclic policies* to enable optimal data collection, according to the objectives described in the introduction. A precise definition of these objectives will follow shortly. In a cyclic policy, the robot visits the stations in a fixed (but unknown *a priori*) cyclic order and waits at each station for a fixed amount of time to collect data. The solution scheduling policy then takes the form $\pi = ((k_1, t_1), \ldots, (k_n, t_n))$ in which k_i's specify the visiting order of the robot and t_i describes the waiting time of the robot at station k_i.

Remark. Having fixed waiting time suggests that the policy is an open-loop (i.e., no feedback) policy. We note that such policies are of practical importance. For example, it may be the case that an aerial mobile robot only has limited energy or computing power to process the data (e.g., a large number of video streams) it collects. Similarly, an underwater robot gathering plankton samples may not have on-board equipment to analyze the collected samples. As another example, the transportation of the data collection equipment can be a nontrivial task, which requires that the travel schedule to be prearranged. In yet another example, in certain scenarios, it may even be desirable not to allow the robot to have immediate semantic understanding of the collected data due to security reasons such as hacking prevention. △

Given a policy π, we define its *period* as

$$T := \sum_{i=1}^{n-1} \tau_{k_i,k_{i+1}} + \tau_{k_n,k_1} + \sum_{i=1}^{n} t_i.$$

For convenience, let $T_{\mathrm{tr}} := \sum_{i=1}^{n-1} \tau_{k_i,k_{i+1}} + \tau_{k_n,k_1}$ be the total travel time per policy cycle and $T_{\mathrm{obs}} := \sum_{i=1}^{n} t_i = T - T_{\mathrm{tr}}$ be the total observation time per policy cycle. Let $N_i(\pi)$ denote the number of events observed at station k_i in one cycle. For the first objective, seeking to ensure maximal and equal priorities are allocated to all stations, we maximize the *fraction of events observed at each station in a balanced manner*, i.e.,

$$J_1(\pi) = \min_i \ \alpha_i(\pi) = \min_i \ \frac{\mathbb{E}[N_i(\pi)]}{\sum_{j=1}^{n}\mathbb{E}[N_j(\pi)]} \tag{1}$$

subject to the additional constraint

$$\pi \in \underset{\pi'}{\operatorname{argmin}} \max_{i,j} |\mathbb{E}[N_i(\pi')] - \mathbb{E}[N_j(\pi')]| \tag{2}$$

which further balances event observation efforts by penalizing large observation discrepancies between different stations. Alternatively, one may view (2) as a higher order effort aimed at balancing event observation than simply maximizing J_1.

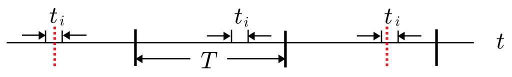


Fig. 2. Illustration of an *instance* or a *sample* of a *delay*. The policy has a period of T and the robot is waiting at station k_i during the intervals with length t_i. The two dotted lines correspond to times when events occur at station k_i. If no other events happen at station k_i in between these two times when the robot is present at station k_i, then the time between these two event occurrences is an instance of the delay at station k_i.

The second objective seeks to minimize large delays between event observations at the same station. We formalize the notion of *delay*, a random variable, as follows. As the robot executes a policy, it arrives at station k_i periodically and waits for t_i time at station k_i each time it gets there. Suppose that during one such waiting time t_i, one or more events occur at station k_i. Let the last event within this particular t_i be t_{start}. An instance of a delay is a period of time that begins at t_{start} and ends when another event occurs at station k_i, while the robot is waiting at station k_i (see, e.g., Fig. 2). More precisely, we define the delay at station k_i for a given policy π, denoted $T_i(\pi)$, as a random variable that maps these instances of delays to probability densities. Our second objective aims at minimizing the *maximum delay across all stations*, i.e.,

$$J_2(\pi) = \max_i \ \mathbb{E}[T_i(\pi)]. \tag{3}$$

Often, both objectives are equally important. One would like to spend as much time as possible at all stations for maximizing the data collection effort and at the same time minimize delays between observations at any given station, which becomes large if the robot lingers at any station for too long. Interestingly, the set of policies that optimizes the first objective function is not unique; in fact, there are infinitely many such policies. We compute the optimal policy for the second objective function among those policies that optimize the first objective function. That is, we compute the policy $\pi^* = \arg\min_{\pi\in\Pi} J_2(\pi)$, with $\Pi := \arg\max_{\pi'} J_1(\pi')$, subject to (2). We then further show that π^* is the *unique* Pareto optimal solution for optimizing both J_1 and J_2.

With the setup so far, we now formally state the persistent monitoring problem studied in this paper.

Problem 1. Given n, $\{\lambda_i\}$, and $\{\tau_{i,j}\}$, find an optimal solution $\pi^* = ((k_1^*, t_1^*), \ldots, (k_n^*, t_n^*))$ that optimizes J_1 and J_2 subject to the constraint equation (2).

To facilitate our analysis, we begin with a special case in which the visiting order of stations is fixed *a priori*. That is, we assume without loss of generality that $k_i = i$.

Problem 2. Given n, $\{\lambda_i\}$, and $\{\tau_{i,j}\}$, and assume that the robot visits the n stations in the cyclic order of $1, 2, \ldots, n$, find an optimal solution $\pi^* = (t_1^*, \ldots, t_n^*)$ that optimizes J_1 and J_2 subject to the constraint equation (2).

It is straightforward to observe that the travel times only matter as a whole, i.e., the policy's dependence on the robot's path in a cyclic policy only hinges on T_{tr}. The immediate gain from proposing Problem 2 is that it removes the need to compute a TSP tour, allowing us to focus our study on the optimality structure induced by J_1 and J_2, which is fairly rich. We dedicate Sections III and IV to Problem 2 and revisit Problem 1 when we discuss algorithmic solutions in Section V.

III. Existence and Uniqueness of Optimal Policy

In this section, we establish the existence and uniqueness of solutions for Problem 2. We also simply refer to a policy π as $\pi = (t_1, \ldots, t_n)$ here and in Section IV. We note that the results from this section go beyond simply showing the existence and uniqueness of an optimal cyclic policy for the robot; an effective means for computing such a policy is also implied. The discussion of the implied algorithmic solution is deferred to Section V.

A. Existence of Optimal Solution

We now establish the existence of optimal solutions to Problem 2. Showing the existence of solution to a multiobjective optimization problem requires showing that the Pareto front is not empty. We achieve this goal through Theorem 1, which works with the two objectives sequentially. Theorem 1 does more than simply showing the Pareto front is nonempty; it actually describes an optimization program that finds a point on the Pareto front.

Theorem 1 (Existence of Optimal Solution). There exists a continuum of policies that maximize J_1 under the constraint (2), given by

$$\Pi := \operatorname*{argmax}_{\pi} \min_i \ \frac{\mathbb{E}[N_i(\pi)]}{\sum_{j=1}^n \mathbb{E}[N_j(\pi)]}.$$

Among all policies in Π, there is a unique policy π^* that minimizes J_2. Moreover, this unique policy $\pi^* = (t_1^*, t_2^*, \ldots, t_n^*)$ is determined by

$$t_i^* = \frac{\sigma}{\lambda_i} T_{\text{obs}}^*$$

in which

$$T_{\text{obs}}^* := \operatorname*{argmin}_{T_{\text{obs}}} \left(\frac{2}{\lambda_{\max}} + \frac{(T_{\text{obs}} + T_{\text{tr}})\lambda_{\max} - \sigma T_{\text{obs}}(1 + e^{-\sigma T_{\text{obs}}})}{(1 - e^{-\sigma T_{\text{obs}}})\lambda_{\max}} \right)$$

with $\lambda_{\max} = \max_i \lambda_i$ being the maximum arrival rate and $\sigma = \left(\sum_{i=1}^n \lambda_i^{-1}\right)^{-1}$ the harmonic sum of λ_i's.[2] The optimization problem that gives T_{obs}^* is a quasi-convex program in one variable, the unique optimal solution for which can be computed efficiently, for example, by using the Newton–Raphson method to compute the root of the derivative of its objective function.

To prove Theorem 1, we need several intermediate results, which are stated and proved through Lemmas 2–5. Our constructive proof of Theorem 1 begins by characterizing policies that maximize the first objective.

Lemma 2. Among all cyclic policies, a cyclic policy π maximizes $J_1(\pi)$ under the constraint (2), for any $T_{\text{obs}} > 0$, if and

[2] Harmonic mean is usually defined as $\lambda_{\text{hm}} = \left((1/n)\sum_{i=1}^n \lambda_i^{-1}\right)^{-1}$. Accordingly, we define the harmonic sum as $\sigma = n\,\lambda_{hm}$.

only if

$$t_i = \frac{\sigma T_{\text{obs}}}{\lambda_i} = \frac{T_{\text{obs}}}{\lambda_i \sum_{j=1}^n \frac{1}{\lambda_j}}. \tag{4}$$

Moreover, such a cyclic policy π satisfies

$$\mathbb{E}[N_1(\pi)] = \mathbb{E}[N_2(\pi)] = \cdots = \mathbb{E}[N_n(\pi)]. \tag{5}$$

Proof: By linearity of expectation, the value of J_1, as defined in (1), remains the same if we only look at a single policy cycle. We show that for an arbitrary $T_{\text{obs}} > 0$, choosing t_i's according to (4) yields the same optimal value for J_1. Fixing a policy π, after spending t_i time at station i, the robot collects $\mathbb{E}[N_i(\pi)] = \lambda_i t_i$ data points in expectation. This yields

$$\alpha_i(\pi) = \frac{\mathbb{E}[N_i(\pi)]}{\sum_{j=1}^n \mathbb{E}[N_j(\pi)]} = \frac{\lambda_i t_i}{\sum_{j=1}^n \lambda_j t_j}.$$

By the pigeonhole principle, $\min_i \alpha_i(\pi)$ is maximized if and only if (5) is satisfied, yielding $J_1 = 1/n$. When (5) holds, the constraint (2) is satisfied since it has a value of zero. Solving the set of equations

$$\begin{cases} \lambda_1 t_1 = \ldots = \lambda_n t_n \\ \sum_{i=1}^n t_i = T_{\text{obs}} \end{cases}$$

then yields (4). □

Remark: Lemma 2 implies that any cyclic policy that equalizes $\mathbb{E}[N_i(\pi)]$ across the stations optimizes the first objective J_1. This provides us with an infinite set of optimal policies for the first objective. Any policy satisfying (4) is optimal, independent of the value of the policy period T. △

Next, we show that, among the set of policies Π provided by Lemma 2, there exists a unique policy π^* that optimizes the second objective J_2. To achieve this, a method for evaluating $\mathbb{E}[T_i(\pi)]$ is required. It turns out that an analytical formula can be derived for computing $\mathbb{E}[T_i(\pi)]$.

Lemma 3. Let $\pi = (t_1, \ldots, t_n)$ be a cyclic policy and let $T = T_{\text{tr}} + \sum_{i=1}^n t_i$ be the period of the cyclic policy. Then

$$\mathbb{E}[T_i(\pi)] = \frac{2}{\lambda_i} + \frac{T - t_i - t_i e^{-\lambda_i t_i}}{1 - e^{-\lambda_i t_i}}. \tag{6}$$

Proof: To compute $\mathbb{E}[T_i(\pi)]$, without loss of generality, fix an observation window at station i and call it observation window 0, or o_0. We may further assume without loss of generality that o_0 contains the arrival of at least one event at station i. We look at all observation gaps on the right of o_0. The left side of o_0 may be safely ignored due to the memoryless property of Poisson processes. Any observation gap g_j contains the following parts, from left to right: 1) t_j^{left}, the overlap of g_j with the observation window on g_j's left end; 2) $T - t_i$, the first observation break (an observation break for station i is the time window between two consecutive visits to station i); 3) $0 \le m < \infty$ additional policy cycles (of length T each); and 4) t_j^{right}, the overlap of g_j with the observation window on g_j's right end. As an example, in Fig. 3, the start and end of the observation gap g_j are marked with the two dotted lines. The parts t_j^{left}, the first observation break $T - t_i$, and t_j^{right} are also marked. The gap g_j further contains two additional policy cycles, i.e., $m = 2$.

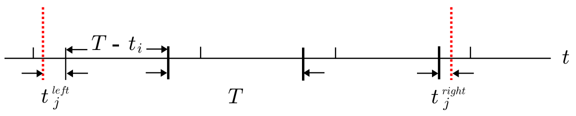


Fig. 3. Illustration of the components of an observation gap.

The computation of $\mathbb{E}[T_i(\pi)]$ may be carried out using a two-step process: 1) compute the probability p_m of a gap g_j containing m additional policy cycles; and 2) compute $\mathbb{E}[T_i(\pi)]$ as

$$\mathbb{E}[T_i(\pi)] = \sum_{m=0}^{\infty} p_m \mathbb{E}_m \tag{7}$$

in which $\mathbb{E}_m$ is the expected length of the gap g_j containing m additional policy cycles. Note that (7) holds as long as the expectations $\mathbb{E}[T_i(\pi)]$ and $\mathbb{E}_m$ have the same underlying distribution. We compute $\mathbb{E}_m$ with

$$\begin{aligned} \mathbb{E}_m &= \mathbb{E}[t_j^{\text{left}}] + \mathbb{E}[t_j^{\text{right}}] + T - t_i + mT \\ &= 2\mathbb{E}[t_j^{\text{left}}] + T - t_i + mT. \end{aligned} \tag{8}$$

A time reversed Poisson process is again a Poisson process with the same arrival rate. Due to this symmetry along the time, the second equality in (8) holds because $\mathbb{E}[t_j^{\text{left}}] = \mathbb{E}[t_j^{\text{right}}]$. To compute p_m, note that we never need to consider the left side of a gap g_j. This is true because as we look at an infinite sequence of consecutive gaps $g_1, \ldots, g_j, \ldots$. The left most observation window (which is o_0) overlapping with g_1 is fixed by assumption. Once the right most observation window overlapping with g_1 is set (with certain probability), this explicitly fixes the left most observation window overlapping with g_2 and recursively, the left most observation window overlapping g_j. Therefore, the probability of g_j spanning m additional policy cycles is

$$p_m = e^{-m\lambda_i t_i}(1 - e^{-\lambda_i t_i}). \tag{9}$$

The first term of (9), $e^{-m\lambda_i t_i}$, is the probability that g_j contains $0, 1, \ldots, m-1$ full policy cycles. The probability of no event happening in each additional cycle in the sequence is $e^{-\lambda_i t_i}$. They can be combined because the exponential distribution is memoryless. The term $(1 - e^{-\lambda_i t_i})$ is the probability that at least one event happens in the right most observation window overlapping g_j. Noting that the terms $2\mathbb{E}[t_j^{\text{left}}] + T - t_i$ appear in all $\mathbb{E}_m$'s, we can rewrite $\mathbb{E}[T_i(\pi)]$ as

$$2\mathbb{E}[t_j^{\text{left}}] + T - t_i + \sum_{m=1}^{\infty} mTe^{-m\lambda_i t_i}(1 - e^{-\lambda_i t_i}) \tag{10}$$

in which

$$\begin{aligned} &\sum_{m=0}^{\infty} mTe^{-m\lambda_i t_i}(1 - e^{-\lambda_i t_i}) \\ &\quad = T(1 - e^{-\lambda_i t_i}) \sum_{m=1}^{\infty} \sum_{k=m}^{\infty} e^{-k\lambda_i t_i} \\ &\quad = T(1 - e^{-\lambda_i t_i}) \sum_{m=1}^{\infty} \frac{e^{-m\lambda_i t_i}}{1 - e^{-\lambda_i t_i}} = \frac{Te^{-\lambda_i t_i}}{1 - e^{-\lambda_i t_i}}. \end{aligned} \tag{11}$$

The computation of $\mathbb{E}[t_j^{\text{left}}]$ is carried out as follows. By assumption, at least one event happens during the given observation window of length t_i. Let the number of events within this t_i time be N_e. The probability of having k events is $Pr(N_e = k) = (\lambda_i t_i)^k e^{-\lambda_i t_i}/k!$. Let $\tau_{1,k}$ be the arrival time of the first event among k events. For each $k \geq 1$, the distribution of the k events is uniform over $[0, t_i]$. We have for $0 \leq t \leq t_i$,

$$Pr(\tau_{1,k} > t) = \left(\frac{t_i - t}{t_i}\right)^k$$

from which we can obtain the probability density function for $\tau_{1,k}$ and subsequently $\mathbb{E}[\tau_{1,k}] = t_i/(k+1)$. Then

$$\begin{aligned}\mathbb{E}[t_j^{\text{left}}] &= \frac{\sum_{k=1}^{\infty} \mathbb{E}[\tau_{1,k}] Pr(N_e = k)}{1 - Pr(N_e = 0)} \\ &= \frac{\sum_{k=1}^{\infty} \frac{t_i}{k+1} (\lambda_i t_i)^k e^{-\lambda_i t_i}/k!}{1 - e^{-\lambda_i t_i}} \\ &= \frac{1}{1 - e^{-\lambda_i t_i}} \sum_{k=1}^{\infty} \frac{t_i (\lambda_i t_i)^k e^{-\lambda_i t_i}}{(k+1)!} \\ &= \frac{1}{\lambda_i (1 - e^{-\lambda_i t_i})} (1 - e^{-\lambda_i t_i} - \lambda_i t_i e^{-\lambda_i t_i}) \\ &= \frac{1}{\lambda_i} - \frac{t_i e^{-\lambda_i t_i}}{1 - e^{-\lambda_i t_i}}.\end{aligned} \tag{12}$$

Finally, plugging (11) and (12) into (10) yields (6). □

Remark: The technique from Lemma 3 is generic and can be used to compute expectations of other types of delays. For example, the current $\mathbb{E}[T_i(\pi)]$ treats delays with different values of m with equal importance. It may be the case that we want to further penalize for not observing any events over longer periods of time. One simple way to enable this is to give weights to delays with larger m values. This can be incorporated easily by updating $\mathbb{E}_m$ to

$$\mathbb{E}_m = \mathbb{E}[t_j^{\text{left}}] + \mathbb{E}[t_j^{\text{right}}] + T - t_i + m^2 T.$$

The remaining steps for computing this alternative expected delay stay unchanged. △

Even with $\mathbb{E}[T_i(\pi)]$ for each of the $1 \leq i \leq n$ stations, finding the optimal policy among Π that minimizes J_2 remains a nontrivial task. To obtain $\min_\pi \max_i \mathbb{E}[T_i(\pi)]$, we have to build the upper envelope over n such expected delays and then locate the minimum on that envelope. Fortunately, $\mathbb{E}[T_i(\pi)]$ has some additional properties that make this task more manageable. One such property is that $\mathbb{E}[T_i(\pi)]$ is quasi-convex in T, meaning that all sublevel sets of $\mathbb{E}[T_i(\pi)]$ are convex.

Lemma 4. The expected delay at a station $\mathbb{E}[T_i(\pi)]$, given by (6), is quasi-convex in T for fixed $\{\lambda_i\}$ and T_{tr}.

Proof: See Appendix A for the mostly technical proof.

Another important property of $\mathbb{E}[T_i(\pi)]$ is its monotonic dependence over λ_i, holding other parameters fixed.

Lemma 5. For fixed σ, policy period T, and policy π given by (4), $\mathbb{E}[T_i(\pi)]$ increases monotonically as λ_i increases.

Proof: Plugging $T_{\text{obs}} := T - T_{\text{tr}}$ and $\sigma := 1/(\sum_{i=1}^n (1/\lambda_i))$ into (6) and treating $\mathbb{E}[T_i(\pi)]$ as a function of λ_i (denote the function as $f_N(\lambda_i)$) with T, T_{tr}, and σ all fixed, we get

$$f_N(\lambda_i) = \frac{2}{\lambda_i} + \frac{T - \frac{\sigma T_{\text{obs}}}{\lambda_i}(1 + e^{-\sigma T_{\text{obs}}})}{1 - e^{-\sigma T_{\text{obs}}}} \tag{13}$$

the derivative of which is

$$f'_N(\lambda_i) = \frac{\sigma T_{\text{obs}} e^{-\sigma T_{\text{obs}}} + \sigma T_{\text{obs}} + 2e^{-\sigma T_{\text{obs}}} - 2}{\lambda_i^2 (1 - e^{-\sigma T_{\text{obs}}})} \tag{14}$$

which is strictly positive for all positive σT_{obs} and arbitrary positive λ_i, implying that $f_N(\lambda_i)$ increases monotonically with respect to λ_i. □

Proof of Theorem 1: Lemmas 2 and 3 show that the optimal policy period is given by

$$\begin{aligned}T^* &:= \underset{T > T_{\text{tr}}}{\arg\min} \max_i \mathbb{E}[T_i(\pi)] \\ &= \underset{T > T_{\text{tr}}}{\arg\min} \max_i \left[\frac{2}{\lambda_i} + \frac{T - t_i - t_i e^{-\lambda_i t_i}}{1 - e^{-\lambda_i t_i}}\right].\end{aligned}$$

By monotonicity of $\mathbb{E}[T_i(\pi)]$ with respect to λ_i (see Lemma 5), $\max_i \mathbb{E}[T_i(\pi)]$ is simply $\mathbb{E}[T_i(\pi)]$ for the station i with the largest λ_i. This reduces computing T^* to finding the minimum on a single function, which is a quasi-convex function by Lemma 4. □

B. Uniqueness of Optimal Solution

For a multiobjective optimization problem, results like Theorem 1 generally only give one optimal solution on the Pareto front with a potentially continuum of optimal solutions. However, the policy given by Theorem 1 is in fact the *unique* optimal solution, due to Theorem 6.

Theorem 6 (Uniqueness of Optimal Solution). The optimal policy π^* provided by Theorem 1 is the unique policy that solves Problem 2.

Proof: We assume that we work with a fixed problem instance and assume the optimal policy computed by Algorithm 1 has a period of T^*. Theorem 1 shows that $J_1 \leq 1/n$ and can always reach $1/n$. To show that no other policy other than π^* lies on the Pareto front, we need to show that no policy with fixed $J_1 < 1/n$ yields better value on J_2.

Assume instead that there is another Pareto optimal solution $\pi' = (t'_1, \ldots, t'_n) \neq \pi^*$ for the same problem instance with $J_1(\pi') = c < 1/n$. Let the period of π' be T' and let $T'_{\text{obs}} = T' - T_{\text{tr}}$. Let π'' be the cyclic policy also with cycle period T' such that $J_1(\pi'') = 1/n$. Note that π'' is unique and $\pi'' = \pi^*$ if $T' = T^*$. For π' to be on the Pareto front, because $J_1(\pi') = c < 1/n = J_1(\pi^*)$, one must have $J_2(\pi') < J_2(\pi^*)$; we show that on the contrary we always have $J_2(\pi') > J_2(\pi'') \geq J_2(\pi^*)$, in which the last inequality is clear. We are left to show $J_2(\pi') > J_2(\pi'')$.

Since $J_1(\pi') = \min_i \alpha_i(\pi')$, we may assume $J_1(\pi') = \alpha_1(\pi') = c$. This implies that $\lambda_1 t'_1 \leq \lambda_i t'_i$ for all $2 \leq i \leq n$. To satisfy constraint (2), which translates to $\min_{i \geq 2}(\lambda_i t'_i - \lambda_1 t'_1)$, we must have $\alpha_2(\pi') = \cdots = \alpha_n(\pi') > 1/n$ because having more $\alpha_i(\pi') < 1/n, i \geq 2$ will only increase $\min_{i \geq 2}(\lambda_i t'_i - \lambda_1 t'_1)$ (note that T'_{obs} is fixed). We then compute $\alpha_i(\pi') =$

$(1-c)/(n-1)$ for $i \geq 2$ and

$$\frac{\lambda_i t_i'}{\lambda_1 t_1'} = \frac{\alpha_i(\pi')}{\alpha_1(\pi')} = \frac{1-c}{c(n-1)}. \tag{15}$$

In addition, we have $\lambda_2 t_2' = \cdots = \lambda_n t_n'$ for $i \geq 2$. Solving this with the constraint $\sum_{i=2}^n t_i' = T_{\text{obs}}' - t_1'$ gives us for $i \geq 2$,

$$\lambda_i t_i' = \frac{T_{\text{obs}}' - t_1'}{\sum_{j=2}^n \frac{1}{\lambda_j}} = \frac{T_{\text{obs}}'}{\sum_{j=2}^n \frac{1}{\lambda_j}} - \frac{1}{\lambda_1 \sum_{j=2}^n \frac{1}{\lambda_j}} \lambda_1 t_1'. \tag{16}$$

Putting (15) and (16) together, we get

$$\lambda_1 t_1' = \frac{c(n-1)T_{\text{obs}}'}{(1-c)\sum_{i=2}^n \frac{1}{\lambda_i} + c(n-1)\frac{1}{\lambda_1}}. \tag{17}$$

For convenience, let $\Delta_{ij}(\pi) := |\mathbb{E}[N_i(\pi)] - \mathbb{E}[N_j(\pi)]|$. From (15)

$$\Delta_{1i}(\pi') = \lambda_i t_i' - \lambda_1 t_1' = \frac{1-cn}{c(n-1)} \lambda_1 t_1'. \tag{18}$$

Plugging (17) into (18) gives us that for some constant C,

$$(\Delta_{1i}(\pi'))^{-1} = C\left((1-c)\left(\frac{1}{\lambda_2} + \cdots + \frac{1}{\lambda_n}\right) + c(n-1)\frac{1}{\lambda_1}\right).$$

Because $(1-c) > (n-1)/n > c(n-1)$, to minimize $\Delta_{1i}(\pi')$ or maximize its inverse, λ_1 must equal $\lambda_{\max}$. This implies that $t_1' < t_1''$ (i.e., the time spent per cycle at a station with $\lambda_{\max}$ is less in π' than in π''). Therefore, because (6) monotonically decreases as $t_i > 0$ increases when the policy period T is fixed, $J_2(\pi') > J_2(\pi'')$. □

IV. Convergence and Robustness Properties of the Optimal Scheduling Policy

In this section, we prove two important "goodness" properties of the unique optimal cyclic policy for Problem 2, namely, the *convergence rate* of the policy toward its desired steady-state behavior and the *robustness* of the policy with respect to small perturbations of the input parameters.

A. Convergence Rate Toward Steady-State Behavior

Given an optimal policy π^* for Problem 2, we have proved that the total number of events observed at a station i, divided by the number of all observed events, converges to $\alpha_i(\pi^*)$ in expectation (i.e., given infinite amount of time). In practice, the execution of a monitoring policy must start at some point of time (instead of at $-\infty$) and only last a finite amount of time. Therefore, it is generally desirable that the sample averages converge quickly to their respective expected values. Here, we characterize the convergence rate of the fraction of observations with respect to the number of executed policy cycles. We do so by looking at the variance of these ratios around their expected values.

Theorem 7 (Convergence of the Fraction of Observations). Suppose the optimal policy for Problem 2 is executed for m cycles, that is, for mT^* amount of time in which T^* is the optimal policy's period. Then, the standard deviation of the fraction of total observations up until time mT^* acquired at any particular station is

$$\frac{1}{\sqrt{m\sigma T_{\text{obs}}^*}}$$

in which σ is the harmonic sum of the arrival rates.[3]

Proof: For convenience, assume that t is an integer multiple of cycle time T^*, i.e., $t = mT^*$, $m = 1, 2, \ldots$. For a fixed m, the Poisson process at station i is equivalent to a Poisson *distribution* with arrival rate

$$\lambda = k\lambda_i t_i = \frac{m(T^* - T_{\text{tr}})}{\sum_j \frac{1}{\lambda_j}}$$

which means that the variance of the number of data points observed is simply λ. The standard deviation of this Poisson distribution is then $\sqrt{\lambda}$, yielding a ratio of

$$\frac{\sqrt{\lambda}}{\lambda} = \sqrt{\frac{1}{\lambda}} = \sqrt{\frac{\sum_i \frac{1}{\lambda_i}}{m(T^* - T_{\text{tr}})}} = \sqrt{\frac{1}{m\sigma T_{\text{obs}}^*}}. \tag{19}$$

□

We note that the standard deviation given by (19) is station-independent. That is, the optimal schedule is such that the convergence occurs at the same rate across all stations. The theorem states that this standard deviation is inversely proportional to the square root of the number of cycles the schedule is executed, which is fairly reasonable.

B. Robustness of Optimal Policy

Another important issue related to solution soundness is its robustness. Under the particular context of this paper, it is desirable to ensure that the computed policy is robust with respect to small perturbations in the input parameters. Here, input parameters to our problem are $\tau_{i,j}$, $\{\lambda_i\}$, and an ordering of the stations. Since the ordering is a combinatorial object, it does not directly subject to perturbations. Therefore, we focus on the other two sets of parameters, which are continuous variables and can be readily perturbed.

1) Robustness with respect to perturbations in $\{\lambda_i\}$*:* We show that, when the optimal policy is deployed, the change in the expected delay $\mathbb{E}[T_i(\pi)]$ at a station i is bounded with respect to small changes in $\lambda_{\max}$. Furthermore, the rate of the change is fairly limited at nearly all stations.

Theorem 8 (Robustness w.r.t Arrival Rate). Let us denote the delay at station i under the optimal schedule as a function of λ_i by letting $f_N(\lambda_i) := \mathbb{E}[T_i(\pi)]$. Holding σ fixed and letting $x := \sigma T_{\text{obs}}$, then

$$\left(\frac{\Delta(f_N(\lambda_i))}{f_N(\lambda_i)}\right) / \left(\frac{\Delta\lambda_i}{\lambda_i}\right) < -\frac{2 - 2e^{-x} - x(1+e^{-x})}{2 - 2e^{-x} - x(1+e^{-x}) + \frac{\lambda_i}{\sigma}x}$$

the RHS of which is always upper bounded, and takes values in $(0,1)$ for all $x \in (0,\infty)$ and $\lambda_i \geq \lambda_{\min} := \min_j \lambda_j$.

Proof: Quantitatively, we want to show that $f_N(\lambda_i)$ [see (13)] does not change fast as λ_i varies. More formally, we seek

[3] In computing this measure, we look at the fraction of the total number of observations in one station versus the total number of observations acquired up until time mT^*.

to prove that $\Delta(f_N(\lambda_i))/f_N(\lambda_i)$ is small for small $\Delta\lambda_i/\lambda_i$. Through Taylor expansion

$$\frac{\Delta(f_N(\lambda_i))}{f_N(\lambda_i)} \approx \frac{f_N'(\lambda_i)\Delta\lambda_i}{f_N(\lambda_i)} = \frac{\lambda_i f_N'(\lambda_i)}{f_N(\lambda_i)}\frac{\Delta\lambda_i}{\lambda_i}.$$

By (13) and (14)

$$\frac{\lambda_i f_N'(\lambda_i)}{f_N(\lambda_i)} = -\frac{2-2e^{-\sigma T_{\rm obs}} - \sigma T_{\rm obs}(1+e^{-\sigma T_{\rm obs}})}{2-2e^{-\sigma T_{\rm obs}} - \sigma T_{\rm obs}(1+e^{-\sigma T_{\rm obs}}) + \lambda_i T}$$
$$\overset{x:=\sigma T_{\rm obs}}{=} -\frac{2-2e^{-x}-x(1+e^{-x})}{2-2e^{-x}-x(1+e^{-x})+\frac{x\lambda_i}{\sigma}+\lambda_i T_{\rm tr}}$$

in which $x > 0$ and $T_{\rm tr} > 0$. Because $\lambda_i/\sigma > 1$, $\lambda_i f_N'(\lambda_i)/f_N(\lambda_i)$ can be shown to be upper bounded by $1/(\lambda_i/\sigma - 1)$. In particular, for all $\lambda_i > \lambda_{\min}$, $\lambda_i/\sigma > 2$ holds, yielding

$$\frac{\lambda_i f_N'(\lambda_i)}{f_N(\lambda_i)} < -\frac{2-2e^{-x}-x(1+e^{-x})}{2-2e^{-x}-x(1+e^{-x})+2x}$$

which takes value in $(0,1)$ for all $x \in (0,\infty)$. If $\lambda_{\max} = \lambda_1 = \ldots = \lambda_n = \lambda_{\min}$, then we can similarly show that $\lambda_i f_N'(\lambda_i)/f_N(\lambda_i) < 1$. □

Since small relative changes to $\lambda_{\max}$ only induce relative changes of smaller or equal magnitude to the corresponding expected delay by Theorem 8, the optimal policy is robust with respect to perturbations to event arrival rates.

2) Robustness with respect to perturbations in $\{\tau_{i,j}\}$: Now suppose instead that elements of $\{\tau_{i,j}\}$ are perturbed. The only relevant change induced by these perturbations is a perturbation to $T_{\rm tr}$, the total travel time in a policy period. Perturbing $T_{\rm tr}$ causes a change in $T^*_{\rm obs}$, which is determined by the largest arrival rate $\lambda_{\max}$. We characterize the relative magnitude of this effect in the theorem below.

Theorem 9 (Robustness w.r.t. Travel Time). Let us denote the delay at the station with the maximum arrival rate as a function of $T_{\rm tr}$ by letting

$$f_{T_{\rm obs}}(T_{\rm tr}) := \frac{2}{\lambda_{\max}} + \frac{(T_{\rm obs}+T_{\rm tr})\lambda_{\max} - \sigma T_{\rm obs}(1+e^{-\sigma T_{\rm obs}})}{(1-e^{-\sigma T_{\rm obs}})\lambda_{\max}}. \quad (20)$$

Holding $\{\lambda_i\}$ and $T_{\rm obs}$ fixed, then

$$\left(\frac{\Delta(f_{T_{\rm obs}}(T_{\rm tr}))}{f_{T_{\rm obs}}(T_{\rm tr})}\right) / \left(\frac{\Delta T_{\rm tr}}{T_{\rm tr}}\right) \in (0,1).$$

PROOF. Following the proof of Theorem 8 and letting $x := \sigma T_{\rm obs}$, we compute

$$\left(\frac{\Delta(f_{T_{\rm obs}}(T_{\rm tr}))}{f_{T_{\rm obs}}(T_{\rm tr})}\right) / \left(\frac{\Delta T_{\rm tr}}{T_{\rm tr}}\right) \approx \frac{T_{\rm tr} f'_{T_{\rm obs}}(T_{\rm tr})}{f_{T_{\rm obs}}(T_{\rm tr})}$$
$$= \frac{\lambda_{\max} T_{\rm tr}(1-e^{-x})}{2(1-e^{-x}) - x(1+e^{-x}) + \lambda_{\max}T_{\rm obs} + \lambda_{\max}T_{\rm tr}}$$
$$< \frac{\lambda_{\max} T_{\rm tr}(1-e^{-x})}{2(1-e^{-x}) - x(1+e^{-x}) + x + \lambda_{\max}T_{\rm tr}}$$
$$= \frac{\lambda_{\max} T_{\rm tr}(1-e^{-x})}{2-2e^{-x} - xe^{-x} + \lambda_{\max}T_{\rm tr}}.$$

The inequality is due to $\lambda_{\max} > \sigma$. Because $2-2e^{-x}-xe^{-x} > 0$ for all $x = \sigma T_{\rm obs} > 0$ and $0 < 1-e^{-x} < 1$, we conclude that $T_{\rm tr} f'_{T_{\rm obs}}(T_{\rm tr})/f_{T_{\rm obs}}(T_{\rm tr}) \in (0,1)$. □

With Theorem 9, we conclude that the optimal policy is robust with respect to perturbing the travel times, $\{\tau_{i,j}\}$.

Remark: We note that the results from Sections III and IV continue to hold when the visiting order of the stations is not predetermined, due to the fact that travel times only matter as a whole (i.e., through $T_{\rm tr}$). The only nonessential difference is that there may be multiple optimal policies yielding the same J_1 and J_2 values, because there may be multiple TSP tours for a given problem instance. Such *degenerate* cases are, however, very rare.[4] Our analysis also implies that Problem 1 is NP-hard because it contains TSP as a subproblem. △

V. Computing the Optimal Scheduling Policy: Algorithm and Complexity Analysis

In this section, we first provide an algorithm for solving Problem 2 and characterize its performance. Then, building on this algorithm and robustness results from Section IV, we provide a polynomial-time approximation scheme (PTAS) for solving Problem 1.

A. *Algorithm for Computing Cyclic Policy With Predetermined Station Visiting Order*

The pseudocode for computing the unique cyclic policy π^* solving Problem 2 is given in Algorithm 1, as a direct consequence of Theorems 1 and 6. First, in Lines 1 and 2, the algorithm computes two useful statistics, namely, the maximum arrival rate (denoted by $\lambda_{\max}$) and σ. Then, in Line 3, the algorithm proceeds by solving an optimization problem in one variable, $T_{\rm obs}$. At this step, the algorithm computes the optimal total observation time denoted by $T^*_{\rm obs}$. Finally, the algorithm computes the optimal total cycle period $T^* = T^*_{\rm obs} + T_{\rm tr}$ in Line 4 and the optimal observation time for the individual stations in Lines 5 and 6.

We emphasize that the optimization problem in Line 3 of Algorithm 1 is a quasi-convex optimization problem in one variable by Theorem 1, which can be solved efficiently in multiple ways. For example, because the value of the function to be minimized can be computed analytically, we may apply the bisection method or the Newton–Raphson method to compute π^* very efficiently.

On the side of computational complexities of Algorithm 1, the following theorem is immediate. We measure the computational complexity of the algorithm by the number of steps executed by the algorithm. A single step is either a comparison, an addition, or a multiplication operation.

Theorem 10 (Complexity of Computing Optimal Schedule). The number of steps performed by Algorithm 1 is $O(n)$, in which n is the number of stations. Moreover, if $\lambda_{\max} = \max_i \lambda_i$ and the harmonic sum $\sigma = 1/\sum_{i=1}^{n}(1/\lambda_i)$ are known, the optimal cycle time can be computed in constant time.

[4]It is possible to show that such cases have zero measure with mild assumption on the station distribution.

Algorithm 1: COMPOPTORDERED

```
  Input : $(\lambda_1, \dots, \lambda_n)$: ordered arrival rates
          $\{\tau_{i,j}\}$: the travel times
  Output: $\pi^* = (t_1^*, t_2^*, \dots, t_n^*)$: the optimal policy

  %Compute relevant statistics
1 $\lambda_{\max} \leftarrow \max_{1\le i\le n} \lambda_i$ ;          %The maximum of $\lambda_i$'s
2 $\sigma \leftarrow \left(\sum_{i=1}^n \lambda_i^{-1}\right)^{-1}$ ;   %The harmonic sum of $\lambda_i$'s

  %Solve a quasi-convex optimization problem
3 $T_{\text{obs}}^* \leftarrow$
  $\arg\min_{T_{\text{obs}}>0} \left(\frac{2}{\lambda_{\max}} + \frac{(T_{\text{obs}}+T_{\text{tr}})\lambda_{\max} - \sigma T_{\text{obs}}(1+e^{-\sigma T_{\text{obs}}})}{(1-e^{-\sigma T_{\text{obs}}})\lambda_{\max}}\right)$;

  %Calculate the optimal policy
4 $T^* \leftarrow T_{\text{obs}}^* + T_{\text{tr}}$  %Calculate optimal cycle time
5 for $i \in \{1, 2, \dots, n\}$ do
      %Calculate optimal observation times.
6     $t_i^* \leftarrow \frac{\sigma}{\lambda_i} T_{\text{obs}}^*$
7 end
8 return $\pi^* = (t_1^*, t_2^*, \dots, t_n^*)$
```

Now, we consider *online* problem instances, in which new stations are added or other existing ones are removed, on the fly, from the list of stations to be serviced. The task is to construct the optimal schedule and maintain it as the list of stations changes.

First consider the problem with addition only. In that case, the online algorithm can be described as follows. At any given time, the algorithm maintains the maximum rate $\lambda_{\max}$ and the harmonic sum $\sigma = (\sum_{i=1}^n \lambda_i^{-1})^{-1}$. Let λ_{new} denote the event arrival rate for the new station. Then, the new statistics, denoted by $\lambda'_{\max}$ and σ', are computed as

$$\lambda'_{\max} \leftarrow \max\{\lambda_{\max}, \lambda_{\text{new}}\}$$
$$\sigma' \leftarrow \left(\sigma^{-1} + 1/\lambda_{\text{new}}\right)^{-1}.$$

Then, solve the quasi-convex optimization problem in Line 3 of Algorithm 1 to compute the optimal cycle time. Notice that these computations (the update and the solution of the quasi-convex optimization problem) can be executed in constant time. The running time of the algorithm that updates the optimal schedule time is independent of the number of stations. Second, consider the case when a new station may be added or an existing one can be removed. In this case, clearly the statistic σ can still be updated in constant time. However, maintaining the statistic $\lambda_{\max}$ is harder in the case of removals, since removing the station with rate $\lambda_{\max}$ requires looking through the remaining stations to find the station with the largest event arrival rate. This cannot be done in constant time. Yet, an ordered list of the stations can be maintained in logarithmic time. More precisely, the robot maintains an ordered list of stations such that the ordering is with respect to the event arrival rates λ_i. Adding a new station or removing a station from this can be performed in $\log(n)$ time, in which n is the number of stations. Once addition or removal is performed, the maximum event arrival rate, $\lambda_{\max}$, can be updated immediately. Hence, the overall update

Algorithm 2: COMPOPTUNORDERED

```
  Input : $\{\lambda_1, \dots, \lambda_n\}$: the arrival rates, unordered
          $\{\tau_{i,j}\}$: the travel times
  Output: $\pi^* = ((k_1^*, t_1^*), \dots, (k_n^*, t_n^*))$: the optimal policy

  %Compute an approximate TSP route
1 Using $\{\tau_{i,j}\}$, compute a $(1+\epsilon)$-optimal TSP solution
  over the distances, yielding $(k_1^*, \dots, k_n^*)$

  %Call the algorithm for Problem 2
2 $(\lambda_1, \dots, \lambda_n \leftarrow (\lambda_{k_1^*}, \dots, \lambda_{k_n^*})$;        %Reorder $\lambda_i$'s
3 $(t_1^*, \dots, t_n^*) \leftarrow$
  COMPOPTORDERED($(\lambda_1, \dots, \lambda_n), \{\tau_{i,j}\}$)

4 return $\pi^* = ((k_1^*, t_1^*), \dots, (k_n^*, t_n^*))$
```

algorithm requires logarithmic time in the number of stations. We summarize this as a corollary of our previous results.

Corollary 11 (Online Complexity). Consider the case in which new stations are added to the list of stations to be served, on the fly. When a new station is added to a list of stations to be observed, the optimal scheduling policy can be updated in constant time, independent of the number of existing stations. Consider the case when the stations are both added to and removed from a list of n stations to be served. Then, when a new station is added or removed, the optimal scheduling policy can be updated in $O(\log(n))$ time.

Remark: First, the space complexity, i.e., the amount of memory required to maintain the optimal cycle time, is constant when there are only additions. The space complexity is linear when there are removals as well. Second, clearly, solely updating the cycle time is not enough for implementing the optimal schedule; one must also update the time spent in each station. However, from a practical point of view, the time spent in each station can be updated as the robot travels to these destinations. This strategy should work well as long as the robot has computational power to evaluate Line 6 of Algorithm 1 (which requires two multiplications and one addition) during the time it spends at station $i-1$ and the time it travels to station i. In other words, the robot can compute the optimal cycle time T^* and start its monitoring of the stations. Right after T^* is computed, the robot can start the implementation of the plan. It computes t_1^* on the way to station 1 and when waiting at station 1, and so on. △

B. Computing Optimal Cyclic Policies Without a Predetermined Station Visiting Order

Fixing an arbitrary $\epsilon > 0$, our algorithm for computing a $(1+\epsilon)$-optimal solution for Problem 1, outlined in Algorithm 2, is a simple routine calling sequentially a TSP subroutine and then Algorithm 1. The flow of Algorithm 2 is straightforward to understand. The challenge is to show that a $(1+\epsilon)$-optimal TSP solution is all we need for computing a $(1+\epsilon)$-optimal solution to Problem 1. We now prove the correctness and the stated time complexity of Algorithm 2.

Theorem 12 (PTAS for Unordered Stations). Fixing a real number $\epsilon > 0$, a $(1+\epsilon)$-optimal policy can be computed for Problem 1 in time polynomial in n, the number of stations.

Proof. Suppose that the optimal total travel time is T^*_{tr} and Line 1 of Algorithm 2 computes a solution with total travel time $T'_{\text{tr}} = (1+\epsilon)T^*_{\text{tr}}$. The optimal policies computed using T^*_{tr} and T'_{tr} (and the associated visiting order), computed by Algorithm 1, have the same optimal J_1 value. Let the optimal values of J_2 for these two polices be J^*_2 and J'_2, respectively. We further let the optimal total observation times per cycle corresponding to T^*_{tr} and T'_{tr} be T^*_{obs} and T'_{obs}, respectively. For convenience, we reuse the definition of $f_{T_{\text{obs}}}(T_{\text{tr}})$ given by (20), which gives us

$$f_{T^*_{\text{obs}}}(T_{\text{tr}}) = \frac{2}{\lambda_{\max}} + \frac{(T^*_{\text{obs}} + T_{\text{tr}})\lambda_{\max} - \sigma T^*_{\text{obs}}(1 + e^{-\sigma T^*_{\text{obs}}})}{(1 - e^{-\sigma T^*_{\text{obs}}})\lambda_{\max}}.$$

We also know that $J^*_2 = f_{T^*_{\text{obs}}}(T^*_{\text{tr}})$ by definition. By Theorem 9, in particular that $T_{\text{tr}} f'_{T_{\text{obs}}}(T_{\text{tr}})/f_{T_{\text{obs}}}(T_{\text{tr}}) \in (0,1)$, we have

$$\begin{aligned} f_{T^*_{\text{obs}}}(T'_{\text{tr}}) &= f_{T^*_{\text{obs}}}((1+\epsilon)T^*_{\text{tr}}) \approx f_{T^*_{\text{obs}}}(T^*_{\text{tr}}) + \epsilon T^* f'_{T^*_{\text{obs}}}(T^*_{\text{tr}}) \\ &< f_{T^*_{\text{obs}}}(T^*_{\text{tr}}) + \epsilon f_{T^*_{\text{obs}}}(T^*_{\text{tr}}) = (1+\epsilon) f_{T^*_{\text{obs}}}(T^*_{\text{tr}}). \end{aligned}$$

On the other hand, it is straightforward to see that $f_{T_{\text{obs}}}(T_{\text{tr}})$ is monotonically increasing in T_{tr} since

$$f'_{T_{\text{obs}}}(T_{\text{tr}}) = \frac{1}{1 - e^{-\sigma T_{\text{obs}}}} > 0.$$

Therefore, for all $T_{\text{obs}} > 0$, we have

$$f_{T_{\text{obs}}}(T'_{\text{tr}}) \geq f_{T_{\text{obs}}}(T^*_{\text{tr}}) \geq f_{T^*_{\text{obs}}}(T^*_{\text{tr}}) = J^*_2$$

and

$$\begin{aligned} J'_2 &= \min_{T > T'_{\text{tr}}} f_{T_{\text{obs}}}(T'_{\text{tr}}) \leq f_{T^*_{\text{obs}}}(T'_{\text{tr}}) < (1+\epsilon) f_{T^*_{\text{obs}}}(T^*_{\text{tr}}) \\ &= (1+\epsilon)J^*_2. \end{aligned}$$

We conclude that $J^*_2 \leq J'_2 \leq (1+\epsilon)J^*_2$. That is, Algorithm 2 produces a $(1+\epsilon)$-optimal solution to Problem 1. To achieve the desired polynomial-time complexity, because Algorithm 1 takes linear time, we only need a PTAS for computing a $(1+\epsilon)$-optimal solution to the embedded TSP problem. Such a PTAS is provided in [25]. □

Remark: It may be the case that a PTAS does not yield practical polynomial-time algorithm. Fortunately, this does not present an issue for us as many fast TSP solvers already exist. For example, Lin–Kernighan Heuristics [26] can compute near optimal solutions for very larger TSP instances very quickly, even for more difficult TSP instances than Euclidean TSP, such as the asymmetric TSP. Typical instances with thousands of locations can be solved in a few minutes to an accuracy of 1% within the true optimal distance on a laptop. △

VI. Computational Experiments

Recall the UAV monitoring application illustrated in Fig. 1. The UAV is tasked with persistently monitoring six locations of interest and hovering over each location for certain periods of time to capture events occurring at these locations. The input consists of the arrival rates for events at each station (denoted by λ_i) and the time needed for traveling between the stations (denoted by $\tau_{i,j}$). Table II lists these parameters. The time unit is hours. Fig. 4 illustrates the stochastic nature of the event arrival times. Note that, in addition to the large range of average arrival rates at different stations (e.g., events arrive at station 3 five times more frequently than they do at station 1), the stochastic arrival times can vary greatly within the same station. The UAV must balance the amount of data collected at all stations despite the different arrival rates while not incurring large delays in event observations between consecutive visits to the same location.

TABLE II
Ground Truth (Event Arrival Rates and Travel Times) Used in Our Simulations

	Station					
	1	2	3	4	5	6
λ_i (1/hr)	0.5	1.3	2.5	1.2	1.6	0.9
$\tau_{i,i+1 \bmod 6}$ (hrs)	0.15	0.25	0.1	0.3	0.2	0.2

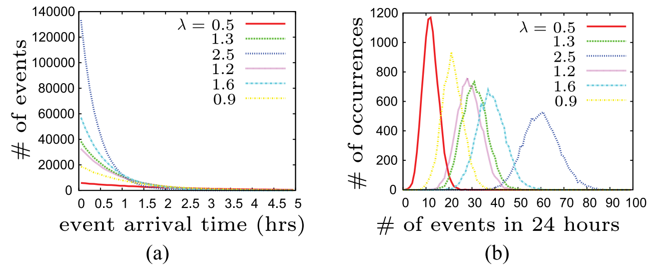


Fig. 4. (a) Histogram over the event arrival times since the last event arrival for the Poisson processes in our experiment over a time horizon of 10 000 days. The bucket size (on the x-axis) is 0.1 h. (b) Histogram over the number of events arriving in an 24-h window for the different Poisson processes over 10 000 runs.

The objective of the computational experiments is to confirm our theoretical findings given in Sections III and IV and offer insights into the structure induced by the optimization problem. Note that we do not lose generality by focusing on the case with known station cyclic order, which we do here. First, we verify the optimality of the computed schedule (see Theorem 1). We show that the schedule returned by the algorithm indeed minimizes the delay across all stations in a balanced way in a practical example scenario. Second, we focus on the convergence properties (see Theorem 7). We show that, in the same scenario, the fraction of observations at each station converges to zero at the rate given in Theorem 7 as the execution time increases. Third, we look at the robustness of the optimal policy (see Theorem 8). We show that, in a variety of selected scenarios, the optimal policy is also robust with respect to the changes in event arrival statistics. We omit the simulation study on Theorem 9, which yields robustness results very similar to that of Theorem 8.

We mention that the source code for our simulation software was developed using the Java programming language, and the simulation software itself was executed on a computer with a 1.3-GHz Intel Core i5 CPU and 4-GB memory. Mathematica 9 was used for computing the optimal policy using the gradient descent optimization procedure. As suggested by Theorem 10, on this computational hardware, the computation of the optimal policy is almost instantaneous, on the order of a few milliseconds.

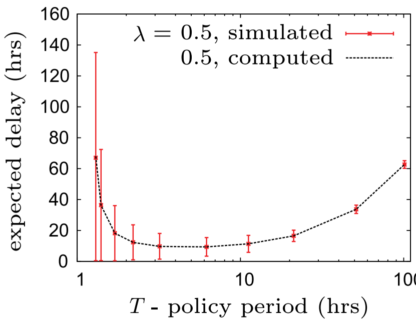


Fig. 5. Simulated versus computed values for $\mathbb{E}[T_i(\pi)]$. We observe that the mean of the simulated runs agrees with the value computed directly from (6) for all choices of T's, whereas the variance grows larger as $T \to T_{\text{tr}}$.

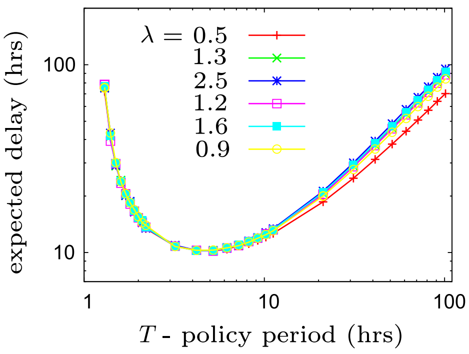


Fig. 6. Computed $\mathbb{E}[T_i(\pi)]$ for $\lambda_1, \ldots, \lambda_6$ and $T \in [1.3, 101.2]$.

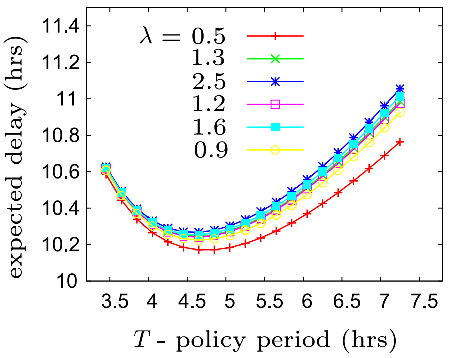


Fig. 7. Computed $\mathbb{E}[T_i(\pi)]$ for $\lambda_1, \ldots, \lambda_6$ and $T \in [6.2, 10.2]$ with $\Delta T = 0.025$ increments.

A. Computing the Optimal policy

In this section, we focus on the optimality of the proposed schedule. First, we show in simulations that our analysis correctly predicts the expected delay. Second, we compare the optimal schedule with an intuitive, but suboptimal policy.

Below, we empirically check the correctness of Theorem 1 through simulations. Our first computational experiment validates (6) by performing both simulation and direct computation side by side and comparing the results, for the aforementioned case. In simulation, for each fixed $T \in \{1.3, 1.4, 1.7, 2.2, 3.2, 6.2, 11.2, 21.2, 51.2, 101.2\}$, we simulated the Poisson process for enough number of periods (roughly 2×10^5 in the worst case) to gather at least 2000 delays by simulating the policy. This gave us 2000 samples of the random variable $T_i(\pi)$ from which we computed the mean and standard deviation. Direct computation based on (6) were also carried out. To avoid cluttering the presentation, only $\lambda = 0.5$ was used (plots for other arrival rates are similar).

The result from this simulation study is presented in Fig. 5, in comparison with the optimal policy that is directly computed using the gradient descent procedure. Notice that the expected delay in the simulation matches exactly that of the computed policy for all choices of T's. We also observe from the simulation study that the delay variance increases as T approaches T_{tr}. This should be intuitively clear, since, as $T - T_{\text{tr}} \to 0^+$, the length of each observation window decreases when compared with T_{tr}; in fact, the ratio of the two approaches zero, causing unbounded increase in the variance.

After empirically verifying that (6) is accurate, we shift our attention to the quasi-convexity of the delay $\mathbb{E}[T_i(\pi)]$ and its monotonicity in λ_i. We compute $\mathbb{E}[T_i(\pi)]$ for all six λ_i's and plot the result at two different scales in Figs. 6 and 7. Fig. 6 shows that $\mathbb{E}[T_i(\pi)]$ is quasi-convex (in this case, convex) for all λ_i's. Fig. 7, a zoomed-in version of Fig. 6, further reveals that $\mathbb{E}[T_i(\pi)]$ depends on λ_i monotonically for fixed policy period T, confirming the claim of Lemma 5.

To compute the optimal cyclic patrolling policy's parameters, by Lemma 5, we only need to look at $\mathbb{E}[T_i(\pi)]$ for $\lambda_i = 2.5$. The period T that minimizes (6) for $\lambda_i = 2.5$ can be easily computed using standard gradient descent methods. Our computation yields $T^* = 4.59$. The corresponding policy is then defined by $\pi = (1.18, 0.45, 0.24, 0.49, 0.37, 0.67)$.

TABLE III
EXPECTED DELAY FOR THREE POLICIES FOR THE SAME ENVIRONMENTS IN THE POLICIES THAT OPTIMIZE J_1

No.	T	π	$\mathbb{E}[T_1(\pi)]$
1	1.2502	$(1, 0.25)$	1.814
2	2.5002	$(2, 0.5)$	2.265
3	3.7502	$(3, 0.75)$	2.632

Remark: We note that $\mathbb{E}[T_i(\pi)]$ is not always *convex*, contrary to what may be suggested by computational experiments (e.g., Fig. 6). To see that $\mathbb{E}[T_i(\pi)]$ is quasi-convex, pick $n = 2$ as the number of stations with $\lambda_1 = 1$, $\lambda_2 = 4$, and $t_{12} = t_{21} = 0.0001$. For $T = 1.2502, 2.5002$, and 3.7502, the optimal policies balancing the observed data and the corresponding $\mathbb{E}[T_1(\pi)]$ are given in Table III, form which one can easily verify that the point $(2.5002, 2.265)$ lies above the line connecting points $(1.2502, 1.814)$ and $(3.7502, 2.632)$, implying that $\mathbb{E}[T_1(\pi)]$ is nonconvex on the interval $[1.2502, 3.7502]$. △

B. Performance on Non-Poisson Distributed Data

As it is often the case that stochastic event arrivals diverge from Poisson process, we are curious how our computed policy would perform on more realistic data. Because a large number of data points are needed to compute entities like average delay, instead of using real world data, we generated data to simulate

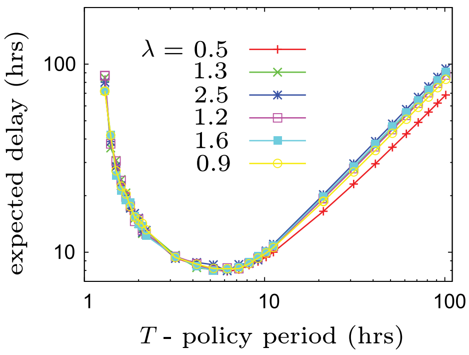


Fig. 8. Simulated average delay using data generated over a variation of the uniform distribution.

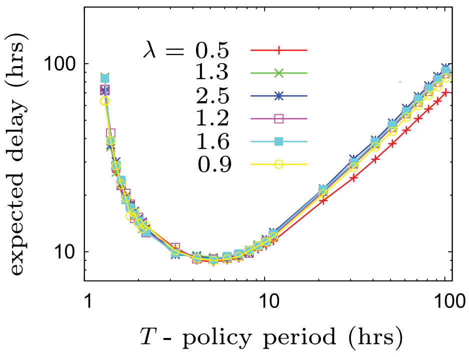


Fig. 9. Simulated average delay using data generated over a variation of the uniform distribution with bursts of arrivals.

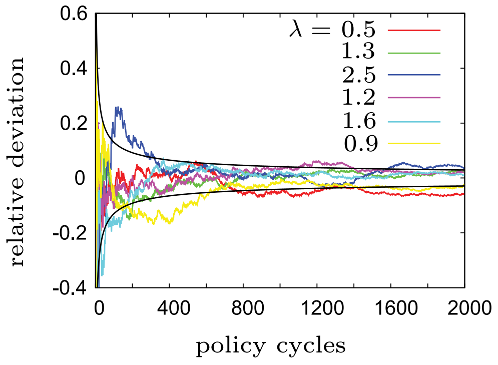


Fig. 10. Relative deviations of the data-collecting process at the six stations with different λ_i's over 2000 policy cycles. The two black lines are computed with Theorem 7.

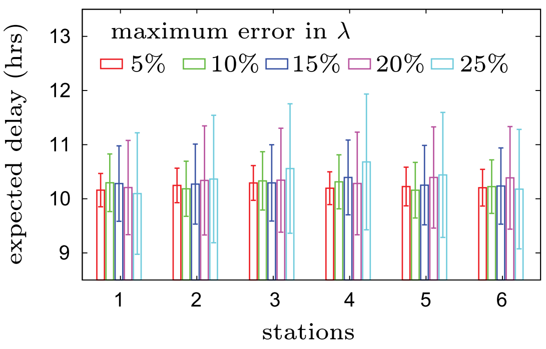


Fig. 11. Simulated delays ($\mathbb{E}[T_i(\pi)]$) when running the optimal policy π in environments with uncertainties in λ_i's.

processes like bus arrivals in the following manner. Using $\{\lambda_i\}$ from our main example, for each station i, we partition the time line into segments of length $1/\lambda_i$. In each segment, a uniformly random point is selected as the arrival time of an event at station i. Clearly, over this dataset, the value of J_1 remains the same. When we use the policies that maximize J_1 to simulate $\mathbb{E}[T_i(\pi)]$ over this dataset, we obtain results that are subsequently plotted in Fig. 8.

We observe that the average delay is actually shorter in this case, implying a better optimal value for J_2. We also note that the general structure of $\mathbb{E}[T_i(\pi)]$ appears to remain the same, i.e., largely convex. Then, to add *burst* behavior that often occurs in practice, in simulating events at station i, with 5% probability, we pick a random integer k between 1 and 9. Otherwise, for the other 95%, we set $k = 1$. We skip $k - 1$ segments of length $1/\lambda_i$ each and pack k events in the next segment of length $1/\lambda_i$. Note that in terms of buses, this data-generating process means that more than 20% of buses come in short bursts. The computed average delay is given in Fig. 9, which yields a larger optimal J_2 that remains smaller than that over Poisson processes.

C. Convergence of the Optimal Schedule

We have shown that the variance of the fraction of observations at each station converges to zero at a particular rate as the number of cycles increases (see Theorem 7). As noted there, the optimal policy is such that the same convergence rate was observed at each station. In other words, the optimal policy not only balances the fraction of observations at each station, but also balances the convergence rates.

Fig. 10 depicts this phenomenon for a single execution of the optimal schedule for 2000 cycles. It is seen that the standard deviation converges to zero roughly with the rate computed in Theorem 7. Moreover, the standard deviations are roughly the same across all stations.

D. Robustness of Optimal Policies

With Theorem 8, one can expect the optimal policy to be robust in the sense that small estimation errors in the arrival rates should not greatly affect the performance of an optimal policy. We now use simulation to illustrate the robustness of an optimal policy. In our simulation based on the same λ_i's (note that in this case, $\lambda_i/\sigma > 2$ holds for all i's), we assume that the actual event arrival rate may vary up to 25% (assuming randomly distributed errors in λ_i's). For each error threshold from 5% to 25%, 100 simulations were performed using environments based on these random (fixed) λ_i's, over which the same optimal policy was run for 10 000 policy cycles. The results were plotted in Figs. 11 and 12.

Fig. 11 shows that using the same policy, one can expect relatively stable performance despite fairly large error in the estimated λ_i's. For example, with up to 25% maximum error, $\mathbb{E}[T_i(\pi)]$ only varies about 10% across all stations at one standard deviation (i.e., it is not very sensitive to the magnitude

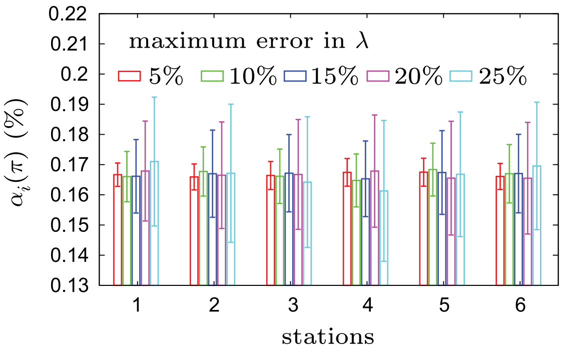


Fig. 12. Simulated $\alpha_i(\pi)$ when running the optimal policy π in environments with uncertainties in λ_i's.

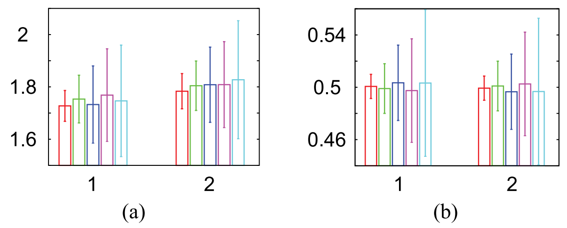


Fig. 13. Simulated $\mathbb{E}[T_i(\pi)]$ and $\alpha_i(\pi)$ when running the optimal policy π in environments with uncertainties in λ_i's. Here, the policy is generated based on $\lambda_1 = 1$ and $\lambda_2 = 100$. The two graphs correspond to Figs. 11 and 12, respectively. We omitted axes labels and legends that have identical meanings with those in Figs. 11 and 12.

of λ_i's). Similar behavior can be observed for $\alpha_i(\pi)$: Up to 25% error in λ_i's yields a standard deviation of about 25% in $\alpha_i(\pi)$ across all stations.

Although not directly implied by Theorem 8, an optimal policy also appears to be stable with respect to widely varying stochastic arrival rates. Taking an extreme example having two stations with $\lambda_1 = 1$, $\lambda_2 = 100$ (here, $\lambda_2/\sigma = 1.01 < 2$), and $\tau_{12} = \tau_{21} = 0.1$, we performed the same experiments on stability, the results of which are captured in Fig. 13. The deviations are similar to what we observed in Figs. 11 and 12. The optimal policy here is $\pi = (0.5702, 0.0057)$.

VII. Conclusion

We have introduced a novel persistent monitoring and data collection problem in which transient events at multiple stations arrive following stochastic processes. We studied the performance of cyclic policies on two objectives: 1) maximizing the minimum fraction of expected events to be collected at each station so that no station receives insufficient or excessive monitoring effort; and 2) minimizing the maximum delay in observing two consecutive events generated by the same process between policy cycles. We focused on an important case in which the locations to be visited form a closed chain. We showed that such a problem admits a (often unique) cyclic policy that optimizes both objectives. We also showed that the second, more complex objective function is quasi-convex, allowing efficient computation of the optimal policy with standard gradient descent methods when the cyclic ordering of the stations is fixed.

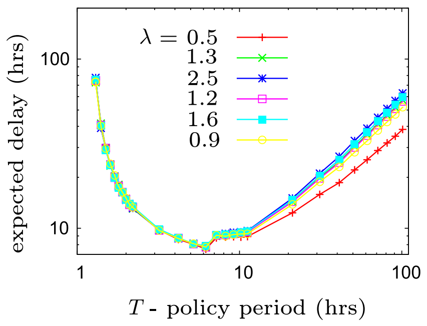


Fig. 14. Simulated average delay with feedback such that the robot will leave a station once the expected number of events per cycle is collected. The corresponding expected delays without feedback are plotted in Fig. 7.

Moreover, our study on important properties of the optimal solution, including the convergence rate and the solution robustness, further offered insights that lead to an polynomial-time approximation algorithm for the more general problem in which a cyclic order is unknown *a priori*.

Our study also raises many interesting and well-formulated open problems; we discuss two here. First, in our formulation, the robot is not required to process the data it collects while waiting at the stations. Whereas this assumption applies to many scenarios, it is perhaps equally natural to assume the opposite and let the robot know when it observes an event. This then gives rise to feedback or adaptive policies. For example, one way to design such a policy is to let the robot move away from a station once it knows enough number of events have been collected at the station. Intuitively, such feedback policies should do better due to the memoryless property of Poisson process. Preliminary simulation result confirms our hypothesis (see Fig. 14). Interestingly, such feedback policies seem to induce discrete jumps in the average delay, which we look forward to understanding in future research. Another interesting and related angle is to allow sensors to have nontrivial footprint. That is, the mobile sensor is able to cover multiple stations simultaneously.

Second, we had initially conjectured that TSP-based cyclic policies might be the best policies for the proposed multiobjective optimization problem without requiring the cyclic policy assumption. This turns out not to be the case; Appendix B provides a counterexample. In the counterexample, a *periodic policy*[5] is proven to be better than the TSP-based cyclic policy. This observation prompts at least two intriguing questions: 1) How we may find the optimal periodic policy for the proposed multiobjective optimization problem? 2) Are periodic policies the best policies without feedback?

Appendix A
Technical Proofs

Proof of Lemma 4: For notational convenience, define $\gamma_i := \sigma/\lambda_i$. Note that we implicitly use the fact that all functions used in the proof are continuous. Substituting $T_{\rm obs} = T - T_{\rm tr}$ and

[5]In contrast with a cyclic policy, which allows a single visit to each station during a policy period, a periodic policy allows multiple visits to the same station during a single policy period.

$t_i = \gamma_i T_{\text{obs}}$ into the RHS of (6) yields

$$\begin{aligned}\mathbb{E}[T_i(\pi)] &= \frac{2}{\lambda_i} + \frac{T - t_i - (T - t_i)e^{-\lambda_i t_i} + (T - 2t_i)e^{-\lambda_i t_i}}{1 - e^{-\lambda_i t_i}} \\ &= \frac{2}{\lambda_i} + \frac{T - t_i - t_i e^{-\lambda_i t_i}}{1 - e^{-\lambda_i t_i}} \\ &= \frac{2}{\lambda_i} + \frac{T_{\text{obs}} + T_{\text{tr}} - \gamma_i T_{\text{obs}} - \gamma_i T_{\text{obs}} e^{-\lambda_i \gamma_i T_{\text{obs}}}}{1 - e^{-\lambda_i \gamma_i T_{\text{obs}}}}.\end{aligned}$$

By scaling the unit of time, we may assume that $\lambda_i = 1$. Using this and letting $x := \gamma_i T_{\text{obs}}$ give us

$$\begin{aligned}\mathbb{E}[T_i(\pi)] &= 2 + \frac{T_{\text{tr}} + (\frac{1}{\gamma_i} - 1)x - xe^{-x}}{1 - e^{-x}} \\ &= 2 + \frac{T_{\text{tr}} + (\frac{1}{\gamma_i} - 2)x}{1 - e^{-x}} + x\end{aligned}$$

in which $T_{\text{tr}} > 0$ and $\gamma_i \in (0, 1)$. For convenience, we let $\alpha := T_{\text{tr}}$ and $\beta := 1/\gamma_i - 2$. Showing that $\mathbb{E}[T_i(\pi)]$ is quasi-convex is equivalent to showing that

$$f(x) := \frac{\alpha + \beta x}{1 - e^{-x}} + x$$

is quasi-convex for $x > 0$,[6] $\alpha > 0$, and $\beta > -1$, the second derivative of which is

$$f''(x) = \frac{e^x(\alpha(e^x + 1) + \beta(e^x(x - 2) + x + 2))}{(-1 + e^x)^3}.$$

Since $e^x(x - 2) + x + 2$ is strictly positive,[7] $f''(x) > 0$ for $\beta \geq 0$. Therefore, $f(x)$ is convex for $\beta \geq 0$. We are left to show that $f(x)$ is quasi-convex for $\beta \in (-1, 0)$. We proceed by first establishing some properties of the function

$$g(x) = \alpha(e^x + 1) + \beta(e^x(x - 2) + x + 2)$$

for $\alpha > 0$, and $\beta \in (-1, 0)$. We have $g(x) \in C^\infty$ for $x \geq 0$, $g(0) = 2\alpha > 0$, $\lim_{x\to\infty} g(x) = -\infty$,

$$g'(x) = (\alpha + \beta x - \beta)e^x + \beta$$

and

$$g''(x) = (\alpha + \beta x)e^x.$$

Because $(\alpha + \beta x)$ is linear, monotonically decreasing and crosses zero at most once, and e^x is positive and strictly increasing, $g''(x)$ has at most a single local extrema (a maxima) before it crosses zero. Therefore, $g'(x)$ has at most two zeros and must first increase monotonically and then decrease monotonically, implying that $g(x)$ has at most three zeros. Since $g(0) > 0$ and $\lim_{x\to\infty} g(x) = -\infty < 0$, $g(x)$ has either one or three (but not two) zeros. For $g(x)$ to have three zeros, $g'(x)$ must have two zeros. Since $\lim_{x\to\infty} g'(x) = -\infty$ (because $\beta x e^x$ eventually dominates and $\beta < 0$), we must have $g'(0) < 0$. This is not possible because $g'(0) = \alpha > 0$. Therefore, $g'(x)$ can cross zero and change sign at most once,[8] implying that $g(x)$ has a single zero. That is, $g(x)$ is positive for small x and then remains negative after crossing zero. Because

$$f''(x) = \frac{e^x g(x)}{(-1 + e^x)^3}$$

and $e^x/(-1 + e^x)^3$ is strictly positive, $f''(x)$ behaves similarly as $g(x)$ (*i.e.*, $f''(0) > 0$, crosses zero only once as x increases, and stays negative after that). This implies that for every fixed $\alpha > 0$ and $\beta \in (-1, 0)$, there exists $x_0 > 0$ such that $f(x)$ is convex on $x \in (0, x_0)$ and concave on $x \in (x_0, \infty)$. Now because $f(x) \to \infty$ for both $x \to 0^+$ and $x \to \infty$, and $f(1) < \infty$, $f(x)$ must have a single local minima (and therefore, a single global minima on $\mathbb{R}^+$). To see that this is the case, as $f(x)$ turns from convex to concave at $x = x_0$, we must have $f'(x_0) \geq 0$ because otherwise $f'(x) < 0$ for $x > x_0$ due to $f(x)$'s concavity. We then have $\lim_{x\to\infty} f(x) < \infty$, a contradiction. Thus, $f(x)$ has a single minimum on $x \in (0, x_0)$. Finally, to see that $f(x)$ is quasi-convex, we note that $\lim_{x\to\infty} f'(x) = 1 + \beta > 0$, implying that $f'(x) > 0$ on all $x \in (x_0, \infty)$. We then have that $f(x)$ is monotonically increasing on $x \in (x_0, \infty)$. From here, the quasi-convexity of $f(x)$ can be easily shown following definitions. □

Appendix B
Nonoptimal TSP Cyclic Policies

In this appendix, we provide an example problem for which the optimal TSP cyclic policy is not the optimal policy for maximizing J_1 and minimizing J_2. We build the problem in two steps. Our initial problem, which is to be updated in a little while, has three stations with input parameters

$$\lambda_1 = \lambda_3 = 1, \lambda_2 = 2, \tau_{1,2} = \tau_{2,3} = 0.1, \tau_{3,1} = 0.2.$$

Using Algorithm 1, we compute the optimal cyclic policy as $\pi_1 = (t_1 = 0.53, t_2 = 0.27, t_3 = 0.53)$, which contains a TSP tour of the stations. We may further compute $\mathbb{E}[T_1(\pi_1)] = \mathbb{E}[T_3(\pi_1)] = 4.15$ and $\mathbb{E}[T_2(\pi_1)] = 4.17$, which implies that $J_2(\pi_1) = 4.17$. Then, we modify π_1 to get another policy π_2, which is a periodic policy, by changing the visiting order of the stations to $1, 2, 3, 2, 1, \ldots$, i.e., station 2 is visited twice as frequently, and letting the robot stay at station 2 for $t_2/2$ time for each visit. Employing the proof technique of Lemma 3, we compute that $\mathbb{E}[T_2(\pi_2)] = 3.26$, whereas $\mathbb{E}[T_i(\pi_2)] = \mathbb{E}[T_i(\pi_1)] = 4.15$ for $i \in \{1, 3\}$. Thus $J_2(\pi_2) = 4.15$. Because $J_1(\pi_1) = J_1(\pi_2) = 1/3$, π_2 is a better periodic policy than π_1.

We now construct the final example problem with three stations and update the parameters to

$$\lambda_1 = \lambda_3 = 1, \lambda_2 = 2, \tau_{1,2} = \tau_{2,3} = 0.1 + \epsilon, \tau_{3,1} = 0.2$$

in which $\epsilon > 0$ is a small perturbation. That is, we make the two paths between stations i and $i + 1$, $i \in \{1, 2\}$, a little longer. As long as $\epsilon > 0$, we have that a robot executing π_2 will travel a

[6]In the rest of the proof, unless explicitly stated otherwise, the domain of x is assumed to be $(0, \infty)$.

[7]To see this, let $h(x) = e^x(x - 2) + x + 2$; then $h(0) = 0$, $h'(0) = 0$, and $h''(x) = xe^x > 0$ for all $x > 0$. Therefore, $h'(x) > 0$ and $h(x) > 0$ for all $x > 0$.

[8]Alternatively, solving $g'(x) = 0$ in Mathematica yields at most a single zero in $(0, \infty)$ at $x = \dfrac{\beta W(-e^{\frac{\alpha}{\beta} - 1}) - \alpha + \beta}{\beta}$, in which $W(\cdot)$ is the (principal) *Lambert W-function*.

strictly longer distance during each policy period than a robot executing π_1, the TSP-based cyclic policy. However, by continuity, for a small enough ϵ, π_2 will remain a better policy than π_1 for optimizing J_1 and J_2.

References

[1] N. Michael, E. Stump, and K. Mohta, "Persistent surveillance with a team of MAVs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2011, pp. 2708–2714.

[2] 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," *J. Field Robot.*, vol. 28, no. 5, pp. 714–741, Sep./Oct. 2011.

[3] S. Alamdari, E. Fata, and S. L. Smith, "Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations," *Int. J. Robot. Res.*, vol. 33, no. 1, pp. 138–154, 2014.

[4] E. Arvelo, E. Kim, and N. C. Martins. (2012, Sep.). Memoryless control design for persistent surveillance under safety constraints. *ArXiv*. [Online]. Available: http://arxiv.org/abs/1209.5805

[5] C. G. Cassandras, X. Lin, and X. Ding, "An optimal control approach to the multi-agent persistent monitoring problem," *IEEE Trans. Autom. Control*, vol. 58, no. 4, pp. 947–961, Apr. 2013.

[6] A. Girard, A. Howell, and J. Hedrick, "Border patrol and surveillance missions using multiple unmanned air vehicles," in *Proc. 43rd IEEE Conf. Decision Control*, 2004, pp. 620–625.

[7] B. Grocholsky, J. Keller, V. Kumar, and G. Pappas, "Cooperative air and ground surveillance," *IEEE Robot. Autom. Mag.*, vol. 13, no. 3, pp. 16–25, Sep. 2006.

[8] X. Lan and M. Schwager, "Planning periodic persistent monitoring trajectories for sensing robots in Gaussian random fields," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 2407–2412.

[9] N. Nigam and I. Kroo, "Persistent surveillance using multiple unmanned air vehicles," in *Proc. IEEE Aerospace Conf.*, 2008, pp. 1–14.

[10] S. L. Smith, M. Schwager, and D. Rus, "Persistent robotic tasks: Monitoring and sweeping in changing environments," *IEEE Trans. Robot.*, vol. 28, no. 2, pp. 410–426, Apr. 2012.

[11] D. E. Soltero, M. Schwager, and D. Rus, "Generating informative paths for persistent sensing in unknown environments," in *Proc. Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 2172–2179.

[12] D. J. Bertsimas and G. J. van Ryzin, "Stochastic and dynamic vehicle routing with general interarrival and service time distributions," *Adv. Appl. Probability*, vol. 25, pp. 947–978, 1993.

[13] M. Pavone, E. Frazzoli, and F. Bullo, "Adaptive and distributed algorithms for vehicle routing in a stochastic and dynamic environment," *IEEE Trans. Automat. Control*, vol. 56, no. 6, pp. 1259–1274, Jun. 2011.

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

[15] H. Choset, "Coverage for robotics—A survey of recent results," *Ann. Math. Artif. Intell.*, vol. 31, pp. 113–126, 2001.

[16] Y. Gabriely and E. Rimon, "Competitive on-line coverage of grid environments by a mobile robot," *Comput. Geometry*, vol. 24, no. 3, pp. 197–224, 2003.

[17] W.-P. Chin and S. Ntafos, "Optimum watchman routes," *Inf. Process. Lett.*, vol. 28, pp. 39–44, 1988.

[18] P. Hokayem, D. Stipanovic, and M. Spong, "On persistent coverage control," in *Proc. 46th IEEE Conf. Decision Control*, 2008, pp. 6130–6135.

[19] S. Ntafos, "Watchman routes under limited visibility," *Comput. Geometry*, vol. 1, pp. 149–170, 1991.

[20] J. A. Fuemmeler and V. V. Veeravalli, "Smart sleeping policies for energy-efficient tracking in sensor networks," *Netw. Sensing Inform. Control*, 2008.

[21] Y. He and E. K. P. Chong, "Sensor scheduling for target tracking in sensor networks," in *Proc. 43rd IEEE Conf. Decision Control*, 2004, pp. 743–748.

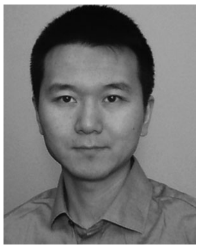
[22] A. O. H. III, C. M. Kreucher, and D. Blatt, "Information theoretic approaches to sensor management," in, *Foundations and Applications of Sensor Management*. New York, NY, USA: Springer, 2008.

[23] J. L. Ny, M. A. Dahleh, E. Feron, and E. Frazzoli, "Continuous path planning for a data harvesting mobile server," in *Proc. 47th IEEE Conf. Decision Control*, 2008, pp. 1489–1494.

[24] J. Yu, S. Karaman, and D. Rus, "Persistent monitoring of events with stochastic arrivals at multiple stations," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2014, pp. 5758–5765.

[25] S. Arora, "Polynomial-time approximation schemes for Euclidean TSP and other geometric problems," *J. ACM*, vol. 45, no. 5, pp. 753–782, 1998.

[26] K. Helsgaun, "An effective implementation of the Lin-Kernighan traveling salesman heuristic," *Eur. J. Oper. Res.*, vol. 126, no. 1, pp. 106–130, 2000.



Jingjin Yu (S'11–M'13) received the B.S. degree in materials science from University of Science and Technology of China, Hefei, China, in 1998; the M.S. degrees in chemistry from University of Chicago, Chicago, IL, USA, in 2000, in mathematics from University of Illinois at Chicago, Chicago, in 2001, and in computer science from University of Illinois at Urbana-Champaign, Champaign, IL, in 2010; and the Ph.D. degree in electrical engineering from University of Illinois at Urbana-Champaign in 2013.

He is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA.

Sertac Karaman (M'14) received the B.S. degrees in mechanical engineering and in computer engineering from Istanbul Technical University, Istanbul, Turkey, in 2007; the S.M. degree in mechanical engineering from Massachusetts Institute of Technology (MIT), Cambridge, MA, USA, in 2009; and the Ph.D. degree in electrical engineering and computer science also from MIT in 2012.

He has been the Charles Stark Draper Assistant Professor of aeronautics and astronautics with MIT since Fall 2012. His research interests include the broad areas of robotics and control theory. In particular, he studies the applications of probability theory, stochastic processes, stochastic geometry, formal methods, and optimization for the design and analysis of high-performance cyber–physical systems.

Daniela Rus (F'15) received the Ph.D. degree in computer science from Cornell University, Ithaca, NY, USA.

She is the Andrew (1956) and Erna Viterbi Professor of electrical engineering and computer science and Director of the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. Prior to joining MIT, she was a Professor with the Computer Science Department, Dartmouth College. Her research interests include robotics, mobile computing, and big data.

Dr. Rus is a Class of 2002 MacArthur Fellow, a Fellow of ACM and AAAI, and a Member of the NAE.