2021

Persistent monitoring of targets with uncertain states

BOSTON UNIVERSITY

COLLEGE OF ENGINEERING

Dissertation

PERSISTENT MONITORING OF TARGETS WITH UNCERTAIN STATES

by

SAMUEL CERQUEIRA PINTO

B.S., Technological Institute of Aeronautics, 2017
M.S., Technological Institute of Aeronautics, 2018

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2021

Approved by

First Reader

Sean B. Andersson, PhD
Professor of Mechanical Engineering
Professor of Systems Engineering

Second Reader

Roberto Tron, PhD
Assistant Professor of Mechanical Engineering
Assistant Professor of Systems Engineering

Third Reader

Christos G. Cassandras, PhD
Distinguished Professor of Engineering
Professor of Electrical and Computer Engineering
Professor and Division Head of Systems Engineering

Fourth Reader

Julien M. Hendrickx, PhD
Professor of Mathematical Engineering
UC Louvain

As far as we are capable of knowledge
we sin in neglecting to acquire it. Gottfried Leibniz

Acknowledgments

First, I want to thank my advisor, prof. Sean B. Andersson. He gave me unconditional support over these years at Boston University, and guided me whenever I needed but also gave me enough freedom to pursue my own ideas and interests. He was always very patient to review my drafts and was always available to engage in discussions about the ideas we were pursuing. Then, I also want to thank especially profs. Christos G. Cassandras and Julien M. Hendrickx, that even though were not exercising the official role of advisors, they provided me with a great amount of support and guidance and were very patient in guiding me through my research. I also want to express my sincere gratitude to prof. Roberto Tron for being part of my thesis committee and for all the support and thoughtful comments that he has given me in the examinations that were part of my PhD.

One of my great joys of my doctorate was to meet such an excellent group of fellow students, that besides collaborating with my research work and classes, also made my days here more enjoyable and fun. In particular, I want to thank my collaborators Shirantha Welikala, Nicholas Vickers, Fatemeh Sharifi and Sean Sanchez for their hard and insightful work in the papers we coauthored. I also want to thank specially all members of the Andersson Lab for your support and great discussions, be it during formal meetings or over very informal conversations in the lab. I also want to thank my friends in the Robotics Lab that were very supportive throughout the entire process.

I am very thankful to the Department of Mechanical Engineering and Center for Information and Systems Engineering staff, especially Mrs. Patty Robinson-Angel and Christina Polyzos, for always helping me when I needed administrative support and for organizing such great social events, that definitively had a big impact in getting to know better the faculty and my fellow students.

Last, but not least, I want to extend my gratitude to my fiancée, my parents and siblings. Thanks for your understanding throughout this process: moving to another country, far from the family, comes with its own challenges. Travel restrictions made things even more difficult, but you always supported me in my decisions and I knew I could rely on you. Finally, I thank my friends from Elmbrook that were really an extended family for me in the United States.

PERSISTENT MONITORING OF TARGETS WITH UNCERTAIN STATES

SAMUEL CERQUEIRA PINTO

Boston University, College of Engineering, 2021

Major Professor: Sean B. Andersson, PhD
Professor of Mechanical Engineering
Professor of Systems Engineering

ABSTRACT

In a wide range of domains, such as pipeline inspection, surveillance in smart cities and tracking of multiple microparticles by an optical microscope, a common goal is to use mobile agents to persistently monitor a set of targets. We refer to this as the persistent monitoring problem. In this dissertation, we assume that each of these targets has an internal state that evolves with linear stochastic dynamics. The agents can observe these states when they are close to the targets, and the goal is to plan agent trajectories such that the sensed data can be used to minimize the uncertainty of the estimation process. We study scalable approaches for planning agent trajectories that minimize the long term uncertainty of the target states. We design algorithms that are computationally efficient and simple to implement, but grounded in mathematically proven performance guarantees.

First we approach the problem from a continuous time perspective with the goal of finding locally optimal agent trajectories using a gradient descent scheme. We assume that trajectories are fully defined by a finite set of parameters and compute the cost gradients. Considering periodic agent trajectories and an infinite time horizon, we

prove that, under some natural assumptions, the uncertainty of each target converges to a limit cycle. We also show that, in 1D environments with bounded controls, an optimal control is parametric. In multidimensional settings, we propose an efficient parameterization using Fourier curves. Simulation results show the efficiency of our approach.

Next, we consider a graph-constrained, single-agent version of the problem, where agents can only move in the edges of the graph and observe the target when they are visiting the node corresponding to it. We prove that, in this scenario, an optimal policy is such that all the agent have a common peak uncertainty. Using this property of the optimal solution, we develop lightweight algorithms that, instead of directly solving the optimization problem, balance the dwelling times to fulfill such property of an optimal policy. In some particular situations, global optimality of the proposed algorithm is proven. Using a custom-designed greedy exploration scheme, we develop an efficient method for obtaining efficient target visiting sequences. We extended this approach to multi-agent scenarios by using a divide-and conquer strategy, where targets are divided in clusters and each of these clusters is only visited by one agent.

Then, we extend those ideas to a discrete time version of the problem. We show that, for a periodic trajectory with fixed cycle length, the problem can be formulated as set of semidefinite programs. This allowed us to leverage efficient SDP solvers to provide fast solutions to the persistent monitoring problem. We design a scheme that leverages the spatial configuration of the targets to guide the search over this set of optimization problems to provide efficient trajectories.

Finally we describe an application of the proposed techniques to the problem of tracking multiple diffusing particles using a feedback-driven confocal microscope. The proposed persistent monitoring algorithm was used as the higher level controller in a hierarchical scheme, defining which particle should be tracked at each instant. Then

an extremum seeking controller was used as a lower level controller in order to track the moving particle and provide efficient observations.

Contents

1 Introduction **1**

1.1 The Persistent Monitoring Problem 1

1.2 Literature Review 2

1.2.1 Related Problems 2

1.2.2 Persistent Monitoring Formulations 4

1.3 Contributions 11

2 Problem Formulation and Steady State Optimization **16**

2.1 Problem Formulation 16

2.2 Transient Optimization 21

2.3 Steady State Persistent Monitoring 23

2.4 Steady State Gradients 25

2.5 Parameterization of an Optimal Trajectory in 1-D with speed bounds 31

2.5.1 Computation of $\frac{\partial s_j(q)}{\partial \theta}$ 36

2.5.2 Initialization of the Optimization 37

2.5.3 1D Simulation Results 38

2.6 Fourier Curves for Multi-Dimensional Persistent Monitoring 41

2.6.1 Initialization 42

2.6.2 2D and 3D Simulation Results 44

3 Minimax Persistent Monitoring Embedded on a Graph **47**

3.1 Infinity norm cost function 48

3.2 Properties of an Optimal Policy 50

3.2.1 Target's Perspective of a Periodic Policy 50
3.2.2 Necessary Condition for Optimality 52
3.3 Optimal Dwelling Sequence on a Constrained Visiting Sequence . . . 60
3.4 Optimal Dwelling Sequence on an Unconstrained Visiting Sequence . 71
3.5 A Greedy Solution for Determining an Optimal Visiting Sequence . . 75
3.5.1 The metric used to evaluate a visiting sequence 76
3.5.2 Possible types of modifications for a visiting sequence 79
3.5.3 Greedy Algorithm 84
3.5.4 Spectral Clustering Based Graph Partitioning 85
3.6 Extension to Multi-Agent Problems 88
3.6.1 Target-exchange scheme (TES) used to refine sub-graphs . . . 91

4 Discrete Time Formulation 104

4.1 Discrete Time Model 105
4.2 An Optimization Approach for Computing the Infinite Horizon Cost . 106
4.3 Optimization of Persistent Monitoring Schedules 111
4.4 Simulation Results 117
4.4.1 Discussion 120

5 Application to Multiple Particle Tracking 122

5.1 Brief Background on Multiple Particles Tracking 122
5.2 Problem Statement 124
5.2.1 Proposed Solution 126
5.3 Extremum Seeking Single Particle Tracking 127
5.4 Scheduling Multiple Particle Tracking 131
5.5 Simulation and Results 132
5.5.1 Trajectory estimation from photon data 136

6 Conclusions and Future Work **139**

6.1 Conclusion 139

6.1.1 Future Work 140

A Conditions on the existence of gradients of the steady state covariance matrix **142**

References **145**

Curriculum Vitae **152**

List of Tables

3.1 Parameters used in the simulation. 70

3.2 Parameters used in the simulation of unconstrained visiting sequence. 75

3.3 Performance comparison of different agent control methods under different persistent monitoring problem setups 94

5.1 Summary of differences between the MPT setting and the PM model, and the assumptions being used in order to apply PM to the MPT model. 132

5.2 Mean number of collected photons per second normalized by $I_{0,i}$ for 100 runs of each of the simulation setups. 135

5.3 RMSE Estimation error of 100 simulation runs. 138

List of Figures

2·1 Results of a simulation with two agents and five targets. (a) Evolution of the overall cost as a function of iteration number on the gradient descent. (b) Trajectories of the agents at the final iteration. The dashed lines indicate the positions of the targets and the grey shaded area the visibility region of the agent. (c) Evolution of the trace of the estimation covariance matrices of the five targets. 40

2·3 Simulation results in a 3D environment with two targets and ten agents. In red, the initial trajectory in the gradient descent optimization, in blue, the trajectory at the end of the optimization. The projection of the final agent trajectories in three planes is plotted in dashed purple. 46

2·4 Evolution of the cost function in the gradient descent optimization in the 3D scenario. 46

3·1 Temporal evolution of the steady state covariance matrix and waiting/observation times. 52

3·2 Illustration of the proof of Proposition 9. 66

3·3 Results of simulating Algorithm 3. In (a), the balanced peak uncertainty, as a function of the cycle period. The red dots mark the values of T that were explored by the golden ratio search. In (b)-(c), we show the evolution of the peak uncertainty and the dwell-time for each target. 69

3·4 Results obtained after optimizing the visiting and dwelling sequences. 70

3·5 Results obtained after optimizing the unconstrained visiting and dwelling sequences. 75
3·6 Three types of cycle modification operations (CMOs). 81
3·7 An example for CMO Type - I. Here, the edge $(1^1, 3^1)$ is replaced by the shortest path $(1^1, 4^2), (4^2, 3^1)$. 82
3·8 Examples for CMO Type II and III. In (c), target 1^2 was inserted to the initial cycle (b). In (d), targets 1^2, 4^3 were added to the initial cycle (b). 83
3·9 Example 1: The Greedy Cycle Construction Process. 95
3·10 Example 2: A Constructed Greedy Cycle. 96
3·11 Example 3: A Constructed Greedy Cycle. 96
3·12 Three types of cycle expansion operations (CEOs). 97
3·13 Greedy cycle expansion process for the computation of disparity values w.r.t. target 1: $\{d(1, j) : j \in \mathcal{V}\}$ using Alg. 7. The red contours in (b)-(g) show the expanded cycle $\bar{\Xi}'$ after executing Steps 5-9 of Alg. 7 with $i = 1$. 98
3·14 Clustering results and the greedy cycles constructed in each sub-graph for individual agents. 99
3·15 Clustering results (for the graph in Fig. 3·14(a)) when the shortest path distance is used as the disparity metric. 100
3·16 Target Exchange Scheme (TES) Example 1. Initial set of sub-graphs (in (b)): $\hat{J}(\mathcal{G}) = 16.3 = \max\{16.3, 6.9, 9.7\}$. Final set of sub-graphs (in (d)): $\hat{J}(\mathcal{G}) = 11.6 = \max\{11.2, 11.6, 10.9\}$ (Balanced and improved by 28.8%). 101

3·17 Target Exchange Scheme (TES) Example 2. Initial set of sub-graphs (in (b)): $\hat{J}(\mathcal{G}) = 12.3 = \max\{7.2, 12.3, 4.4\}$. Final set of sub-graphs (in (e)): $\hat{J}(\mathcal{G}) = 8.3 = \max\{7.7, 8.3, 5.6\}$ (Balanced and improved by 32.5%). 102

3·18 The persistent monitoring problem setups used in the performance comparison (at the initial condition). 103

4·1 Example graph for illustrating Alg. 10. 117

4·2 Results of the simulation with three targets. (a) Comparison of the trajectories generated by the *RRC* and the *SDP-PM* approaches. The trajectory displayed for RRC is the one with lowest cost among 5 independent runs of the algorithm. The presented trajectory was obtained after 200 iterations of the SDP-PM algorithm and 1500 iterations of *RRC*. The grey area represents the positions for which the agent can sense a given target. (b) Cost and cumulative computation time as a function of the iteration number for *SDP-PM*. (c) Cost and cumulative computation time as a function of the iterations of *RRC*. The solid lines represent average among 5 runs and the dashed lines are the observed maximum and minimum of the cost and computational time. None of the 5 instances of *SDP-PM* found a feasible solution before 345 iterations. 118

4·3 Simulation results in the setting containing 7 targets 120

5·1 Simulation of Extremum Seeking Controller trajectories starting at two different positions showing failure to converge (blue), and convergence (black). The arrows indicate the movement direction. 129

5·2 Number of cycles to convergence as a function of the initial relative position of the laser and particle. The initial distance is normalized by the radius R. 130
5·3 Mean squared tracking error as a function of K_p, for $f = 60$ Hz and particle diffusion coefficient $D_x = D_y = 0.1\ \mu\text{m}^2/\text{s}$. 130
5·4 Particle and laser trajectories, while tracking three particles in the first simulation scenario. The laser trajectory is in black and the particles are in blue, yellow and red. 134
5·5 Photons collected at each time step ($T_s = 10^{-4}$ s) in the first simulation scenario. The colors in the plot indicate which particle emitted those photon. 134
5·6 Illustration of a raster scanning trajectory considering a similar simulation setup. The agent trajectory is in black, while the particle trajectories are colored. 136
5·7 Estimation error over time using the offline estimator. The colors of the plots match the colors of the particle in Fig. 5·4. The shaded areas mark when the laser was orbiting around each particle. 137
5·8 Estimation error over time with perturbed parameters. The shaded areas mark when the laser was orbiting around each particle. 137

List of Abbreviations

CMO	.....	Cycle Modification Operations
ES	.....	Extremum Seeking
ESC	.....	Extremum Seeking Controller
KF	.....	Kalman Filter
MPT	.....	Multiple Particle Tracking
MSEE	.....	Minimum Squared Estimation Error
MTSP	.....	Multiple Traveling Salesman Problem
MVRP	.....	Multiple Vehicle Routing Problem
PM	.....	Persistent Monitoring
RRC	.....	Rapid-Exploring Random Cycle
RRT	.....	Rapid-Exploring Random Tree
TSP	.....	Traveling Salesman Problem

Chapter 1

Introduction

1.1 The Persistent Monitoring Problem

The general problem of multi-agent persistent monitoring involves a collection of mobile agents moving through a spatial domain to interact with targets at specific locations to, in some sense, control or monitor some state of those targets. We assume that the variable that one wants to control is inherently dynamic and that its evolution is affected by some stochastic noise. Therefore, this is not a static estimation problem, but rather a dynamic one, where these variables cannot be measured just once but rather must be monitored over time. Furthermore, we consider only cases where the number of available agents to do this monitoring is lower than the number of targets to be monitored, since in this case the agents have to move through the environment in order to periodically visit the targets, as opposed to having a static sensor being assigned to each target.

This paradigm finds applications across a wide range of domains, such as in ecological monitoring (Lin et al., 2018), infrastructure safety verification (Ostertag et al., 2019), ocean temperature surveillance (Lan and Schwager, 2016), deep-sea exploration (Alam et al., 2018) and multiple particle tracking (Pinto et al., 2021b).

The design goal is to obtain an optimal motion policy for the agents that minimizes a measure of the uncertainty in the estimates of the states of the targets. Therefore, the task of planning the agent trajectories is intrinsically related to the estimation of the targets states.

This general persistent monitoring description covers many different instances of the problem. For example, the evolution of the uncertainty over time can be described by different dynamics, the space can be continuous, discrete or even abstracted by a graph, the cost function can be any metric of the covariance matrix, etc. In the next sections of this introduction we explore different approaches the persistent monitoring (PM) problem and its relation to previous literature. We then place our specific formulation of the persistent monitoring problem within this larger context and describe how this dissertation contributes to the literature in the field.

1.2 Literature Review

1.2.1 Related Problems

This section gives a brief overview of problems that do not fully match the general description of the persistent monitoring problem, but have common features with it. This overview is by no means exhaustive and only reflects the problems that, in the author's judgement, are the most closely related to the persistent monitoring problem. Both the similarities and the distinctions between the related problems and persistent monitoring are highlighted.

This problem is closely related to the Multi Traveling Salesman Problem (MTSP) (Bektas, 2006) and Multi-Vehicle Routing Problem (MVRP) (Laporte, 2009), where, given a set of targets (possibly constrained to a graph-based structure), the goal is to find a cycle in which the agents efficiently visit all the targets in order to minimize the traveled distance or total travel time. These problems are proved to be computationally intractable (NP-hard) and most of the scalable solutions to these problems rely either on local optimization or heuristics (Pasqualetti et al., 2012; Bektas, 2006; Laporte, 2009). The major difference between the MTSP and MVRP and PM is that the optimization goal we consider is to minimize the uncertainty rather than

distance or time between two consecutive observations of a given target. Note that the estimation uncertainty is subject to a dynamic evolution in time. Thus, in PM the agents must not only visit targets, but also dwell on them for some time in order to control their uncertainty. Therefore, in addition to planning trajectories in space, PM also requires planning the time used to effectively collect the data.

Another closely related problem is the sensor allocation problem (Le Ny et al., 2010), where a set of sensors can observe a set of targets, but, due to the fact that the number of available sensors is lower than the total number of targets, some of the sensors have to switch among the targets they observe. The goal is to design a time multiplexing policy that minimizes the estimation error and an efficient globally optimal algorithms for this problem is given in (Le Ny et al., 2010). While in this case the optimization goal is equivalent to the PM objective, the sensor allocation formulation assumes that sensors are fixed and can instantly transition to observe different targets. As a result, it cannot model situations where there is a delay associated to transitioning to different targets. In fact, the optimal policy described in (Le Ny et al., 2010) consists in switching the target to be observed with an unbounded frequency. This, of course, is impractical for real world sensors.

The coverage control problem also has similarities with the persistent monitoring problem. There, agents are assumed to have a radius from which they can observe the environment. The goal is to cover as much of the environment as possible by efficiently spreading the agents around it (Wang, 2010; Sun et al., 2020). However, in this problem the agents eventually converge to fixed positions that maximize the overall coverage. This is a major contrast to persistent monitoring, where, for optimal solutions, agents do not converge to static positions, but rather move around the environment persistently.

Lastly, persistent monitoring also has a close link to the distributed estimation

problem (Olfati-Saber, 2007; Chong et al., 1982), where data from different sensors in the network is fused to estimate the state of variables of interest. While the estimation aspect of persistent monitoring can be approached as a distributed estimation problem, in the distributed estimation literature it is usually assumed that the agent trajectory is either fixed or known a priori. In the case of persistent monitoring, the goal is to design the agent trajectory such that maximum information can be extracted during the estimation process.

1.2.2 Persistent Monitoring Formulations

In this subsection, we give an overview of different formulations and approaches to the persistent monitoring problem available in the literature. In order to organize the vast literature in the field, some key aspects of the general persistent monitoring problem (such as type of dynamic evolution of the uncertainty, scalability of the proposed solution and time/space discretization) were identified. We identify how previous publications approach these different aspects and discuss how our present work relates to them.

Uncertainty dynamics and cost function

While the general goal of PM is persistently observe an uncertain variable, the definition of uncertainty and assumptions on how the variable evolves over time varies drastically within the persistent monitoring literature. Some works frame the problem as that of planning agent trajectories in order to maximize the chance of detecting events happening randomly at specific locations in the mission space (Pasqualetti et al., 2012; Baykal et al., 2020). In (Jones et al., 2015), there is no explicit uncertainty model, but temporal logic constraints are used to ensure that targets are visited sporadically, with inter-visit times defined by a frequency range. Note that this formulation does not explicitly define a cost function related to uncertainty and

as a result it lacks a clear notion of optimality. A recent work (Chen et al., 2020) does not assume a prior model for the uncertainty dynamics, but tries to learn it. It focuses on jointly designing trajectories and learning the target uncertainty evolution using reinforcement learning. However, this solution needs to be trained specifically for the mission space of interest, and thus, is computationally expensive and lacks generality. Other works consider a simple model for the uncertainty evolution over time, with it increasing with a constant rate when the target is not being observed and decreasing with a constant rate otherwise (Cassandras et al., 2013; Welikala and Cassandras, 2021a; Yu et al., 2018).

In this dissertation, however, we consider a version of Persistent Monitoring which was initially introduced in a very general sense in (Grocholsky et al., 2003), where the goal was to sense and estimate a dynamic processes evolving at fixed locations in space rather than detecting events. This paper models the variable of interest as a process evolving with a nonlinear dynamical model corrupted by noise. Additionally, it considers a nonlinear observation model, again corrupted by noise. Nonetheless, (Grocholsky et al., 2003) does not discuss efficient solutions to the problem proposed and the generality of the formulation hinders the ability to develop efficient computational solution methods.

Later work (Hussein, 2008; Lan and Schwager, 2014) assumes a more specific model, considering linear dynamics and observation models, with additive Gaussian noise. This is the model that we will consider in the present dissertation. It can be shown that, for this model, the maximum likelyhood estimator is a Kalman Filter (or Kalman-Bucy filter, in the continuous time case). This simplifies the problem in the sense that the optimal estimator can be designed completely decoupled from the agent trajectories. Moreover, the dynamics of the uncertainty are given by a differential Riccati equation. The optimization goal in these works is to minimize

Mean Squared Estimation Error (MSEE). They use an optimal control approach that relies on a solution of the two-point boundary value problem resulting from a Hamiltonian analysis.

Additionally, (Lan and Schwager, 2013; Lan and Schwager, 2016) introduces a variant of the Rapid-Exploring Random Tree (RRT) algorithm designed for cyclic Persistent Monitoring in discrete time, named Rapid-Exploring Random Cycle (RRC). In contrast to all the approaches cited so far that only consider the transient version of the problem, (Lan and Schwager, 2013; Lan and Schwager, 2016) explicitly aims to optimizes the steady state trajectory, and thus it only plans for one cycle of the periodic trajectory and the algorithm does not scale with the time horizon.

Mission Space

In terms of space discretization, the approaches in the literature can be divided into countinuous space, discretized space and graph-abstraction.

The continuous space formulations allow the agents to move throughout a continuous set. Some of these formulations assume the environment is obstacle free (Cassandras et al., 2013; Zhou et al., 2018; Lin and Cassandras, 2014). Other continuous space formulations are able to incorporate the presence of regions that the agent cannot move through into their formulation (Lan and Schwager, 2016; Wang et al., 2019). Moreover, some of these formulations, seeking computational or mathematical simplicity for their solutions, are restricted to one dimensional settings (Cassandras et al., 2013; Zhou et al., 2018; Ostertag et al., 2019). The discrete space formulation consists in discretizing the entire continuous space by a series of voxels (Chen et al., 2020). This approach works well when coupled with reinforcement learning, since learning over continuous space adds some additional challenges. Both continuous and discrete time formulations can easily consider that agents can sense targets even if their position do not coincide, a property that is true in many real-world sensors.

Abstracting the environment by a graph, where nodes are the targets and the edges represent the travel time between different targets, is also a common in the scientific literature (Pasqualetti et al., 2012; Welikala and Cassandras, 2020; Welikala and Cassandras, 2021b; Yu et al., 2018). By abstracting the environment using a graph, the problem of planning a trajectory is simplified to that a graph-exploration problem coupled to the decision of how long to dwell at each node (Yu et al., 2018; Yu et al., 2017; Welikala and Cassandras, 2020; Welikala and Cassandras, 2021b). Often these approaches assume that only one agent will visit a given target at each time instant and the solutions tend to be more computationally scalable than those presented for continuous or discrete settings. However, usually these formulations assume that the target can only be sensed when the agent visits that node, but a notable exception is (Zhou et al., 2019; Zhou et al., 2020), where no such assumption is made.

This present dissertation will approach the problem both from continuous space (Chapters 2 and 4) and graph-based model (Chapter 3) points of view. The solutions for continuous space formulations will carry a higher computational load, but allow for a more complete problem formulation. Meanwhile, the algorithm for designing persistent monitoring policies for the graph-based model is more efficient computationally, but does not give as much flexibility in the problem formulation.

Time discretization

In terms of time discretization, approaches in the literature are either assumed to be continuous (Grocholsky et al., 2003; Cassandras et al., 2013; Lan and Schwager, 2014; Yu et al., 2017) or discrete-time formulations (Lan and Schwager, 2013; Lan and Schwager, 2016). One noticeable fact is that the literature in (distributed) sequential Bayesian estimation (often the base for designing the estimation algorithms for the persistent monitoring problem, such as the Kalman Filter) are much more well devel-

oped for the discrete time case. It is also important to notice that many real world sensors operate in dicrete-time and some of them (such as LIDARs) indeed have very low rates compared to the dynamics of the processes they can be used to observe. Thus, in these cases a discrete-time formulation may be desirable. However, often solving the continuous time version of the problem allows solutions over the set of real numbers instead of requiring combinatorial (integer) decisions, and thus permits for more computationally efficient algorithms (Welikala and Cassandras, 2021b; Cassandras et al., 2013; Welikala and Cassandras, 2020). Also, many real world sensors (such as sonar) have a rate that is usually faster than the dynamics of the system they are used to observe, thus a continuous time approximation of its behavior is acceptable.

In this dissertation, we consider both a discrete time formulation in Chapter 4 and a continuous time one in Chapters 2 and 3.

Solution Approach

Some of the persistent monitoring formulations available do not focus on the scalability and computational cost, and therefore are limited to simple settings, with few targets and agents (Grocholsky et al., 2003; Smith et al., 2011; Chen et al., 2020). Some other works (Hussein, 2008; Lan and Schwager, 2014) approach the problem by an optimal control perspective and aim at directly solving the two-point boundary value problem resulting from the Halmitonian analysis. However the solution of the two-point value problem is numerically challenging and computationally expensive. More specifically, there is no algorithm that can ensure that a solution will be found for these numerical problems, even when one exists. Typical algorithms rely on shooting methods that involve the numerical solution of matrix differential equations. This imposes a computational burden that grows with the time horizon. Additionally, a Hamiltonian analysis only gives necessary conditions for optimality and it is

often the case that solutions encountered using these methods are severely suboptimal (Lin and Cassandras, 2014; Lan and Schwager, 2014), since this Hamiltonian analysis establishes necessary but not sufficient conditions for optimality.

Another class of approaches is based on sampling methods (Lan and Schwager, 2013; Lan and Schwager, 2016), inspired by the well known Rapid-Exploring Random Tree (RRT) algorithm. In this case, the space is randomly sampled and the resulting samples are arranged in a tree-based structure, so that when a full cycle is found it can be efficiently extracted from the structure. However, as shown in (Pinto et al., 2021a), this algorithm can fail to converge within a reasonable time even in setups with few targets and a single agent, especially when the process can only be sensed from a finite range from the targets. Note that the practical success of RRT is directly linked to the ability of biasing the samples towards the goal. However, the extension of this algorithm to cycles does not have the same capability of introducing efficient samples by using biased distributions, since the goal is not to reach a specific location, but rather a cycle. Therefore, the exploration is done in a completely blind manner, which hinders the practical applicability of this algorithm in many scenarios.

The approaches for planning persistent monitoring trajectories that will be explored in this dissertation largely depend on gradient-based optimization. These can be understood as a natural development of previous work done at Boston University, led by Andersson and Cassandras (Cassandras et al., 2013; Zhou et al., 2018; Yu et al., 2018), where the uncertainty metric for each of the targets that grew linearly with time when the agent was not observed or decreased linearly when an agent visited it. Using Hamiltonian analysis, these works have shown that in the uni-dimensional case, the optimal control admits a given finite dimensional parameterization. When multiple dimensions are considered, efficient parameterizations for the agents trajectories can also be obtained. This ability to parameterize the optimal trajectories

allowed for the usage of gradient-based methods, which are able to provide scalable solutions with respect to the number of agents, targets and time horizon, while still being numerically stable. A drawback of these gradient based methods is that they are not guaranteed to converge to global optima. In this dissertation we will, at least in some cases, give algorithms that achieve global optimality. Some previous works have studied initialization techniques aimed at escaping poor performing local optima (Zhou et al., 2018) and the distributed computation of the gradients (Zhou et al., 2020).

One of the main challenges of the gradient-based algorithms just mentioned is that their computational cost increases with the time horizon. However, when performing persistent monitoring, one wants to observe a variable for a very long amount of time. A work that was done concurrently with this thesis (Welikala and Cassandras, 2020) introduced a performance metric evaluated only on the asymptotic behavior of cyclic agent trajectories. This means that the transient performance was not part of the optimization. This approach overcomes the difficulty of planning for long time-horizons. However, it relied on one key assumption: that each target would only be observed by at most one agent over the entire mission. Given this assumption, a divide-and-conquer strategy was employed, and the targets were partitioned using clustering techniques. A similar divide-and-conquer strategy will also be used in this dissertation, as we developed lightweight algorithms with guaranteed global optimality for some specific settings in the single agent case.

A recent work (Welikala and Cassandras, 2021b) considered a receding horizon, distributed approach that only relied on lightweight computations. This algorithm is compatible with real-time implementation and produces very efficient trajectories in most of the scenarios analyzed. However, this approach has the drawback that, depending on the graph structure and the parameters that define the targets uncertainty

evolution, the agents may never visit a unstable target, making the infinite horizon uncertainty become unbounded. In our present dissertation, we focus on algorithms that guarantee that the infinite horizon uncertainty will be bounded.

1.3 Contributions

In this present dissertation, we consider the uncertainty model where each target has an internal state that evolves with linear stochastic dynamics corrupted by additive Gaussian noise and likewise each agent can observe the internal state with a linear observation model. The signal to noise ratio of the observation is a function of the distance between the agent and the target. In this setting, the maximum likelihood estimator is a Kalman-Bucy filter in the continuous time case (and a simple Kalman filter in the discrete time version) and the mean estimation error is directly related to the covariance matrix of this filter. Since the probability distribution associated to this filter is also Gaussian, it is fully defined by its mean and covariance. The covariance, in particular, captures entirely the concept of uncertainty, and thus is always present in the cost functions that we consider in this work.

In the first contribution of this dissertation, we use continuous time models, and study the asymptotic behavior of the covariance matrix under cyclic agent policies to understand under which conditions we can expect the agent trajectory to lead to a bounded estimation error over infinite time horizons. There we prove that, as long as the system has some controllability and observability properties, the covariance matrix will converge to a limit cycle that is independent of the initial conditions. Then, we explore the computation and existence of derivatives of the covariance matrix with respect to the agent trajectory parameters, both in the transient and steady-state versions of the problem. Note that the steady-state version is especially interesting because when the optimization is done in this setting, we overcome the

issue of having the computational cost scale with the time horizon.

We then study the minimization of the minimum squared error among all the targets. We show that, under certain conditions, an optimal trajectory in 1D environments is parametric. Moreover, we introduce an efficient parameterization for multi-dimensional scenarios based on Fourier curves. Using these parameterizations, we use gradient-based methods to optimize the trajectory. In order to provide an efficient initialization that leads to a bounded uncertainty, we take advantage of MTSP solutions. Then we use an optimization problem to transform this MTSP solution into agent trajectories parameters values.

Using these parameterizations, we provide tools to efficiently represent and optimize the schedules for agents visiting targets. If we consider finite horizon schedules, as time grows to infinity, the number of parameters to represent a trajectory also tends to grow infinitely large. We, however, restrict ourselves to a periodic trajectory and approach the problem from an infinite horizon perspective. We show that under some very natural assumptions the estimation error converges to a limit cycle and we provide tools for optimizing one period of the limit cycle trajectory. This approach is particularly useful, since the agent trajectory is usually represented using only a small number of parameters.

However, this gradient-based parametric control solution relies on the solution of $N \times M \times P$ matrix differential equations obtained at each gradient step, where N is the number of targets, M the number of agents and P the number of parameters used to describe the trajectory of an agent. Therefore, solving these matrix differential equations is a major computational burden for settings with many agents and targets. Our next contribution is to overcome such computational limitations by using simple, lightweight algorithms that can scale to networks with large numbers of targets. These algorithms are developed from a single-agent perspective and later extended to multi-

agent settings using a divide and conquer strategy. Further, unlike previously, we assume that the PM goal is to minimize the worst-case (instead of the average) uncertainty over all the target states. This choice of the PM goal not only leads to a considerable reduction in the computational burden, but is also an appropriate choice in many PM applications. For example, when monitoring safety-critical systems that cannot operate over a given threshold (for instance, a maximum temperature), one wants to optimize the "worst-case" performance (as opposed to optimizing an "average" chance of violating it). Some examples of applications where a critical threshold on the state uncertainty should not be exceeded include monitoring wildfire or faults in civil infrastructure systems using unmanned aerial vehicles (Lin et al., 2018; Shakhatreh et al., 2019).

For this formulation, we prove that, for a fixed sequence of target visits (*visiting sequence*), the worst-case uncertainty is *the same for all targets* when the agent uses the corresponding optimal sequence of dwell-times. This property alone is sufficient to determine the optimal dwelling sequence when each target in the considered visiting sequence is visited only once during a single *cycle*. In particular, this problem of determining the optimal dwelling sequence can be seen as a resource allocation problem where each target competes for the agent's dwell-time at that target. We next prove that a simple feedback law can be used to determine this optimal dwelling sequence efficiently. This same notion is then extended to the case where each target in the considered visiting sequence is allowed to be visited multiple times during a cycle. The only remaining problem, which we also address, is that of determining the optimal visiting sequence. We show that if each target is visited at most once during a cycle, a high-performing sub-optimal visiting sequence can be found by solving a Traveling Salesman Problem (TSP) and then executing a sequence of greedy Cycle Modification Operations (CMOs) on the obtained TSP solution.

The process of greedy cycle exploration requires an extensive number of evaluations of the cost of a given visiting sequence. Thus, we design a novel lower bound of the cost that does not require the computation of the dwelling times, and thus is much more efficient computationally and can be used as a proxy for the cost. Beyond using this lower bound on the greedy exploration, we also share ideas on how to use it to generate targets clusters, in order to extend our ideas to multi-agent scenarios using a divide-and-conquer strategy.

In addition, we explore a discrete time formulation of the problem, where both the computation of the steady state uncertainty and the local optimization of the trajectory can be framed as a single optimization problem, a semidefinite program (SDP). We then benefit from efficient and reliable SDP solvers and are able to quickly solve a local version of the persistent monitoring problem. Moreover, we are not limited to local optimality, as we have also embedded this local SDP-based optimizer into a higher level algorithm that searches globally for different periodic trajectories. This higher level scheme leverages the spatial distribution of the targets and then feeds the lower level optimization with configurations that will lead to feasible schedules. Due to the infinite number of candidate trajectories, we still are not able to guarantee global optimality. However, simulation results show that the approach proposed in this dissertation is able to efficiently handle problems with a small to moderate number of targets, providing trajectories with good performance even in the initial iterations of the higher level algorithm and also significantly improving them as it runs longer. Moreover, trajectories generated with the approach proposed here give a significant reduction (91%) in terms of estimation error when compared to RRC (Lan and Schwager, 2016) in a simple simulation scenario, while also showing significant computational time reduction.

The last contribution of this dissertation is to apply the minimax PM approach

into the domain of Multiple Particle Tracking (MPT) using a confocal feedback-driven microscope. In this case, the agent is the microscope laser that can move through the sample, acquiring a scalar signal (intensity) from the targets (particles). However, the targets in this scenario are not static, as assumed in the PM formulation, and the observations are not linear. Therefore, in order use to PM tools in this domain, we use a lower level extremum seeking controller (Ashley and Andersson, 2016) that is responsible for tracking a single particles during the dwelling time given by the PM planner.

Chapter 2

Problem Formulation and Steady State Optimization

In this chapter, we give the general formulation of the persistent monitoring problem that will be used throughout this dissertation. Small variations of this formulation will be discussed in the other chapters, and the distinctions from the formulation here presented will be clearly stated.

In addition to the formulation, in this chapter we study properties of the persistent monitoring problem as the time horizon goes to infinity. In particular, we describe sufficient conditions for the existence of a bounded covariance matrix and how to compute the gradients of this covariance matrix with respect to parameters of a parametric trajectory.

We introduce convenient parameterizations for both one dimensional and multi-dimensional scenarios. Then, using gradient-descent based techniques we optimize these trajectories, considering a cost given by the $\mathcal{L}-2$ norm of the uncertainty. An initialization scheme that leads to efficient locally-optimal trajectories is also given. The results in this chapter have been previously published in (Pinto et al., 2019; Pinto et al., 2020c; Pinto et al., 2020a; Pinto et al., 2020b).

2.1 Problem Formulation

Consider an environment with a set of M points of interest (targets) at fixed positions $x_i \in \mathbb{R}^P$, $i = 1, ..., M$. Each of these targets has an internal state $\phi_i \in \mathbb{R}^{L_i}$ that

needs to be monitored and that evolves according to linear time-invariant stochastic dynamics:

$$\dot{\phi}_i(t) = A_i\phi_i(t) + w_i(t), \tag{2.1}$$

where $w_i(t)$ is a white noise process distributed according to $w_i(t) \sim \mathcal{N}(0, Q_i)$, $i = 1, \ldots, M$, and $w_i(t)$ and $w_j(t)$ are statistically independent if $i \neq j$. Suppose that there is a collection of N mobile agents at positions $s_i(t) \in \mathbb{R}^P$ that can move with the following kinematic model:

$$\dot{s}_j(t) = u_j(t), \qquad u_j(t) \in \mathcal{U}, \qquad j = 1, ..., N, \tag{2.2}$$

where u_j is a controllable input, and $\mathcal{U}$ is the set of admissible inputs.

Each of these agents is equipped with sensors that can observe the targets according to the following model:

$$z_{i,j}(t) = \gamma_j\left(s_j(t) - x_i\right) H_i\phi_i(t) + v_{i,j}(t), \tag{2.3}$$

where $v_{i,j}(t)$ is a white noise process distributed according to $v_{i,j}(t) \sim \mathcal{N}(0, R_i)$ with $v_{i,j}(t)$ independent of $v_{k,l}$ if $i \neq k$ or $j \neq l$, and $\gamma_{i,j} : \mathbb{R}^N \mapsto \mathbb{R}$ is a function that captures the interdependence of measurement quality and the relative position from a given agent to a target. The intuition behind this function is that the instantaneous signal to noise ratio (SNR) can be computed as:

$$\frac{E\left[\left\|z_{i,j}(t) - v_{i,j}(t)\right\|^2\right]}{E[\left\|v_{i,j}(t)\right\|^2]} = \gamma_{i,j}^2\left(s_j(t) - x_i\right)\frac{\left\|H_i\phi_i(t)\right\|^2}{\mathrm{tr}(R_i)}, \tag{2.4}$$

where $\mathrm{tr}(\cdot)$ is the trace of the matrix. Notice that the term $\left\|H_i\phi_i(t)\right\|^2\left(\mathrm{tr}(R_i)\right)^{-1}$ is a deterministic scalar that does not depend on the relative position between the target and the agent. Therefore, the function $\gamma_{i,j}$ captures entirely how the position of the agent affects the quality of the measurement. It is worth noting that in most of the

applications of mobile agents to sensing there is a limited sensing range or the quality of the measurement gets worse as the agent moves farther away from the target. The general model of $\gamma_{i,j}$ is capable of capturing both the finite range and the dependence between measurement quality and relative position of the target from the agent. Even though the analysis in this dissertation does not depend on the specific $\gamma_{i,j}$, for the sake of concreteness we use the following form in this chapter:

$$\gamma_{i,j}(\alpha) = \begin{cases} 0, & \|\alpha\| > r_{i,j}, \\ \sqrt{1 - \frac{\|\alpha\|}{r_{i,j}}}, & \|\alpha\| \leq r_{i,j}. \end{cases} \tag{2.5}$$

The intuition behind this specific form is that the best measurement quality is achieved when the agent's location coincides with that of the target with the SNR decaying linearly as it moves away, until the agent reaches the distance of its sensing radius $r_{i,j}$, after which only noise is observed.

In this work we approach the problem from a centralized perspective. Therefore, at a given instant, the combined observations from all the agents of a single target can be grouped in a vector $\tilde{z}(t)$ as:

$$z_i(t) = \begin{bmatrix} z'_{i,1} & ... & z'_{i,N} \end{bmatrix}' = \tilde{H}_i(s_1, ..., s_n)\phi_i(t) + \tilde{v}_i(t), \tag{2.6}$$

where

$$\tilde{H}_i = \begin{bmatrix} \gamma_1(s_1 - x_i)H'_i & \cdots & \gamma_N(s_N - x_i)H'_i \end{bmatrix}', \qquad \tilde{v}_i(t) = \begin{bmatrix} v'_{i,1}(t) & ... & v'_{i,N}(t) \end{bmatrix}', \tag{2.7}$$

and, since $v_{i,j}(t)$ is independent of $v_{i,k}(t)$ if $k \neq j$, $E[\tilde{v}'_i(t)\tilde{v}_i(t)] = \tilde{R}_i = diag(R_i, ..., R_i)$. The overall goal is to obtain estimators $\hat{\phi}_i(t, z(t))$ and open-loop control inputs $u_j(t)$

to minimize the following cost function:

$$J = \frac{1}{t_f} \int_0^{t_f} \left(\sum_{i=1}^{M} E[e_i'(\zeta) e_i(\zeta)] + \xi \sum_{j=1}^{N} u_j'(\zeta) u_j(\zeta) \right) d\zeta, \tag{2.8}$$

where $e_i(t) = \hat{\phi}_i(t) - \phi_i(t)$ and t_f is the time horizon. This cost function represents a weighted sum of the mean squared estimation error and the control effort; thus, the weighting factor ξ is responsible for balancing the importance of these two optimization goals. The model in (2.7) defines a linear time-varying stochastic system. Based on a similar statement from (Lan and Schwager, 2014), we have the following proposition:

Proposition 1. *The optimal unbiased estimator $\hat{\phi}_i$ for the the cost function* (2.8)*, dynamics* (2.1) *and observation model* (2.3)*, with given trajectories $s_j(t)$ is the Kalman-Bucy filter, which is given by the following equations:*

$$\dot{\hat{\phi}}_i(t) = A_i \hat{\phi}_i(t) + \Omega(t)_i \tilde{H}_i'(t) \tilde{R}_i^{-1} \left(\tilde{z}_i(t) - \tilde{H}_i(t) \hat{\phi}_i(t) \right), \tag{2.9a}$$

$$\dot{\Omega}_i(t) = A_i \Omega_i(t) + \Omega_i(t) A_i' + Q_i - \Omega_i(t) \tilde{H}_i' \tilde{R}_i^{-1} \tilde{H}_i \Omega_i(t), \tag{2.9b}$$

where $\Omega_i(t)$ is the covariance matrix of the estimator.

Proof. The set of all linear unbiased estimators $\hat{\phi}_i(t)$ of $\phi_i(t)$, as discussed in Sec. IV of (Athans and Tse, 1967), is:

$$\dot{\hat{\phi}}_i(t) = \left(A_i - G_i(t) \tilde{H}_i(t) \right) \hat{\phi}_i(t) + G_i(t) \tilde{z}_i(t), \tag{2.10}$$

with $E[\hat{\phi}_i](0) = E[\phi_i(0)]$ and $G(t)$ is a gain function. If $\Omega_i(t) = E[e_i(t) e_i'(t)]$, where $e_i = \hat{\phi}_i(t) - \phi_i(t)$, then

$$\begin{aligned} \dot{\Omega}_i(t) = &\left(A_i - G_i(t) \tilde{H}_i(t) \right) \Omega_i(t) + G_i(t) \tilde{R}_i G_i'(t) \\ &+ Q_i + \Omega_i(t) \left(A_i' - \tilde{H}_i(t)' G_i'(t) \right), \end{aligned} \tag{2.11}$$

and $\Omega_i(0) = \Omega_{i,0}$. Defining the following cost:

$$J = \int_0^{t_f} \left(\sum_{i=1}^{M} \text{tr}\left(\Omega_i(t')\right) + \beta u'(t')u(t') \right) dt' \tag{2.12}$$

The Hamiltonian is then

$$\mathcal{H} = \sum_{i=1}^{M} \text{tr}\left(\Omega_i(t)\right) + \beta u_j'(t)u_j(t) + \sum_{i=1}^{M} \text{tr}\left(\Gamma_i(t)\dot{\Omega}_i(t)\right) + \sum_{j=1}^{N} \alpha_j(r)s_j(t), \tag{2.13}$$

where Γ_i is the costate of Ω_i and $u_j(t)$ and $s_j(t)$ are considered to be known. Using Pontryagin's minimum principle, at an optimal trajectory, since G_i is unconstrained, we have

$$\frac{\partial \mathcal{H}^\star}{\partial G_i} = 0. \tag{2.14}$$

Substituting the dynamics of the covariance matrix (2.11) on (2.14), we get

$$-\Gamma_i\Omega_i\tilde{H}_i' - \Gamma_i'\Omega_i\tilde{H}_i' + \Gamma_i'G_i\tilde{R}_i + \Gamma_iG_i\tilde{R}_i = 0. \tag{2.15}$$

Now, again from the minimum principle,

$$\dot{\Gamma}_i = -\frac{\partial \mathcal{H}}{\partial \Omega_i} - (A_i - G_i\tilde{H}_i)'\Gamma_i - \Gamma_i(A_i - G_i\tilde{H}_i) - I. \tag{2.16}$$

Since $\Gamma_i(t_f) = 0$ due to the boundary conditions of Pontryagin's minimum principle, the symmetric nature of this ODE allow us to see that Γ_i will be symmetric for $t \in [0, t_f]$. Moreover, note that the ODE is linear and the single non-homogeneous term is -I. Since $\Gamma_i(t_f) = 0$,

$$\Gamma_i(t) = -\int_{t_f}^{t} \Phi'(t, t_f)\Phi(t, t_f)dt, \qquad \Phi(a, b) = \exp\left(\int_a^b (A_i - G(\beta)\tilde{H}_i(\beta))d\beta \right). \tag{2.17}$$

This implies that $\Gamma_i(t) \succ 0$ for $t \in [0, t_f)$. Therefore, since $\Gamma_i(t)$ is invertible and symmetric, (2.15) can be reduced to

$$\Omega_i\tilde{H}_i' + \Omega_i'\tilde{H}_i' = 2G_i\tilde{R}_i. \tag{2.18}$$

Since the covariance matrix Ω_i is also symmetric,

$$G_i(t) = \Omega_i(t)\tilde{H}_i(t)\tilde{R}_i^{-1}(t). \tag{2.19}$$

Plugging this expression into (2.11) and (2.10), we get the usual Kalman-Bucy filter equations, which along with the initial conditions $\Omega_i(0) = \Omega_{i,0}$ and $\hat{\phi}_i(0) = E[\phi_i(0)]$, have unique solutions. □

Using (2.7), we can rewrite (2.9b) as:

$$\dot{\Omega}_i(t) = A_i\Omega_i(t) + \Omega_i(t)A_i' + Q_i - \Omega_i(t)G_i\Omega_i(t)\sum_{j=1}^{N}\gamma_{i,j}^2(t), \tag{2.20}$$

where $G_i = H_i'R_i^{-1}H_i$ and $\gamma_{i,j}(t) = \gamma_{i,j}(s_j(t) - x_i)$. Using the fact that $E\left[e_i'(t)e_i(t)\right]$ $= \mathrm{tr}(E\left[e_i(t)e_i'(t)\right]) = \mathrm{tr}(\Omega_i(t))$, we can rewrite the cost function in (2.8) as

$$J = \frac{1}{t_f}\int_0^{t_f}\left(\sum_{i=1}^{M}\mathrm{tr}(\Omega_i(\zeta)) + \xi\sum_{j=1}^{N}u_j'(\zeta)u_j(\zeta)\right)d\zeta. \tag{2.21}$$

The goal is then to minimize the cost (2.21) subject to the dynamics in (2.20) and (2.2). In other words, we aim to design a trajectory that minimizes a weighted sum of the mean control effort and the mean estimation error, given that the control is such that $u_j \in \mathcal{U}$. The estimation error is linked to the trajectory through the dynamics of the covariance matrix of the Kalman-Bucy Filter.

2.2 Transient Optimization

Even tough we focus on the optimization of infinite horizon trajectories, we briefly review the procedure for optimizing trajectories with finite time horizon in order to later extend to the infinite horizon setting. In this section, we establish a general formulation and in the next sections we approach specific settings. We assume that the agent trajectories can be fully defined by a finite set of parameters since, as will

be discussed for specific setups, parameterizations tend to naturally fit the persistent monitoring problem. In this chapter, we aim at computing locally optimal solutions with respect to these parameters using gradient descent, and later in this dissertation we target more broad notions of optimality. Therefore, we initially discuss how to compute the gradients for the finite horizon version of the problem. We define the set of parameters that fully describe the trajectory for $t \in [0, t_f]$ as $\Theta = \{\theta_1, ..., \theta_D\}$. Recalling the expression for the cost (2.21), we can compute the partial derivative with respect to one of the parameters of the trajectory θ_d as:

$$\frac{\partial J}{\partial \theta_d} = \frac{1}{t_f} \int_0^{t_f} \left(\sum_{i=1}^{M} \text{tr} \left(\frac{\partial \Omega_i}{\partial \theta_d}(\zeta) \right) + \xi \sum_{j=1}^{N} \frac{\partial (u_j' u_j)}{\partial \theta_d}(\zeta) \right) d\zeta. \tag{2.22}$$

Note further that, given the dynamics of the covariance matrix in (2.20), $\frac{\partial \Omega_i}{\partial \theta_d}$ is the solution of the following ODE:

$$\frac{\partial \dot{\Omega}_i(t)}{\partial \theta_d} = A_i \frac{\partial \Omega_i}{\partial \theta_d}(t) + \frac{\partial \Omega_i(t)}{\partial \theta_d} A_i' + Q_i - \left(\frac{\partial \Omega_i(t)}{\partial \theta_d} G_i \Omega_i(t) + \Omega_i(t) G_i \frac{\partial \Omega_i(t)}{\partial \theta_d} \right) \sum_{j=1}^{N} \gamma_{i,j}^2(t) - \Omega_i(t) G_i \Omega_i(t) \sum_{j=1}^{N} \frac{\partial \gamma_{i,j}^2}{\partial \theta_d}(t), \tag{2.23}$$

with initial conditions $\frac{\partial \Omega_i}{\partial \theta_d}(0) = 0$. Also, we know that

$$\frac{\partial \gamma_{i,j}^2(t)}{\partial \theta} = \sum_{p=1}^{P} \frac{\partial \gamma_{i,j}^2(t)}{\partial s_j^{e_p}} \frac{\partial s_j^{e_p}(t)}{\partial \theta_d}, \tag{2.24}$$

where e_p, $p = 1, ..., P$ is the p-th coordinate of the space the agents move in. Given the specific definition of $\gamma_{i,j}$ in (2.5), we can easily see that

$$\frac{\partial \gamma_{i,j}^2}{\partial s_j^{e_i}} = \begin{cases} \frac{s_j^{e_p} - x_i^{e_p}}{r_j \|s_j - x_i\|}, & \text{if } \|s_j - x_i\| < r_j, \\ 0, & \text{otherwise.} \end{cases} \tag{2.25}$$

The only terms that we have not yet given a procedure to compute are $\frac{\partial(u_j' u_j)}{\partial \theta_d}(t)$ and $\frac{\partial s_j^{ep}}{\partial \theta_d}(t)$. The computation of both of these terms is intrinsically related to the specific parameterization chosen and details of their computation will be discussed later in this work.

2.3 Steady State Persistent Monitoring

For a persistent monitoring task to be successful, it is necessary that targets are visited infinitely often as time goes to infinity, because otherwise their uncertainty can become unbounded. Periodicity naturally fits into the persistent monitoring paradigm, since targets need to be visited infinitely often and, although a periodic structure of the solution is not necessarily optimal, simulation results in the transient case show that the trajectories tend to converge to oscillatory behavior (Pinto et al., 2019). On top of that, previous results show that in the discrete time version of this problem, periodic schedules can approximate arbitrarily well the cost of an optimal schedule (Zhao et al., 2014) and it is natural to extend this result to the continuous time setting. Moreover, if periodicity is assumed, the infinite horizon trajectory is fully defined by the trajectory of a single period. This often leads to the need of only a very small number of parameters to describe the infinite horizon trajectory and, as a consequence, only a small number of parameters have to be optimized in order to generate efficient trajectories. With that in mind, we explore the properties of periodic solutions to the persistent monitoring problem when the system fulfills the following very natural assumptions.

Assumption 1. *The pair (A_i, H_i) is detectable, for every $i \in \{1, ..., M\}$.*

Assumption 2. *Q_i (the covariance matrix of the additive noise of the internal state dynamics) and the initial internal state covariance matrix $\Sigma_i(0)$ are positive definite, for every $i \in \{1, ..., M\}$.*

The intuition behind the first assumption is that sensing can make the uncertainty of each target bounded even for long horizons. The second one guarantees that the covariance matrix will always be positive definite, a fact that will be used to prove Prop. 3, that gives insights into the computation of the steady-state covariance derivatives. Under these assumptions, first we explore conditions under which the convergence of the covariance matrix is achieved. For the sake of notation conciseness, we define

$$\eta_i(t) = \sum_{j=1}^{N} \gamma_{i,j}^2(t), \tag{2.26}$$

which represents the instantaneous power level of the sensed signal, combining all the agents' observations of the same target i. Using a procedure similar to the one used in the proof of *Lemma 9* in (Le Ny et al., 2010), we establish the following proposition:

Proposition 2. *If $\eta_i(t)$ is T-periodic and $\eta_i(t) > 0$ for some non-degenerate interval $[a, b] \in [0, T]$, then, under Assumption 1, there exists a unique non-negative stabilizing T-periodic solution to* (2.20).

Proof. According to (Bittanti et al., 1984, p. 130), a pair $(A_i, \eta_i(t)H_i)$ of a periodic system is detectable if and only if for every eigenpair (x, λ) with $x \neq 0$,

$$A_i x = \lambda x \implies \exists\ [a, b] \in [0, T]\ s.t.\ \eta_i(t) e^{\lambda t} H_i x \neq 0, \tag{2.27}$$

$\forall t \in [a, b]$ and $[a, b]$ is non-degenerate. Notice that, due to Assumption 1, for any eigenvector x of A_i, $H_i x \neq 0$. Therefore, when $\eta_i(t) > 0$ (i.e. any $t \in [a, b]$), $\eta_i(t) e^{\lambda t} H_i x \neq 0$, which implies that $(A_i, \eta_i(t)H_i)$ is detectable. Therefore, the collorary to Theorem 3 in (Nicolao, 1992, p. 95) shows that there exists a non negative T-periodic solution to (2.20), $\bar{\Omega}_i(t)$, and

$$\lim_{t \to \infty} (\Omega_i(t) - \bar{\Omega}_i(t)) = 0$$

for any solution $\Omega_i(t)$ with positive definite initial condition $\Omega_i(0)$. □

Prop. 2 implies that, if $\eta_i(t)$ is periodic, given any initial covariance matrix $\Omega_i(0)$, the estimation covariance for target i converges to a T-periodic matrix $\bar{\Omega}_i(t)$, as long

as target i is visited for some non-zero amount of time in the periodic trajectory. Therefore,

$$\forall \delta > 0, \exists\ t_0\ s.t.\ |\tilde{\Omega}_i(t) - \Omega_i(t)| \leq \delta,\ \forall t \geq t_0,$$

which implies that

$$\lim_{t\to\infty} \frac{1}{t} \int_0^t |\mathrm{tr}(\tilde{\Omega}_i(t') - \Omega_i(t'))|\, dt' \leq \delta. \tag{2.28}$$

This discussion implies that, if we run a periodic trajectory for long enough, the mean estimation error will become arbitrarily close to the mean steady state estimation error. Therefore, if we plan only (one period of) the steady state trajectory, the actual estimation error will be arbitrarily close to that of the planned trajectory as time goes to infinity. Even though Prop. 2 states that the solution of the periodic Riccati equation is globally attractive, it does not provide any convergence rate for its numerical computation. However, the problem of computing numerical solutions to this Riccati equation has been studied in other works and we refer the reader to (Varga, 2013) for a good review and discussion of these methods.

Similarly to the transient case, we intend to optimize the trajectory of the agents using gradient descent. However, the computation of the steady state gradients of the covariance matrix is more challenging than the transient case. In the sequel, we provide the procedure to compute these gradients when they exist.

2.4 Steady State Gradients

Assuming that the trajectory is periodic and all the targets are visited, we introduce the change of variable $q = t/T$, where T is the period of the trajectory. The steady

state cost can be rewritten as:

$$J = \int_0^1 \left(\sum_{i=1}^{M} \mathrm{tr}(\bar{\Omega}_i(q)) + \xi \sum_{j=1}^{N} \bar{u}_j'(q)\bar{u}_j(q) \right) dq, \tag{2.29}$$

where $\bar{u}(q) = u(qT)$. Similar to (2.22), we know that, given some parameter $\theta_d \in \Theta$:

$$\frac{\partial J}{\partial \theta_d} = \int_0^1 \left(\sum_{i=1}^{M} \mathrm{tr}\left(\frac{\partial \bar{\Omega}_i(q)}{\partial \theta} \right) + \xi \sum_{j=1}^{N} \frac{\partial (\bar{u}_j' \bar{u}_j)(q)}{\partial \theta_d} \right) dq. \tag{2.30}$$

The dynamics of $\bar{\Omega}_i(q)$ are

$$\dot{\bar{\Omega}}_i(q) = \frac{d\bar{\Omega}_i(q)}{dq} = T(A\bar{\Omega}_i(q) + \bar{\Omega}_i(q)A' + Q - \eta_i(q)\bar{\Omega}_i(q)G\bar{\Omega}_i(q)). \tag{2.31}$$

If we want to compute the gradient with respect to a parameter θ_d, we get the following dynamics (note that the period may be a function of the parameters or a parameter itself):

$$\frac{\partial \dot{\bar{\Omega}}_i(q)}{\partial \theta_d} - T\left(A\frac{\partial \bar{\Omega}_i(q)}{\partial \theta_d} + \frac{\partial \bar{\Omega}_i(q)}{\partial \theta_d}A' - \eta_i(q)\bar{\Omega}_i(q)G\frac{\partial \bar{\Omega}_i(q)}{\partial \theta_d} - \eta_i(q)\frac{\partial \bar{\Omega}_i(q)}{\partial \theta_d}G_i\bar{\Omega}_i(q) \right) = T\frac{\partial \eta_i(q)}{\partial \theta_d}\bar{\Omega}_i(q)G_i\bar{\Omega}_i(q) + \frac{\partial T}{\partial \theta_d}\frac{\dot{\bar{\Omega}}}{T}. \tag{2.32}$$

If the partial derivative $\partial\bar{\Omega}_i(q)/\partial\theta_d$ exists, it is a 1-periodic solution of (2.32), since $\bar{\Omega}_i$ itself is periodic with period one. We define the following auxiliary problems:

$$\dot{\Sigma}_H - T\left(A - \eta_i\bar{\Omega}_iG\right)\Sigma_H = 0,\ \Sigma_H(0) = I, \tag{2.33}$$

$$\dot{\Sigma}_{ZI} - T\left(A - \eta_i\bar{\Omega}_iG\right)\Sigma_{ZI} - T\Sigma_{ZI}'\left(A - \eta_i\bar{\Omega}_iG\right)' = T\frac{\partial \eta_i}{\partial \theta}\bar{\Omega}_iG\bar{\Omega}_i + \frac{\partial T}{\partial \theta_d}\frac{\dot{\Omega}_i}{T},\ \Sigma_{ZI}(0) = 0, \tag{2.34}$$

where the time dependence of $\eta_i(q),\ \Omega_i(q), \Sigma_{ZI}(q)$ and $\Sigma_H(q)$ was omited for concise-

ness. Then, we can use the following proposition to compute the partial derivatives with respect to the parameters that define the trajectory:

Proposition 3. *Suppose* Σ_H *is a solution of* (2.33)*,* Σ_{ZI} *is a solution of* (2.34)*, Assumptions* 1 *and* 2 *hold, and that target* i *is observed at least once in the period* T*. Then, the equation*

$$\Lambda = \Sigma_H(1)\Lambda\Sigma_H'(1) + \Sigma_{ZI}(1) \tag{2.35}$$

has a unique solution Λ*. Additionally, when* $\frac{\partial\bar{\Omega}_i(q)}{\partial\theta_d}$ *exists,*

$$\frac{\partial\bar{\Omega}_i(q)}{\partial\theta_d} = \Sigma_H'(q)\Lambda\Sigma_H(q) + \Sigma_{ZI}(q).$$

Proof. Suppose Λ and $\tilde{\Lambda}$ are solutions of (2.33), then

$$\Lambda - \tilde{\Lambda} = \Sigma_H(1)\left(\Lambda - \tilde{\Lambda}\right)\Sigma_H'(1) \tag{2.36}$$

which is equivalent to

$$vec\left(\Lambda - \tilde{\Lambda}\right) = \left(\Sigma_H(1) \otimes \Sigma_H(1)\right) vec\left(\Lambda - \tilde{\Lambda}\right), \tag{2.37}$$

where $vec(\cdot)$ is the operator the performs the matrix vectorization and $\otimes$ represents the matrix Kronecker product. Notice that $\Lambda = \tilde{\Lambda}$ is a solution of (2.37). This solution is the unique solution if and only if 1 is not an eigenvalue of $\Sigma_H(1) \otimes \Sigma_H(1)$. On the other hand, the eigenvalues of $\Sigma_H(1) \otimes \Sigma_H(1)$ are all in the form $\mu_1\mu_2$, where μ_1 and μ_2 are distinct eigenvalues of $\Sigma_H(1)$ (Zhang, 2011).

In the following we show that all the eigenvalues of $\Sigma_H(1)$ have absolute value lower than one. For that, first notice that since Q is positive definite, $\bar{\Omega}_i$ is also positive definite and hence, invertible. Define

$$\mathcal{W} = \bar{\Omega}_i^{-1},$$

and, since $\dot{\mathcal{W}} = -\bar{\Omega}_i^{-1}\dot{\bar{\Omega}}_i\bar{\Omega}_i^{-1} = -\mathcal{W}\dot{\bar{\Omega}}_i\mathcal{W}$, using (2.20) and (2.26), the dynamics of $\mathcal{W}$ can be expressed as:

$$\dot{\mathcal{W}} = -T(\mathcal{W}A + A'\mathcal{W} + \mathcal{W}Q\mathcal{W} - \eta_i G). \tag{2.38}$$

Therefore, if we define the Lyapunov Function $V = \Sigma_H' \mathcal{W} \Sigma_H$, we have that:

$$\begin{aligned} \frac{d}{dq}\left(\Sigma_H' \mathcal{W} \Sigma_H\right) &= \Sigma_H' \left(T\mathcal{W}A + TA'\mathcal{W} + T\eta_i G + \dot{\mathcal{W}}\right) \Sigma_H \\ &= -T\Sigma_H' \mathcal{W} Q \mathcal{W} \Sigma_H. \end{aligned} \tag{2.39}$$

By integrating the previous relation, we have

$$\Sigma_H'(1)\mathcal{W}(1)\Sigma_H(1) - \Sigma_H(0)\mathcal{W}(0)\Sigma_H(0) = -T \int_0^1 \Phi(q,0)'\mathcal{W}Q\mathcal{W}\Phi(q,0)\, dq, \tag{2.40}$$

where $\Phi(q_1, q_2)$ is the transition matrix of the system (2.33) betwen times q_1 and q_2. Moreover, since $\bar{\Omega}_i(q)$ is periodic with period one and $\Sigma_H(0) = I$, we have that

$$\Sigma_H'(1)\mathcal{W}(0)\Sigma_H(1) - \mathcal{W}(0) = -T \int_0^1 \Phi(q,0)'\mathcal{W}Q\mathcal{W}\Phi(q,0)\, dq. \tag{2.41}$$

Note that $\mathcal{W}\Phi(q,0)$ is full rank on a nontrivial set, since $\mathcal{W}$ is positive definite and $\Phi(q,0)$ is full rank for at least a non-degenerate interval due to Assumption 1 and the fact that target i is observed at least once in an period. This, along with the fact that Q is positive definite, implies that the integral in (2.41) will be a positive definite matrix. Therefore,

$$\Sigma_H'(1)\mathcal{W}(0)\Sigma_H(1) - \mathcal{W}(0) \prec 0. \tag{2.42}$$

Consequently, one can see that

$$\frac{(\Sigma_H(1)x)'\mathcal{W}(0)(\Sigma_H(1)x)}{x'\mathcal{W}(0)x} < 1, \tag{2.43}$$

for every nonzero x. Since $\mathcal{W}(0)$ is positive definite, (2.43) shows that the norm of the matrix $\Sigma_H(1)$ induced by $\mathcal{W}(0)$ (i.e., $\|\Sigma_H(1)\|_{\mathcal{W}}(0)$) is less than 1, therefore its spectral radius is smaller than 1. This implies that the absolute value of all the eigenvalues of $\Sigma_H(1)$ are smaller than 1. Hence, $\Sigma_H(1) \otimes \Sigma_H(1)$ is stable, and $\Lambda = \tilde{\Lambda}$.

Moreover, (2.35) has one solution given by

$$\Lambda = \sum_{j=1}^{\infty} \left(\Sigma_H(1)\right)^j \Sigma_{ZI}(1) \left(\Sigma_H(1)'\right)^j . \tag{2.44}$$

We point out that the sum in (2.44) converges, since the absolute value of the eigenvalues of $\Sigma_H(1)$ are all lower than 1.

Now, note that (2.32) is a first order linear matrix differential equation and its general solution is given by

$$\Sigma(q) = \Sigma_H'(q)\Sigma(0)\Sigma_H(q) + \Sigma_{ZI}(q). \tag{2.45}$$

Since there is a unique solution to (2.45), and when $\partial\bar{\Omega}_i(q)/\partial\theta_d$ exists it must satisfy (2.45), we know that $\Sigma(q) = \partial\bar{\Omega}_i(q)/\partial\theta_d$. □

Also, note that the Lyapunov equation in (2.33) can be efficiently solved for low-dimensional systems using the algorithm proposed in (Barraud, 1977) and implemented in MATLAB function *dlyap*. We also highlight that, in order to compute the gradient, the partial derivatives of the steady state covariance matrices must be computed using the procedure in Prop. 3. Then, these partial derivatives are used along with (2.30) to compute the partial derivatives of the cost, which compose the gradient ∇J. Algorithm 1 summarizes the procedure to compute the steady state gradients.

In order to locally optimize the trajectories, the gradient computation needs to be used along with some gradient descent scheme. We describe the optimization procedure we used in Alg. 2, where κ_l is a scalar positive gain, the *proj* operator projects the parameters into the set of feasible parameters ($u_j(t) \in \mathcal{U}$) by minimizing the $\mathcal{L}$-2 norm. As a side note, this projection might be difficult to compute in general and, therefore when choosing a parameterization it is important to make sure that there are efficient ways to compute this projection numerically.

Algorithm 1 Steady State Gradient Computation

procedure ComputeSteadyStateGradient
  Input: Θ
  Compute $s_1(q), ..., s_N(q)$ from the parameterization
  for i ranging from 1 to M **do**
    Compute the steady state covariance $\bar{\Omega}_i(q)$
  Compute $\frac{\partial}{\partial\theta}\int_0^{t_f}\sum_{j=1}^{N} u_j'(\zeta)u_j(\zeta)d\zeta$ according to the parameterization
  Compute $\frac{\partial s_j(t)}{\partial\theta}$ and $\frac{\partial T}{\partial\theta}$ according to the parameterization
  for every θ in Θ **do**
    for i ranging from 1 to M **do**
      Compute $\frac{\partial\Omega_i(q)}{\partial\theta}$ as indicated in Prop. 3.
    Compute $\frac{\partial J}{\partial\theta}$ using (2.30)
  Output: ∇J

Algorithm 2 Gradient Descent

procedure Gradient Descent
  Input: Θ^0,
  $||\nabla J|| \leftarrow \infty$
  $l \leftarrow 0$
  while $||\nabla J|| > \epsilon$ **do**
    $\nabla J \leftarrow$ComputeGradient(Θ^l)
    $\Theta^{l+1} \leftarrow \text{proj}(\Theta^l - \kappa_l \nabla J)$
    $l \leftarrow l + 1$
  Output: $\underline{\Theta}^l$

2.5 Parameterization of an Optimal Trajectory in 1-D with speed bounds

When the agents and targets are constrained to a line, a particularly interesting case arises when the absolute value of controls is bounded ($\mathcal{U} = \{u \in \mathbb{R} \mid |u| < u_{\max}\}$) and there is no penalty for control effort in the optimization cost J (i.e. $\xi = 0$). The reason for considering this case is that we can represent optimized controls using a simple parameterization that could even lead to global optimality. It is worth noticing that in many real-world applications of persistent monitoring, agents are constrained to (possibly multiple) uni-dimensional mobility paths, such as powerline inspection agents, cars on streets, and autonomous vehicles in rivers.

Assuming proper rescaling, we can consider $-1 \leq u_j \leq 1$, i.e., $\mathcal{U} = [-1, 1]$. In the remainder of this section, we derive properties of the optimal control, establish a parameterization that is able to represent an optimal control, and then compute the gradients necessary in order to optimize the trajectories.

In order to derive the properties of an optimal control, we first introduce the following lemma. The intuition behind it is that if a target is observed for a longer time (or with better quality), its uncertainty will be lower. We note that, although this lemma is introduced in this Section, it is not restricted to the 1D setting with bounded input.

Lemma 1. *Given $\Omega_1(t)$ and $\Omega_2(t)$, two bounded covariance matrices under the dynamics in (2.20) with $A = A_1 = A_2$, $G = G_1 = G_2$, $Q = Q_1 = Q_2$, then if $\Omega_1(0) - \Omega_2(0)$ is negative semi-definite and $\eta_1(t) \geq \eta_2(t)$ $\forall t$, then $\Omega_1(t) - \Omega_2(t)$ is a negative semi definite matrix for all $t \geq 0$.*

Proof. Define $\beta = \Omega_1(t) - \Omega_2(t)$. The dynamics of β is described by the following equation.

$$\dot{\beta}(t) = A\beta(t) + \beta A' - \eta_1(t)\Omega_1(t)G\Omega_1(t) + \eta_2(t)\Omega_2(t)G\Omega_2(t). \tag{2.46}$$

Adding and subtracting the terms $\eta_1(t)\Omega_2(t)G\Omega_2(t)$ and $\eta_1(t)\Omega_1(t)G\Omega_2(t)$ to the equation, we can rewrite (2.46) as:

$$\dot{\beta}(t) = A\beta(t) + \beta A' - \eta_1(t)\left[\Omega_1(t)G\beta(t) + \beta(t)G\Omega_2(t)\right] + \left[\eta_2(t) - \eta_1(t)\right]\Omega_2(t)G\Omega_2(t). \quad (2.47)$$

From Thm. 1.e in (Kriegl et al., 2011), since $\beta(t)$ is a C^1 matrix, its eigenvalues can be C^1 time parameterized. Let μ_n denote the n^{th} eigenvalue of $\beta(t)$ and $x_n(t)$ the corresponding unit norm eigenvector. Then, from Thm. 5 in (Lancaster, 1964) we have that

$$\dot{\mu}_n = x_n'\dot{\beta}x_n.$$

Also, notice that by using (2.47) and the fact that $\lambda_{\min}\left(\frac{D+D'}{2}\right) \leq \frac{x'Dx}{\|x\|} \leq \lambda_{\max}\left(\frac{D+D'}{2}\right) = \|D\|$, for any square matrix D,

$$\begin{aligned}\dot{\mu}_n &\leq \|A\|\,\mu_n - \eta_1\beta\mu_n + \left[\eta_2 - \eta_1\right]x_n'\Omega_2G\Omega_2x_n \\ &\leq \|A\|\,\mu_n - \eta_1\beta\mu_n,\end{aligned}$$

where $\beta = \lambda_{min}\left((\Omega_1 + \Omega_2)G + G(\Omega_1 + \Omega_2)\right)$. Using Gronwall's inequality (Gronwall, 1919) and the fact that the solution of a first order linear homogeneous ODE does not change sign, we conclude that $\mu_n(t) \leq 0, \forall\ t \in [0, T]$ and, therefore, $\beta(t)$ is negative semidefinite. □

In Lemma 1, Ω_1 and Ω_2 can also be understood as covariance matrices for the same target but under different agent trajectories.

Before proceeding to the proposition about an optimal control structure, a few definitions are necessary. We define an *isolated target i* as a target such that

$$\min_{k\neq i}|x_i - x_k| > 2r_{\max}, \quad r_{\max} = \max_{i,j}\{r_{i,j}\}.$$

Therefore, an isolated target is a target for which an agent cannot see another target when visiting it. Referring to the regions in space where an agent can sense a target

as "visible area", the minimum distance between visible areas $d_{\min}$ is defined as:

$$d_{\min} = \min_{i,k} |x_i - x_k| - 2r_{\max} > 0,$$

We can then claim the following proposition.

Proposition 4. *In an environment where all the targets are isolated, given any policy $u_j(\xi)$, $j = 1, ..., N$, then there is a policy $\tilde{u}_j(\xi)$ with $\tilde{u}_j(\xi) \in \{-1, 0, 1\}$ $\forall \xi \in [0, t]$ and with the number of control switches for each agent (i.e. discontinuities in $\tilde{u}_j(\xi)$) upper bounded by $2\frac{t}{d_{\min}} + 4$ such that $J(u_1, ..., u_N, t) \geq J(\tilde{u}_1, ..., \tilde{u}_N, t)$.*

Proof. We prove this result by construction: given a policy $u_j(t')$ with $\eta_i(t')$ associated to it (as defined by (2.26)), we will construct an alternative policy $\tilde{u}_j(t')$ associated with $\tilde{\eta}_i(t')$ such that $\tilde{\eta}_i(t') \geq \eta_i(t')$ $\forall t' \in [0, t]$ and $i = 1, ..., M$, and then use Prop. 1, along with the definition of the cost (2.21), to show that the alternative policy has lower or equal cost than the original one.

Initially, we focus on the policy $u_j(t')$. We say that an agent j "visits" a target i if at some time t', $|s_j(t') - x_i(t')| < r_j$. For every agent in the policy $u_j(t')$, there is an ordered collection of targets it visits in $[0, t]$. Therefore, there must exist a set of indices of all the targets visited by agent j: $\{y_0^j, ..., y_{K_j}^j\} \in \{1, ..., M\}$, such that $y_p^j \neq y_{p-1}^j$ and agent j visited no other target in the time between visiting targets y_p^j and y_{p-1}^j. This is the sequence of all the targets that agent j visited over $[0, t]$, not considering consecutive visits to the same target. In other words, the same target can be present more than once in the sequence $\{y_0^j, ..., y_{K_j}^j\}$ but, if that is the case, it will not be in consecutive positions.

For each of these visits, we can define the initial visiting time t_p^j for $p = 1, ..., K_j$ as

$$t_p^j = \inf\{t' | t' > t_{p-1}^j \text{ and agent } j \text{ visits target } y_p^j \text{ at time } t'\},$$

and $t_0^j = 0$ and $t_{K_j+1}^j = t$. Also note that while $t' \in [t_{p-1}^j, t_p^j)$, agent j only influences the value of $\eta_i(j)$ of the target it is currently visiting. We propose the following

alternative policy, where $\tilde{u}_j(t')$ for $t' \in [t_{p-1}^j, t_p^j)$ is such that:

$$\tilde{u}_j(t') = \begin{cases} \frac{s_j(t_p^j) - s_j(t')}{|s_j(t_p^j) - s_j(t')|}, & \text{if } \frac{|s_j(t_p^j) - s_j(t')|}{t_p^j - t'} \leq 1, \\ \frac{x_{y_p^j} - s_j(t')}{|x_{y_p^j} - s_j(t')|}, & \text{if } \frac{|s_j(t_p^j) - s_j(t')|}{t_p^j - t'} > 1 \text{ and} \\ & \qquad s_j(t') \neq x_{y_p^j}. \\ 0, & \text{otherwise.} \end{cases}$$

Notice that this construction provides a feasible trajectory, since the original trajectory is assumed feasible. Also, in the alternative policy $\tilde{u}_j(t') \in \{-1, 0, 1\}\ \forall t' \in [0, t]$, since the speed is either zero or a scalar divided by its absolute value.

The intuition behind the proposed alternative policy is that at the beginning of each visit, the agent moves with maximum speed towards the target y_p^j and if it reaches the target, it dwells on top of it. However, it must move in a way such that it begins the next visit at the same time as in the original policy, i.e., the positions of agent j associated to the alternative policy $\tilde{s}_j(t')$ is such that $\tilde{s}_j(t_p^j) = s_j(t_p^j)$.

Also, for time $t' \in [t_p^j, t_{p+1}^j]$ both the original and the alternative policies only influence the value of η_i for $i = y_p^j$, since in the alternative policy the agent is closer (or at least as close) to the currently visited target. Thus, from (2.26) we have that

$$\tilde{\eta}_i(t') \geq \eta_i(t'),\ \forall t' \in [0, t],\ i \in \{1, ..., M\}.$$

Therefore, using Lemma 1 and the cost definition (2.21), we get that

$$J(\tilde{u}_1, ..., \tilde{u}_N, t) - J(u_1, ..., u_N, t) = \frac{1}{t} \int_0^t \sum_{i=1}^{M} \mathrm{tr}\Big(\tilde{\Omega}_i(t') - \Omega_i(t')\Big) \leq 0.$$

which shows that the alternative policy has a lower or equal cost compared to the original one. Note that, due to velocity constraints, in both the original and the alternative policy there is a maximum of $\frac{t}{d_{\min}} + 1$ visits to targets per agent. Moreover, in the alternative policy, an agent has at most 2 velocity switches at each target visit. Therefore, at most $2\frac{t}{d_{\min}} + 4$ velocity switches can happen due to target visits, plus one switch to match the initial position of the original policy and another to match the terminal position of the original policy. This implies that the maximum number of velocity switches in the alternative policy is $2\frac{t}{d_{\min}} + 4$. □

The result in Prop. 4 implies that when the targets are isolated, we can always get an optimized control such that $u_j(t) \in \{-1, 0, 1\},\ \forall t$, even if the trajectory is constrained to be periodic. This property allows the optimal trajectory to be described by a finite set of parameters. In this work in particular we are looking into periodic trajectories and, hence, this property implies that the movement in each period of agent j consists of a sequence of dwelling at the same position for some duration of time followed by moving at maximum speed to another location, until the agent returns to its initial position at that cycle. Therefore, one period of the trajectory of an agent j can fully be described by the following set of parameters:

1. T, the period of the trajectory.

2. $s_j(0)$, the initial position.

3. $\omega_{j,p},\ p = 1, ..., P_j$, the normalized dwelling times for agent j, i.e., the agent dwells for $\omega_{j,p}T$ units of time before it moves with maximum speed for the p-th time in the cycle.

4. $\tau_{j,p},\ p = 1, ..., P_j$, the normalized movement times for agent j, i.e., the agent j moves for $\tau_{j,p}T$ units of time to the right (if p is odd) or to the left (if p is even) after dwelling for $\omega_{j,p}T$ units of time in the same position.

To enforce non-negative movement and dwell times, we add the following constraints:

$$\tau_{j,m} \geq 0,\ \omega_{j,m} \geq 0,\ T \geq 0. \tag{2.48}$$

Notice that this description does not exclude transitions of u_j of the kind $\pm 1 \to \mp 1$ and $\pm 1 \to 0 \to \pm 1$, since it allows $\omega_{j,m} = 0$ and $\tau_{j,m} = 0$. In addition to the constraints in (2.48), in order to ensure periodicity, we need to make sure that the sum of the movement times and dwelling times does not exceed one period and that

the total time spent moving to the left is equal to the total time spent moving to the right over one period (i.e. the agent returns to its initial position at the end of the period). Therefore, we have the additional constraints:

$$\sum_{m=1}^{P_j}(\tau_{j,m}+\omega_{j,m}) \leq 1, \quad \sum_{m=1}^{P_j}(-1)^m \tau_{j,m} = 0. \tag{2.49}$$

This parameterization defines a hybrid system in which the dynamics of the agents remain unchanged between events and abruptly switch when an event occurs. Events are given by a change in control value at completion of movement and dwell times. Note that these may occur simultaneously, for instance, if the dwell time is zero (representing a switch of control from ± 1 to ∓ 1).

Given this parameterization, we use the procedure given in Alg. (1) to optimize the cost. However, one item missing for computing the gradient of the covariance matrix was the gradient of the agent position with respect to the parameters defining the trajectory($\frac{\partial s_j(q)}{\partial \theta}$), which will be given in the next subsection.

2.5.1 Computation of $\frac{\partial s_j(q)}{\partial \theta}$

The position of agent j at normalized time q, after the k-th event and before the $k+1$-th is

$$s_j(q) - s_j(0) = \begin{cases} T\Bigg((-1)^{k/2+1}\Bigg(q - \sum_{p=1}^{k/2-1}(\tau_{j,p}+\omega_{j,p}) \\ \quad + \omega_{j,\frac{k}{2}}\Bigg) + \sum_{p=1}^{k/2}(-1)^{p+1}\tau_p\Bigg), \ k \text{ even}, \\ T\sum_{p=1}^{\frac{k-1}{2}}(-1)^{p+1}\tau_{j,p}, \ k \text{ odd}. \end{cases} \tag{2.50}$$

Therefore,

$$\frac{\partial s_j}{\partial \tau_{j,m}} = \begin{cases} \left((-1)^{\frac{k}{2}+1} + (-1)^p\right)T, & m < \frac{k}{2}, \ k \text{ even}, \\ (-1)^{m+1}T, \ m \leq \frac{k-1}{2}, & k \text{ odd}, \end{cases} \tag{2.51}$$

$$\frac{\partial s_j}{\partial \omega_{j,m}} = \begin{cases} 1, \ m < \frac{k}{2}, & k \text{ even}, \\ 0 & , \text{ otherwise}, \end{cases} \tag{2.52}$$

$$\frac{\partial s_j(q)}{\partial T} = \frac{s_j(q) - s_j(0)}{T}, \tag{2.53}$$

$$\frac{\partial s_j}{\partial s_j(0)} = 1. \tag{2.54}$$

2.5.2 Initialization of the Optimization

While in Alg. 2 we use a gradient descent approach to locally minimize the cost function, it is necessary to find an initial parameter configuration. Therefore, we propose a method to efficiently compute a starting point for the optimization. Proposition 2 states that if every target is visited at least once in a periodic trajectory, then the steady-state covariance matrix exists. However, if in a periodic trajectory one of the targets is never visited and its internal state dynamics is unstable, then the estimation error will grow without bounds as time goes to infinity. Therefore, such initial trajectories are excluded. We now discuss a method for finding initial trajectories that will always lead to a feasible initial configuration. Note that due to the local nature of our optimization procedure, different initial conditions can lead to different local optima. We, therefore, leverage intuition to provide reasonable initial solutions with the hope that they will converge to good local optima.

The idea of finding a schedule where all the targets are visited fits naturally into a graph search paradigm, where the targets are modelled as nodes and the edge weights between nodes are the distances between the targets. The problem of finding a feasible schedule can be translated to the one of finding N sequences (that represent the schedule of each agent) of nodes where each target belongs to at least one of these

sequences. One can add to that a cost function that guides the way in which these sequences are created. A goal that intuitively will lead to reasonable initial solutions is to minimize the distance of the agent that has the longest travel path. This is the well known MTSP (see (Bektas, 2006) for a good overview of this problem and approaches to solve it). It is worth mentioning that the MTSP is NP-hard, and, therefore, intractable in the general setting. However, meta-heuristic approaches can provide feasible, though not necessarily optimal, solutions. In this work, we use the genetic algorithm described in (Tang et al., 2000) to find heuristic solutions. This approach is interesting because it finds a feasible solution in the first iteration and refines it as the number of iterations increases. Therefore, one can decide how much computation time to spend, leveraging the tradeoff between optimality and computation effort spent in generating this initial trajectory.

The MTSP problem finds a minimal length cycle and therefore can be immediately converted to parameters that represent one period of the steady state solution. We choose the dwelling times to be initially zero.

2.5.3 1D Simulation Results

In the simulations, we wanted to highlight one interesting aspects of the solution, rather than simply give an example of the techniques discussed in this subsection. We analyzed a steady state problem with 2 agents and 5 targets. We used the following matrices in the state evolution model

$$A_i = \begin{bmatrix} -1 & -0.1 \\ -0.1 & 0.01 \end{bmatrix}, \quad Q_i = \mathrm{diag}(1,1),$$

and the following parameters for the observation model

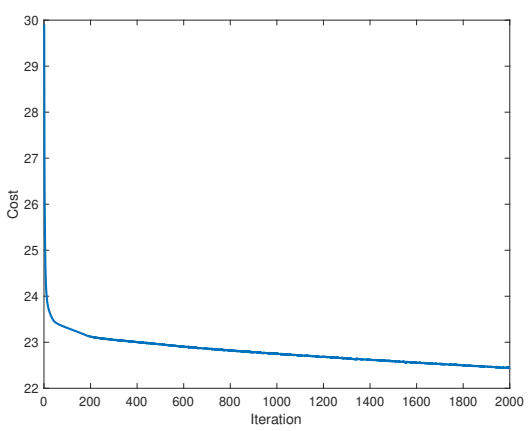
$$H_i = \mathrm{diag}(1,1), \quad R_i = \mathrm{diag}(1,1), \quad r_j = 0.9.$$

However, instead of using the initialization method proposed in this section, we used the following set of parameters:

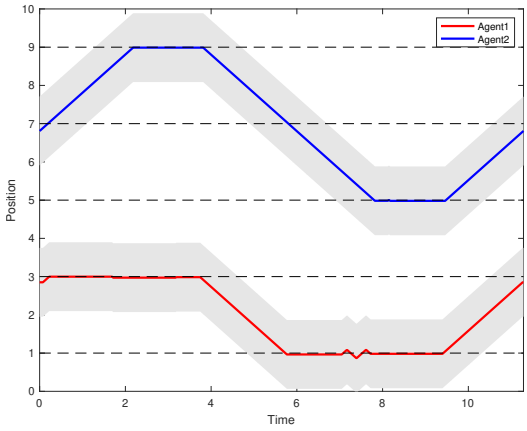
$$s_1^0(0) = 2.7, \quad s_2(0) = 6.8, \quad T^0 = 6, \tau_1^0 = \tau_2^0 = 0.01[10, 1, 10]^3,$$
$$\omega_1^0 = \omega_2^0 = 0.0125[1]^{11}.$$

The goal of using these initialization parameters was to have both agents sharing one target in the first iteration of the optimization process and then explore whether or not they would remain sharing the target after the local optimization procedure. The gradient descent step size was set to be constant, $\kappa_0 = \kappa_l = 0.02$.

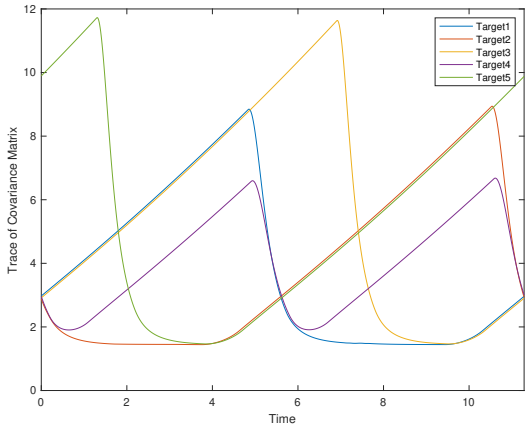
Figure 2·1 shows the results of the optimization in this scenario. Notice that even though both agents and all the targets have the same dynamical models, the solution at the last iteration of the optimization was such that one of the agents visits three of the targets and the other two of them. One interesting aspect of the trajectories of the targets in Fig. 2·1b is that, while in the period between times 6 and 8 agent 1 makes a movement with small amplitude around target 1, the effects of this oscillatory movement are hard to notice in the trace of the covariance of target 1 in Fig. 2·1c. Therefore, even though it is intuitively clear that staying still rather than moving with this oscillatory behavior will lead to a lower cost solution, the difference in terms of cost is minor. Also, notice that the solution has not yet fully converged, as can be seen in Fig. 2·1a; the simulation was terminated at this point due to computation time. Finally, we highlight that while the maximum number of switches in a direction allowed to each agent was set to 11, the final solution appears to have fewer because some of the movement and dwelling times in the final solution are essentially zero.



(a) Cost vs. iteration number



(b) Agent trajectories at final iteration



(c) Trace of the covariance for each target

Figure 2·1: Results of a simulation with two agents and five targets. (a) Evolution of the overall cost as a function of iteration number on the gradient descent. (b) Trajectories of the agents at the final iteration. The dashed lines indicate the positions of the targets and the grey shaded area the visibility region of the agent. (c) Evolution of the trace of the estimation covariance matrices of the five targets.

2.6 Fourier Curves for Multi-Dimensional Persistent Monitoring

For the 1D case we derived a parameterization with a finite number of parameters of the optimal solution. Unfortunately, the same result does not extend to the multi-dimensional persistent monitoring problems, since the multi-dimensional Hamiltonian analysis leads a much larger set of singular arcs, where the behavior of the agent is undefined in the general case. Therefore, instead of looking for an exact representation of the optimal trajectory, we focus on a family of parameterized curves that can approximate very general curves. While our approach is general, we pick as an illustration the case where speed is not bounded, since in that setting the projection operation in line 7 of Alg. 2 becomes trivial. Note that whenever the constant that weights the control effort penalization is not zero, i.e. $\xi \neq 0$ as defined in (2.29), the fact that the control effort is considered in the total cost will not allow the control to be unbounded. An appropriate choice of ξ can provide adequate speed bounds for any given dynamics of the system. As a side note, we highlight that bounded speeds can also be handled in this framework, however the projection operator in the gradient descent optimization becomes more complex. Thus, for the sake of simplicity, we only discuss the case without control bounds.

Since periodicity is an essential feature of the steady-state analysis discussed in this work, a natural choice is to use a truncated Fourier series to represent the movement of the agents in each of the coordinates e_p, $p = 1, ..., P$, i.e.

$$s_j^{e_p}(q) = s_{j,0}^{e_p} + \sum_{k=1}^{K} a_{j,k}^{e_p} \sin(2\pi f_k q) + b_{j,k}^{e_p}(\cos(2\pi f_k q) - 1), \tag{2.55}$$

where f_k are integer frequencies and, therefore, $s_j^{e_p}(q)$ is periodic with period 1. The set of parameters that fully characterize all the agents trajectories is $\Theta = \{\{a_{j,k}^{e_p}\}, \{b_{j,k}^{e_p}\}, \{s_{j,0}^{e_p}\}, T\}$, $j = 1, .., N$, $p = 1, ..., P$, $k = 1, ..., K$. As in the 1D case, in

order to compute the derivative of the covariance matrix, it is necessary to compute $\frac{\partial s_k}{\partial \theta}$. Note that

$$\frac{ds_j}{dq} = T\frac{ds_j}{dt}. \tag{2.56}$$

Using (2.55), we can compute

$$\sum_{j=1}^{N}\int_0^1 \left\|\frac{ds_j}{dt}\right\|^2 dq = \sum_{j=1}^{N}\sum_{p=1}^{P}\sum_{k=1}^{K}\frac{(2\pi f_k)^2}{2T^2}\left(\left(a_{j,k}^{e_p}\right)^2 + \left(b_{j,k}^{e_p}\right)^2\right), \tag{2.57}$$

and, therefore,

$$\frac{\partial}{\partial a_{j,k}^{e_p}}\sum_{j=1}^{N}\int_0^1 \left\|\frac{ds_j}{dt}\right\|^2 dq = \frac{(2\pi f_k)^2}{2T^2}a_{j,k}^{e_p}, \tag{2.58a}$$

$$\frac{\partial}{\partial b_{j,k}^{e_p}}\sum_{j=1}^{N}\int_0^1 \left\|\frac{ds_j}{dt}\right\|^2 dq = \frac{(2\pi f_k)^2}{2T^2}b_{j,k}^{e_p}, \tag{2.58b}$$

$$\frac{\partial}{\partial s_{j,0}^{e_p}}\sum_{j=1}^{N}\int_0^1 \left\|\frac{ds_j}{dt}\right\|^2 dq = 0, \tag{2.58c}$$

$$\frac{\partial}{\partial T}\sum_{j=1}^{N}\int_0^1 \left\|\frac{ds_j}{dt}\right\|^2 dq = \sum_{j=1}^{N}\sum_{p=1}^{P}\sum_{k=1}^{K}\frac{-(2\pi f_k)^2}{T^3}\left(\left(a_{j,k}^{e_p}\right)^2 + \left(b_{j,k}^{e_p}\right)^2\right). \tag{2.58d}$$

2.6.1 Initialization

In the multi-dimensinal optimization, we still use the suboptimal solution of the MTSP problem as a starting point. However, unlike the 1-D scenario with the movement and dwelling time parameterization, the heuristic solution of the MTSP problem cannot be directly converted to a Fourier Curve trajectory. The solution of the MTSP problem gives, for each agent j, a cyclic schedule of targets $\mathcal{S}_j = \{y_j^1, ..., y_j^{Y_j}, y_j^1\}$ and, therefore, it is still necessary to obtain the parameters $\Theta = \{\{a_{j,k}^{e_p}\}, \{b_{j,k}^{e_p}\}, \{s_{j,0}^{e_p}\}, T\}$ from this schedule. We define d_j^m as the cumulative distance that the agent has traveled when it reaches the m-th target in the schedule $\mathcal{S}_j$, and D_j as the total distance

traveled by an agent in one cycle. We then look for a feasible truncated Fourier series trajectory such that at the normalized time $q = d_j^m/(D_j T)$, the agent is at a distance lower or equal to the sensing radius (multiplied by a factor $1-\delta$, $0 < \delta < 1$, in order to give some distance margin) from the target. The position of the agent at the beginning of the cycle is set to be the position of the first target in the schedule $\mathcal{S}_j$.The period T can be set to any positive number. For each agent, the following optimization problem gives a set of feasible $\{a_{j,k}^{e_p}\}, \{b_{j,k}^{e_p}\}$.

$$\begin{aligned} \underset{a_{j,k}^{e_p},\, b_{j,k}^{e_p}}{\text{minimize}} \quad & \sum_{p=1}^{P}\sum_{k=1}^{K} f_k|a_{j,k}^{e_p}| + f_k|b_{j,k}^{e_p}| \\ \text{subject to} \quad & \left\| s_j\left(\frac{d_j^m}{D_j}\right) - x_{y_j^m} \right\|_2 < (1-\delta)r_j,\ m = 1,..,Y_j. \end{aligned} \tag{2.59}$$

Note that if we substitute the definition in (2.55) into the constraint in (2.59), this optimization can be formulated as a Quadratically Constrained Program, which is a convex optimization problem and thus can be solved efficiently. From our experience, minimizing a weighted sum of absolute values in the objective function of (2.59) has led to smooth initial trajectories. However, other optimization objectives could be used.

It is worth observing that for each of the agents, the trajectory generated by the heuristic solution of the MTSP problem consists of segments of straight lines that visit each of the targets in the schedule $\mathcal{S}_j$. Note that this trajectory, as a function of time, is composed by sequence of straight lines that can be projected in each of the axis e_p and the projection in that axis will still be a sequence of segments of straight lines. Since piecewise linear functions can be well approximated by Fourier series, there always exist a K large enough such that there is a solution to (2.59) because for that K there is a representation of the trajectory that would be close enough to the original MTSP solution such that it is able to satisfy the constraint in (2.59).

Therefore, we can always find feasible solutions to (2.59) if we have a MTSP solution.

2.6.2 2D and 3D Simulation Results

We demonstrate the results of the algorithm in a simulated 2D scenarios, with three agents and 15 targets. All the internal states of the targets have the same state dynamics, evolving according to (2.1) with

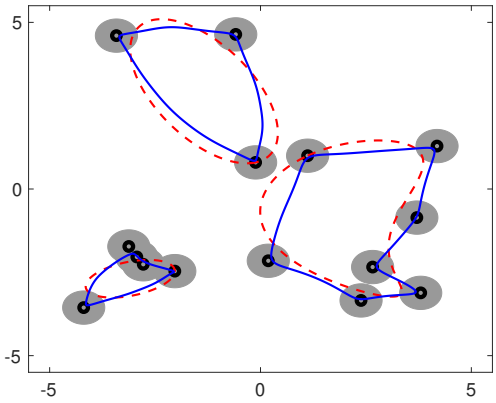
$$A_i = \begin{bmatrix} -1 & -0.1 \\ -0.1 & 0.01 \end{bmatrix}, \quad Q_i = \text{diag}(1, 1),$$

and the agents observation models are given by (2.3) with

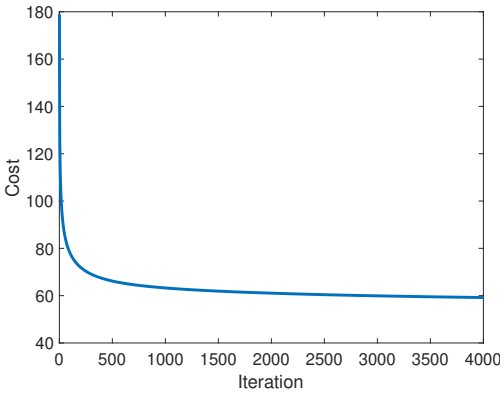
$$H_i = R_i = \text{diag}(1, 1), \quad r_j = 0.5, \quad \eta = 10^{-3}.$$

For each of the agents, their trajectories had the first five harmonics in each axis, i.e., $f_k = k$, $k = 1, ..., 5$, $\forall\ j$. In the initial step of the optimization, the period T was set to 1. The initial coefficients $a_{j,k}^{e_p}, b_{j,k}^{e_p}$ were obtained by solving the optimization problem in (2.59). The MTSP solution was obtained after 3000 iterations of the genetic algorithm proposed in (Tang et al., 2000) for solving the associated MTSP. The initial position of each agent was set to coincide with the position of the first target in the solution of the MTSP. A constant descent stepsize $\kappa_l = 10^{-4}$ was used in the gradient descent.

The positions of the targets were generated randomly from independent uniform distributions ranging from -5 to 5 in both axis. Fig. 2·2a compares the trajectories of the agents in the first and last step of the gradient descent optimization, while Fig. 2·2b shows the evolution of the cost as a function of the gradient descent step. The results of the optimization show that the solution of (2.59) led to smoother trajectories that still visited all the targets. The gradient descent changed the trajectories geometry but did not change the visiting order. As can be observed in Fig. 2·2b, the



(a) Trajectories of the targets in the first (red dashed line) and last (blue continuous line) iterations of the gradient descent optimization on the scenario with 15 targets and 3 agents. The target's locations are marked in black and the grey shaded are represent the regions where the target can be sensed by an agent.



(b) Evolution of the cost function in the gradient descent optimization in the scenario with 15 targets and 3 agents.

cost has an abrupt reduction in the beginning of the optimization and then the convergence speed reduces significantly. The optimization process lead to very significant reductions of the cost, reducing it to less than one third of its initial value.

In order to illustrate the extension of techniques proposed in this paper to higher dimensions, we present a result in a 3D environment, with 2 agents and 10 targets. The A_i, Q_i, H_i, R_i matrices and r_j are the same as in the 2D simulations. A constant gradient descent stepsize $\kappa_l = 10^{-2}$ was used. The target locations were drawn from a uniform distribution in the cube with coordinates ranging from $[-5, 5]$ in each axis. The trajectories after 4000 gradient descent iterations are shown in Fig. 2·3 and the evolution of the cost is diplayed in Fig. 2·4.

The 3D results follow a very similar trend of the 2D ones. The trajectories provided by the initialization procedure tend to be smoother, while the shape of the optimized ones are stiffer.

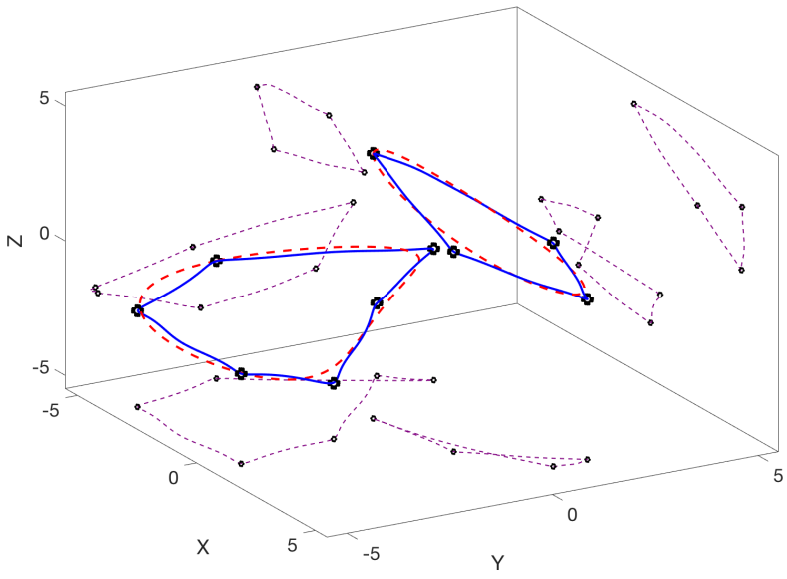


Figure 2·3: Simulation results in a 3D environment with two targets and ten agents. In red, the initial trajectory in the gradient descent optimization, in blue, the trajectory at the end of the optimization. The projection of the final agent trajectories in three planes is plotted in dashed purple.

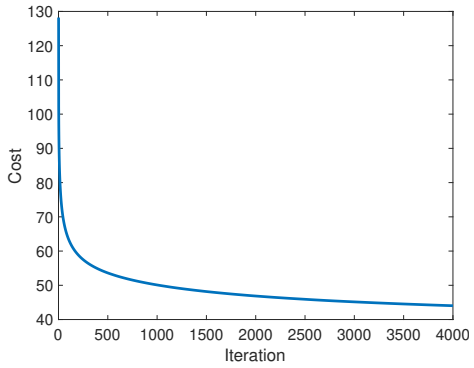


Figure 2·4: Evolution of the cost function in the gradient descent optimization in the 3D scenario.

Chapter 3

Minimax Persistent Monitoring Embedded on a Graph

In the previous section, we presented a gradient-based approach for solving the PM problem. Although the computational effort for computing gradients scales well (linearly) with the number of agents and targets, it requires the solution of $N \times M \times P$ matrix differential equations at each step of the iterative gradient-descent process, where N is the number of targets, M is the number of agents and P the number of parameters that define an agent trajectory. Therefore, the usage of gradient-based methods are impractical for settings with large number of agents and targets.

In this chapter, we aim to introduce lightweight algorithms that can efficiently address the PM problem. Towards that goal, we make some simplifications to the problem formulation that will prove helpful in the development of a more computationally effective algorithm. Namely, the modifications in the problem formulation are: instead of minimizing the $\mathcal{L}$-2 norm of the covariance, we consider the $\mathcal{L}$-∞ norm. This cost function is also an appropriate choice in many PM applications. For example, when monitoring safety-critical systems that cannot operate over a given threshold (for instance, a maximum temperature), it is natural to optimize the "worst-case" performance (as opposed to optimizing an "average" chance of violating it). Some examples of applications where a critical threshold on the state uncertainty should not be exceeded include monitoring wildfire or faults in civil infrastructure systems using unmanned aerial vehicles (Lin et al., 2018; Shakhatreh et al., 2019).

Moreover, we assume that each target cannot be observed by multiple agents; the agent can no longer move freely, but is constrained to a graph-based structure; and the agent can only observe the target when their position coincide.

With this new problem formulation, we are able to show that, for an agent trajectory, the peak of the norm of the steady state covariance matrix should be the same among all the targets. This allows us to develop an algorithm such that instead of explictly minimizing the cost function, we instead try to calibrate dwelling times in order to have this property hold. In some cases, this property alone is sufficient to uniquely define the agent trajectory and, under certain conditions, leads to global optimality.

It is important to note, however, that the algorithms described in this chapter are not necessarily intended to replace the one given in the previous one. Rather, when the goal is to minimize the $\mathcal{L}$-2 norm, the trajectories given by this approach can be used as efficient initial solutions in the gradient-based approach, even if their formulations are not completely equivalent to each other. As a side note, since in this chapter we consider that each target is observed by only one agent, and mostly approach the problem from a single agent perspective, in order to simplify the notation, we omit the index j from our variables. Later in the chapter we extend this to multi-agent systems by using a divide-and-conquer strategy. The results in this chapter, especially regarding the greedy exploration of cycles, were developed in close collaboration with Shirantha Welikala (PhD, Sytems Engineering, Boston University, 2021).

3.1 Infinity norm cost function

In this section, we assume that the movement of the agent is constrained to a graph-based structure. The environment to be surveilled is described by an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where the set $\mathcal{V} = \{1, 2, \ldots, M\}$ represents M nodes (targets) and

the set $\mathcal{E} = \{(i,j) : i,j \in \mathcal{V}\}$ represents all the edges (available for agents to travel between targets). Each edge $(i,j) \in \mathcal{E}$ has an associated value $d_{i,j}$ that represents the travel-time an agent has to spend in order to travel from target i to target j. Each target has an internal state that evolves according to the same linear dynamics corrupted by additive noise described in the previous chapter.

Similarly to the previous chapter, we also consider here parametric policies. The graph-based nature of the current model yields a natural parameterization: the agent trajectory is described by its sequence of target visits (visiting sequence): $\Xi = [\xi_1, \xi_2, ..., \xi_N]$ where each $\xi_k \in \mathcal{V}$ and the corresponding sequence of dwell-times (dwelling sequence) spent at each visited target: $\mathcal{T} = [t_1, ..., t_N]$ where each $t_k \geq 0$. Moreover, here we assume that the internal states are observed only when target and agent positions coincide, i.e., instead of the expression (2.5), we use the following definition for $\gamma_{i,j}$:

$$\gamma_{i,j}(\alpha) = \begin{cases} 1, & \|\alpha\| = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

Our goal is to design an agent trajectory (i.e., to design Ξ and $\mathcal{T}$) that minimizes the persistent monitoring objective

$$J(\Xi, \mathcal{T}) = \max_{i \in \mathcal{V}} \limsup_{t \to \infty} g_i(\|\Omega_i(t)\|), \tag{3.2}$$

which represents the maximum over time and over all the targets of a weighted norm of the long term covariance matrix. In (3.2), the target-specific possibly non-linear weighting function $g_i(\cdot)$ is strictly increasing with $g_i(0) = 0$ and $\lim_{x \to \infty} g_i(x) = \infty$ and $\|\cdot\|$ is a norm on the space of positive semi-definite matrices. A usual choice is to have $g_i(x) = \alpha_i x$, where α_i is a constant target-specific weight, and $\|X\| = \text{tr}(X)$. If an optimal agent trajectory $(\Xi^*, \mathcal{T}^*)$ exists, then we denote its associated cost

$J^*(\Xi^*, \mathcal{T}^*) = \min_{\Xi,\mathcal{T}} J(\Xi, \mathcal{T})$.

As in the previous chapter, here we consider the steady-state version of the PM problem and we constrain the agent trajectory to be cyclic. Therefore, as long as each target with unstable dynamics is visited for any finite amount of time (see Prop. 2), we can guarantee that its covariance matrix will converge to a limit cycle that does not depend on the initial conditions. Moreover, this proposition also implies that the lim sup in (3.2) can be replaced by the maximum over one period, i.e.

$$\limsup_{t\to\infty} g_i(\|\Omega_i(t)\|) = \max_{t\in[0,T]} g_i(\left\|\bar{\Omega}_i(t)\right\|). \tag{3.3}$$

3.2 Properties of an Optimal Policy

3.2.1 Target's Perspective of a Periodic Policy

We begin by defining some notations used in the remainder of this section. Recall that the goal is to optimize the visiting sequence Ξ and the corresponding dwelling sequence $\mathcal{T}$. Also, recall that the common length N_Ξ of the vectors Ξ and $\mathcal{T}$ correspond to one full cycle of the periodic trajectory.

Recall that the goal is to optimize the visiting sequence Ξ and the corresponding dwelling sequence $\mathcal{T}$. We now discuss the conversion of indices from the agent's perspective (i.e., Ξ and $\mathcal{T}$) to the target's perspective. First, for each target i, we group all the instances where this target was visited and define the vector $\mathcal{P}_i = [p_i^1, ..., p_i^{N_i}]$ where p_i^j is the position in the visiting sequence Ξ that i is visited for the j^{th} time. Consider, for example, a visiting sequence $\Xi = [1, 2, 1]$. Then, $N_1 = 2$, $N_2 = 1$ and $\mathcal{P}_1 = [p_1^1, p_1^2] = [1, 3]$, $\mathcal{P}_2 = [p_2^1] = [2]$.

Moreover, we define the tuple $(a(q), b(q))$ as the pair such that $p_{a(q)}^{b(q)} = q$. Hence, $a(q)$ is the target being visited at the agent's q^{th} visit and $b(q)$ represents the number of times this target has been visited so far (including the current visit).

Finally, we highlight some important timings and covariance matrix values at the

steady state covariance profile. Figure 3·1 illustrates all the definitions that we will give. We start with $t^k_{\text{on},i}$ and $t^k_{\text{off},i}$ (defined in Section 3.2.1) which are given by

$$t^k_{\text{on},i} = t_{p^k_i}, \tag{3.4a}$$

$$t^k_{\text{off},i} = d_{i,a(p^k_i+1)} + \sum_{q=p^k_i+1}^{p^{k+1}_i-1} \left(t^{b(q)}_{\text{on},a(q)} + d_{a(q),a(q+1)} \right). \tag{3.4b}$$

In (3.4b), $d_{i,a(c(k,i)+1)}$ is the travel time between target i and the next target the agent visits. The index q varies over all the visits the agent makes until it returns to target i (note that $c(k,i)$ and $c(k+1,i)$ give the index of two consecutive visits to target i, from the agent's perspective). Moreover, $t^{b(q)}_{\text{on},a(q)}$ is the time the agent spent at its q-th visit and $d_{a(q),a(q+1)}$ is the travel time between the agent's q^{th} and $(q+1)^{\text{th}}$ visit.

Additionally, we define $\overline{\tau}^k_i$ as the instant when the k^{th} visit to target i started and $\overline{P}^k_i$ is the covariance at the beginning of this visit. Intuitively, variables with a bar over them refer to a local maximum peak ($\overline{P}^k_i$) and instant ($\overline{\tau}^k_i$). Similarly, $\underline{\tau}^k_i$ and $\underline{P}^k_i$ are respectively the time instant and the covariance at the end of the k^{th} visit and represent locally minimum peaks, as represented in Fig. 3·1. More formally:

$$\overline{\tau}^k_i = \sum_{m=0}^{k-1} \left(t^m_{\text{on},i} + t^m_{\text{off},i} \right), \qquad \underline{\tau}^k_i = \overline{\tau}^k_i + t^k_{\text{on},i}, \tag{3.5a}$$

$$\overline{P}^k_i = \bar{\Omega}(\overline{\tau}^k_i), \qquad \underline{P}^k_i = \bar{\Omega}(\underline{\tau}^k_i). \tag{3.5b}$$

Also, since both the agent trajectories and the steady state covariance are periodic, we have that $t^k_{\text{on},i} = t^{k+N_i}_{\text{on},i}$, $t^k_{\text{off},i} = t^{k+N_i}_{\text{on},i}$, $\overline{P}^k_i = \overline{P}^{k+N_i}_i$ and $\underline{P}^k_i = \underline{P}^{k+N_i}_i$. On the other hand, visiting instants are not periodic, but are spaced by T, hence $\overline{\tau}^{k+N_i}_i = T + \overline{\tau}^k_i$ and $\underline{\tau}^{k+N_i}_i = T + \underline{\tau}^k_i$.

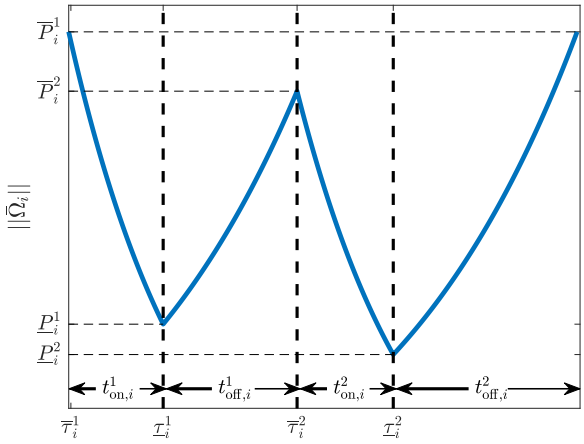


Figure 3·1: Temporal evolution of the steady state covariance matrix and waiting/observation times.

3.2.2 Necessary Condition for Optimality

In this section, our main goal is to show that, for any visiting sequence Ξ that contains every target in $\mathcal{V}$, the corresponding optimal dwelling sequence $\mathcal{T}$ must be such that $\limsup_{t\to\infty} g_i(\|\Omega_i(t)\|)$ is the same for all $i \in \mathcal{V}$.

Towards this goal, we first introduce an auxiliary result that states that the steady-state covariance increases when the target is not observed ($\eta_i = 0$) and decreases otherwise ($\eta_i = 1$). However, we stress that this only holds at steady state. If, for example, the initial uncertainty over target i is very small, then the transient uncertainty could temporarily grow even if i is being observed. Similarly, the uncertainty along certain directions may initially decrease even when i is not observed if it was very large at the initial time.

Lemma 2. *If $t^k_{off,i} > 0$ and $t^p_{on,i} > 0$ for some p, k such that $1 \le p, k \le N_i$, then $\dot{\bar{\Omega}}_i(t) \prec 0$ when the target is observed ($\eta_i(t) = 1$) and $\dot{\bar{\Omega}}_i(t) \succ 0$ otherwise (i.e. when $\eta_i(t) = 0$).*

Proof. Let $\Omega_{i,ss}$ be the solution of the algebraic Riccati equation given by (2.20), when $\eta_i(t) = 1$ and $\dot{\Omega}_i = 0$, i.e.

$$A_i\Omega_{i,ss} + \Omega_{i,ss}A_i' + Q_i - \Omega_{i,ss}G\Omega_{i,ss} = 0. \tag{3.6}$$

Since the system is observable and $Q_i \succ 0$, $\Omega_{i,ss}$ is guaranteed to be unique and positive definite (Bittanti et al., 2012).

We then define a similar concept to $\Omega_{i,ss}$, but for the case where $\eta_i(t) = 0$, $\forall t$. Intuitively, we want to define Ω_i^∞ as being the covariance matrix when the target is never observed. Note that when A_i is unstable, the covariance matrix will diverge thus Ω_i^∞ is not well defined. In particular, if A_i is unstable but some of the eigenvalues of A_i are negative, the covariance does not diverge in every direction. Thus, in order to overcome these peculiarities in "Ω_i^∞", for a given vector $\zeta \in \mathbb{R}^{L_i}$ (i.e. ζ has the same dimension as the the target state), we define $\zeta'\Omega_i^\infty\zeta$ as:

$$\zeta'\Omega_i^\infty\zeta = \lim_{t\to\infty} \zeta' \left(\int_0^t \exp(A_i\gamma) Q_i \exp(A_i'\gamma) d\gamma \right) \zeta. \tag{3.7}$$

Note that, if $\eta_i(t) = 0$, $\lim_{t\to\infty} \zeta'\Omega_i(t)\zeta = \zeta'\Omega_i^\infty\zeta$, independently of the initial condition $\Omega_i(0)$ (Bittanti et al., 2012).

Using Theorem 2 in (Dieci and Eirola, 1996), we get that the steady state covariance matrix $\bar{\Omega}_i(t)$ generated by a cyclic schedule where target i observed by some time part of the cycle (but not the entire cycle) is such that:

$$\zeta'\Omega_{i,ss}\zeta < \zeta'\bar{\Omega}_i(t)\zeta < \zeta'\Omega_i^\infty\zeta, \tag{3.8}$$

for $\forall \zeta \neq 0 \in \mathbb{R}^{L_i}$ such that $\zeta'\Omega_i^\infty\zeta$ is bounded. Informally, one can think of $\Omega_{i,ss}$ and Ω_i^∞ as being respectively lower and upper bounds on the covariance matrix.

Now that we have defined these lower and upper bounds, we go back to analyze the steady state covariance matrix resulting from an agent trajectory that visits target i. First we show that, at steady state, the covariance will increase when the target is not observed. Towards that, we analyze dynamics of the steady state covariance matrix $\dot{\bar{\Omega}}_i$ in time instants where the target is not observed ($\eta_i = 0$), thus $\dot{\bar{\Omega}}_i = A_i\bar{\Omega}_i + \bar{\Omega}_i A_i' + Q_i$. If we pick ζ to be a right eigenvector of $(A_i + A_i')$ and its corresponding eigenvalue λ, we get that

$$(A_i + A_i')\zeta = \lambda\zeta \implies A_i'\zeta = \lambda\zeta - A_i\zeta. \tag{3.9}$$

Analogously, we have that $\zeta'A_i = \lambda\zeta' + \zeta'A_i'$. Using these relations, we get that $\zeta'(A_i\bar{\Omega}_i + \bar{\Omega}_i A_i')\zeta$ is given by

$$\zeta'(A_i\bar{\Omega}_i + \bar{\Omega}_i A_i')\zeta = 2\lambda\zeta'\bar{\Omega}_i\eta - \left(\zeta'A_i'\bar{\Omega}_i\zeta + \zeta'\bar{\Omega}_i A_i\zeta\right). \tag{3.10}$$

Since $\zeta' A_i' \bar{\Omega}_i \zeta + \zeta' \bar{\Omega}_i A_i \zeta$ is a scalar, it is equal to its transpose, $\zeta'(A_i \bar{\Omega}_i + \bar{\Omega}_i A_i')\zeta$. Therefore, we conclude that

$$\zeta'(A_i \bar{\Omega}_i + \bar{\Omega}_i A_i')\zeta = 2\lambda \zeta' \bar{\Omega}_i \zeta. \tag{3.11}$$

Using this result, we compute $\zeta' \dot{\bar{\Omega}}_i \zeta$, which is given by:

$$\zeta' \dot{\bar{\Omega}}_i \zeta = \zeta'(A_i \bar{\Omega}_i + \bar{\Omega}_i A_i' + Q_i)\zeta = \zeta' \left(\lambda \bar{\Omega}_i + Q_i\right) \zeta. \tag{3.12}$$

Note that, if λ is non-negative, then the expression (3.12) is necessarily positive, thus $\zeta' \dot{\bar{\Omega}}_i(t) \zeta > 0$. However, if λ is negative, then (3.7) implies that $\zeta' \Omega_i^\infty \zeta$ is bounded and and that its time derivative, $\lim_{t \to \infty} \zeta'(A_i \Omega + \Omega A_i + Q)\zeta$, converges to zero, for any initial condition $\Omega(0)$. Since $\zeta'(A_i \Omega + \Omega A_i)\zeta = \lambda \Omega$, we get that $\lambda \zeta' \Omega_i^\infty \zeta + \zeta' Q_i \zeta = 0$. Since here we consider the case where $\lambda < 0$ and we know that $\bar{\Omega}_i \prec \Omega_i^\infty$, we conclude that $\lambda \zeta' \bar{\Omega}_i \zeta + \zeta' Q_i \zeta > \lambda \zeta' \Omega_i^\infty \zeta + \zeta' Q_i \zeta = 0$. Thus, we claim that $\zeta' \dot{\bar{\Omega}}_i \zeta > 0$ whenever $\eta_i = 0$. Since the eigenvectors of $A_i + A_i'$ are a basis of $\mathbb{R}^{L_i}$, we have that any $\xi \in \mathbb{R}^{L_i}$ can be written as a linear combination of these eigenvectors. Therefore, $\chi' \bar{\Omega}_i \chi$ is positive for any $\chi \in \mathbb{R}^{L_i}$ whenever $\eta_i = 0$, which implies that $\bar{\Omega}_i(t) \succ 0$ if $\eta_i(t) = 0$.

We then consider the instants when the target is observed ($\eta_i = 1$). Note that $\dot{\Omega}_{i,ss} = 0$. Then we define $U = \bar{\Omega}_i - \Omega_{i,ss}$, for which $\dot{\bar{\Omega}}_i = \dot{U}$. Using (2.20) and (3.6), we get that

$$\begin{aligned} \dot{U} &= A_i(\bar{\Omega}_i - \Omega_{i,ss}) + (\bar{\Omega}_i - \Omega_{i,ss})A_i' - \bar{\Omega}_i G_i \bar{\Omega}_i \\ &\qquad + \Omega_{i,ss} G_i \Omega_{i,ss} \\ &= A_i U + U A_i' + \Omega_{i,ss} G_i \Omega_{i,ss} - \bar{\Omega}_i G_i \bar{\Omega}_i \\ &= (A_i - G_i \Omega_{i,ss})U + U(A_i - G_i \Omega_{i,ss})' - U G_i U. \end{aligned} \tag{3.13}$$

Additionally, using the fact that $\dot{U}^{-1} = U^{-1} \dot{U} U^{-1}$, we have

$$\dot{U}^{-1} = U^{-1}(A_i - \Omega_{i,ss} G_i) + (A_i - \Omega_{i,ss} G_i)' U^{-1} - G_i.$$

We now consider a right eigenvector ζ of $((A_i - \Omega_{i,ss} G_i) + (A_i - \Omega_{i,ss} G_i)')$ and its corresponding eigenvalue λ. Note that the algebraic Riccati equation (3.6) can be rewritten as:

$$(A_i - \Omega_{i,ss} G_i)\Omega_{i,ss} + \Omega_{i,ss}(A_i - \Omega_{i,ss} G_i)' + Q_i + \Omega_{i,ss} G_i \Omega_{i,ss} = 0. \tag{3.14}$$

Multiplying this equation by ζ on the right and ζ' on the left, we get

$$\lambda\zeta'\Omega_{i,ss}\zeta + \zeta'(Q_i + \Omega_{i,ss}G_i\Omega_{i,ss})\zeta = 0, \tag{3.15}$$

which implies that $\lambda < 0$, since $\zeta'\Omega_{i,ss}\zeta$ and $\zeta'(Q_i+\Omega_{i,ss}G_i\Omega_{i,ss})\zeta$ are strictly positive. Then, computing $\zeta^*\dot{U}^{-1}\zeta$, we get

$$\zeta'\dot{U}^{-1}\zeta = \lambda\zeta'U^{-1}\zeta - \zeta'G_i\zeta,$$

thus $\zeta'\dot{U}^{-1}\zeta < 0$, since $U = \bar{\Omega}_i - \Omega_{i,ss} \succ 0$ and $\lambda < 0$. Since all the eigenvectors of $((A_i - \Omega_{i,ss}G_i) + (A_i - \Omega_{i,ss}G_i)')$ together form a basis of $\mathbb{R}^{L_i}$, we have that $\dot{U}^{-1}$ is negative definite and thus $\dot{\bar{\Omega}}_i = \dot{U}$ is also negative definite, which concludes the proof of the lemma. □

The intuitive explanation of the importance of this Lemma is that it gives an insight on how to optimize the dwelling times: when local maximum uncertainty peaks are different, it is possible to observe for less time the target with the lowest peaks (increasing its uncertainty) and for more time the targets with largest peak (thus decreasing it). This way, maximum uncertainty over all the targets would go down.

We now show that the peak uncertainties can only be achieved at very specific instants of time: those where the target switches from not being observed to being observed. This is an important result, since it guarantees that we can compute the cost function by only looking at a finite number of time instants. As a reminder, we define the covariance at the beginning of the k^{th} observation of that target as $\overline{P}_i^k$.

Lemma 3. *If target i is visited for a strictly positive amount of time, then* $\limsup_{t\to\infty} g_i(\|\Omega_i(t)\|) = \max_{1\le k\le N_i} g_i(\|\overline{P}_i^k\|)$.

Proof. First, note that since $\Omega_i(t)$ converges to the bounded periodic function $\bar{\Omega}_i(t)$ and $g_i(\cdot)$ is continuous, $\limsup_{t\to\infty} g_i\left(\|\Omega_i(t)\|\right) = \max_{0\le t\le T} g_i\left(\|\bar{\Omega}_i(t)\|\right)$. For any time t for which $\exists\ \epsilon > 0$ such that $\eta_i(t+\epsilon) = 0$, Lemma 2 implies that $\bar{\Omega}_i(t+\epsilon) \succ \bar{\Omega}_i(t)$. Conversely, if $\exists\ \epsilon > 0\ \eta_i(t-\epsilon) = 1$, $\bar{\Omega}_i(t-\epsilon) \succ \bar{\Omega}_i(t)$.

Therefore, the maximization of $g_i\left(\left\|\bar{\Omega}_i\right\|\right)$ can only happen in one of the instants when the target switches from not being observed ($\eta_i = 0$) to being observed ($\eta_i = 1$). The covariance at these instants is given by $\overline{P}_i^k$ (see also Fig. 3·1). □

Using Lemmas 2, 3, we next show how the peak values $\overline{P}_i^k$ vary with $t_{\text{on},i}^m$ and $t_{\text{off},i}^m$, for any k, m where $1 \leq k, m \leq N_i$. For this, we use the fact that the parameters $t_{\text{on},i}^m$ and $t_{\text{off},i}^m, \forall m$ fully define (up to a time-shift) the evolution of the steady state covariance matrix $\bar{\Omega}_i(t)$. The following proposition has an intuitive interpretation: when a target is observed for a longer time, its peak uncertainty will be lower. Conversely, if the time between observations increases, then the peak uncertainty will be higher.

Proposition 5. $\frac{\partial \overline{P}_i^k}{\partial t_{on,i}^m} \prec 0$ *and* $\frac{\partial \overline{P}_i^k}{\partial t_{off,i}^m} \succ 0$.

Proof. First, we prove that $\frac{\partial \overline{P}_i^k}{\partial t_{\text{on},i}^m} \prec 0$. Towards that, we define Φ_i^m as

$$\Phi_i^m = \int_0^{t_{\text{on},i}^m} \exp\left(A_i - \bar{\Omega}_i(t + \overline{\tau}_i^m) G_i\right) dt. \tag{3.16}$$

To understand the intuition behind Φ_i^m, we first recall Prop. 3 which states that the derivative $\frac{\partial \bar{\Omega}}{\partial t_{\text{on},i}^m}(t)$, when it exists, for $t \in (\overline{\tau}_i^m, \underline{\tau}_i^m)$ is given by:

$$\frac{\partial \bar{\Omega}}{\partial t_{\text{on},i}^m}(t) = \Sigma^T(t) \frac{\partial \bar{\Omega}}{\partial t_{\text{on},i}^m}(\overline{\tau}_i^m) \Sigma(t), \tag{3.17}$$

where Σ is the solution of the following ODE:

$$\dot{\Sigma}(t) - (A - \bar{\Omega}_i(t) G_i)\Sigma(t) = 0,\ \Sigma(\underline{\tau}_i^m) = I. \tag{3.18}$$

Moreover, $\Phi_i^m = \Sigma(\overline{\tau}_i^m)$ can be interpreted as the transition matrix between times $\overline{\tau}_i^m$ and $\underline{\tau}_i^m$ of the homogeneous version of the ODE for which the derivative $\frac{\partial \bar{\Omega}}{\partial t_{\text{on},i}^m}(t)$ is a solution. The existence of the derivative is discussed in Appendix A of this dissertation.

Now, by computing the derivative of $\bar{\Omega}_i(t)$ with respect to $t_{\text{on},i}^m$ as $t \to (\overline{\tau}_i^k)^-$ (we recall that that at time $t = \overline{\tau}_i^k$ the derivative is discontinuous, since at this instant the target switches from not being observed to being observed), we obtain the following

recursive expression:

$$\frac{\partial \bar{\Omega}}{\partial t^m_{\text{on},i}}(\underline{\tau}^m_i) = \frac{\partial \underline{P}^m_i}{\partial t^m_{\text{on},i}} = (\Phi^m_i)^T \frac{\partial \overline{P}^{m-1}_i}{\partial t^m_{\text{on},i}} \Phi^m_i + \dot{\bar{\Omega}}_i((\underline{\tau}^m_i)^-). \tag{3.19}$$

Furthermore, defining $\Psi^m_i = \exp(A_i t^m_{\text{off},i})$ and using again Prop. 3, we get that:

$$\frac{\partial \overline{P}^{m+1}_i}{\partial t^m_{on,i}} = (\Psi^m_i)^T \frac{\partial \underline{P}^{m+1}_i}{\partial t^m_{on,i}} \Psi^m_i. \tag{3.20}$$

Now, repeating the same steps and propagating the previous expression to the k-th visit, $m \leq k \leq m + N_i - 1$, we get the recursion:

$$\frac{\partial \overline{P}^k_i}{\partial t^m_{\text{on},i}} = (\Lambda^{k,m}_i)^T \left(\frac{\partial \overline{P}^{m-1}_i}{\partial t^m_{\text{on},i}} + (\Phi^m_i)^{-T} \dot{\bar{\Omega}}_i((\underline{\tau}^m_i)^-)(\Phi^m_i)^{-1} \right) \Lambda^{k,m}_i, \tag{3.21}$$

where $\Lambda^{k,m}_i = \prod_{\alpha=m}^{k-1} \Psi^\alpha_i \Phi^\alpha_i$. In particular, for $k = m + N_i - 1$, due to periodicity, we have:

$$\frac{\partial \overline{P}^{m-1}_i}{\partial t^m_{\text{on},i}} = (\Lambda^{m+N_i-1,m}_i)^T \left(\frac{\partial \overline{P}^{m-1}_i}{\partial t^m_{\text{on},i}} + (\Phi^m_i)^{-T} \dot{\bar{\Omega}}_i((\underline{\tau}^m_i)^-)(\Phi^m_i)^{-1} \right) \Lambda^{m+N_i-1,m}_i, \tag{3.22}$$

which is a Lyapunov equation. Note that $\Lambda^{m+N_i-1,m}_i$ is stable, as discussed in Proposition 3. Therefore all of its eigenvalues have modulus lower than one. Also, since $\Lambda^{m+N_i-1,m}_i$ is a product of matrix exponentials, its null space is trivial. This implies that, since Lemma 2 tells us that $\dot{\bar{\Omega}}_i((\underline{\tau}^m_i)^-) \prec 0$, the Lyapunov equation has a unique negative definite solution and therefore $\frac{\partial \overline{P}^{m-1}_i}{\partial t^m_{\text{on},i}} \prec 0$.

Moreover, note that, for $m \leq k < m + N_i - 1$,

$$\frac{\partial \overline{P}^{m-1}_i}{\partial t^m_{on,i}} = (\Lambda^{m+N_i-1,k}_i)^T \frac{\partial \overline{P}^k_i}{\partial t^m_{on,i}} \Lambda^{m+N_i-1,k}_i, \tag{3.23}$$

which leads us to conclude that, $\forall k$, $\frac{\partial \overline{P}^k_i}{\partial t^m_{on,i}} \prec 0$.

The argument to claim that $\frac{\partial \overline{P}^k_i}{\partial t^m_{\text{off},i}} \succ 0$ is very similar to the one we just use to

show that $\frac{\partial \overline{P}_i^k}{\partial t_{\text{on},i}^m} \prec 0$. Therefore, only a brief summary will be given. Note that

$$\frac{\partial \overline{P}_i^m}{\partial t_{\text{off},i}^m} = (\Lambda_i^{m,m-1})^T \frac{\partial \overline{P}_i^{m-1}}{\partial t_{\text{off},i}^m} \Lambda_i^{m,m-1} + \dot{\bar{\Omega}}_i((\overline{\tau}_i^m)^-). \tag{3.24}$$

Using a similar recursion as in the previous proof, we get that

$$\frac{\partial \overline{P}_i^{m-1}}{\partial t_{\text{off},i}^m} = (\Lambda_i^{m+N_i,m-1})^T \frac{\partial \overline{P}_i^{m-1}}{\partial t_{\text{off},i}^m} \Lambda_i^{m+N_i,m-1} + (\Lambda_i^{m+N_i,m})^T \dot{\bar{\Omega}}_i((\overline{\tau}_i^m)^-) \Lambda_i^{m+N_i,m}. \tag{3.25}$$

As $\dot{\bar{\Omega}}_i((\overline{\tau}_i^m)^-) \succ 0$, then $\frac{\partial \overline{P}_i^{m-1}}{\partial t_{\text{off},i}^m} \succ 0$ and thus $\frac{\partial \overline{P}_i^k}{\partial t_{\text{off},i}^m} \succ 0$. □

In the sequel, we present the main result in this section in the form of a proposition that can be interpreted analogously to resource allocation problems where different targets are competing for the same resource $t_{\text{on},i}^k$. If all the targets have the same utility, except possibly those where the agent never dwells, an equilibrium in the minimax sense is reached. Hence, we denote as "active" a target that is visited for a non-null amount of time at least once during a cycle. In other words, a target i is said to be **active** if $\sum_{k=1}^{N_i} t_{\text{on},i}^k > 0$. A target is said to be **inactive** if the agent visits it but $\sum_{k=1}^{N_i} t_{\text{on},i}^k = 0$.

Unlike typical resource allocation problems, here the total resource $\sum_{i=1}^{M} \sum_{k=1}^{N_i} t_{\text{on},i}^k$ is not fixed. The reason why the total resource does not go to infinity (i.e. the period is guaranteed to be finite) is that increasing $t_{\text{on},i}^k$ to one target has an adverse effect on all other active targets.

Let us define the set of all active targets as $\mathcal{A} \subseteq \mathcal{V}$. Note that a target with unstable internal state dynamics has to be active; otherwise, the cost (3.2) will be unbounded. However, if a target has a stable A_i, its infinite horizon uncertainty without ever being observed can be lower than some other target's uncertainty, and thus the optimal policy would be to make such targets inactive.

Proposition 6. *For a fixed visiting sequence Ξ, a corresponding dwelling time sequence $\mathcal{T}$ that optimizes the cost* (3.2) *satisfies:*

$$\limsup_{t\to\infty} g_i(\|\Omega_i(t)\|) = \limsup_{t\to\infty} g_j(\|\Omega_j(t)\|),$$

for all $i, j \in \mathcal{A}$. Additionally, if $i \in \mathcal{A}$ and $p \notin \mathcal{A}$, then

$$\limsup_{t\to\infty} g_i(\|\Omega_i(t)\|) \geq \limsup_{t\to\infty} g_p(\|\Omega_p(t)\|).$$

Proof. First we focus on active targets and prove the first part of the proposition by contradiction, showing that if the property given in the proposition does not hold, then there is a way to re-balance the observation times that is guaranteed to improve the performance. Suppose that for some target i

$$g_i\left(\left\|\overline{P}_i^{\max}(t_{\text{on},i}^{1:N_i}, t_{\text{off},i}^{1:N_i})\right\|\right) < g_j\left(\left\|\overline{P}_j^{\max}(t_{\text{on},j}^{1:N_j}, t_{\text{off},j}^{1:N_j})\right\|\right),$$

where the upper index max indicates the maximum over k of $g_i(\|P_i^k\|)$. We now propose to decrease the amount of time of all observations of i by ϵ, while maintaining all other observation times for all the targets constant. According to (3.4b) in the proof of Lemma 2, this implies that the waiting time between observations for all the other targets will decrease. This updated policy (dwelling sequence) generates a new set of observation times for target i, (denoted by $\tilde{t}_{\text{on},i}^{1:N_i}$), and updated waiting times between visits for all the other active targets, (denoted by $\tilde{t}_{\text{off},j}^{1:N_j}$), while maintaining $t_{\text{off},i}^{1:N_i}$ and $t_{\text{on},j}^{1:N_j}$ constant. Note that $\exists\, \epsilon > 0$ such that $\tilde{t}_{\text{on},i}^{k} = t_{\text{on},i}^{k} - \epsilon$ for all $k \in \{1, .., N_i\}$ and $\tilde{t}_{\text{off},j}^{m} < t_{\text{off},j}^{m}$ for some $m \in \{1, ..., N_j\}$. Using Proposition 5, we get:

$$\overline{P}_i^{\max}(\tilde{t}_{\text{on},i}^{1:N_i}, t_{\text{off},i}^{1:N_i}) \succ \overline{P}_i^{\max}(t_{\text{on},i}^{1:N_i}, t_{\text{off},i}^{1:N_i}), \tag{3.26}$$

$$\overline{P}_j^{\max}(t_{\text{on},j}^{1:N_j}, \tilde{t}_{\text{off},j}^{1:N_j}) \prec \overline{P}_j^{\max}(t_{\text{on},j}^{1:N_j}, t_{\text{off},j}^{1:N_j}). \tag{3.27}$$

Using the fact that both the norm and the derivative are continuous and strictly increasing, we can always pick an ϵ small enough such that the new peak of target i is lower or equal to the new peak of j, i.e.,

$$g_j\left(\left\|\overline{P}_j^{\max}(t_{\text{on},j}^{1:N_j}, \tilde{t}_{\text{off},j}^{1:N_j})\right\|\right) < g_j\left(\left\|\overline{P}_j^{\max}(t_{\text{on},j}^{1:N_j}, t_{\text{off},j}^{1:N_j})\right\|\right),$$

$$g_i\left(\left\|\overline{P}_i^{\max}(\tilde{t}_{\text{on},i}^{1:N_i}, t_{\text{off},i}^{1:N_i})\right\|\right) \leq g_j\left(\left\|\overline{P}_j^{\max}(t_{\text{on},j}^{1:N_j}, \tilde{t}_{\text{off},j}^{1:N_j})\right\|\right).$$

Since under the updated policy, all the peaks $\overline{P}_j^m$, $1 \leq m \leq N_j$, $1 \leq j \leq M$, are lower for all the targets except i, we recall Lemma 3 and conclude that this updated policy has a lower cost than the previous one. Hence, the previous policy cannot be optimal, which proves the first part of the proposition.

For the second part of the proposition, we note that in an optimal solution if a target is inactive, its constant steady state covariance matrix has to be bounded, since otherwise the cost of the PM problem would be unbounded. Additionally, we note that its uncertainty at all times could be reduced by giving it a positive dwell-time (and thus increasing the common peak uncertainty of the active targets, and, as a consequence, the cost), using a very similar argument as in the first part of this proof. Thus, in an optimal dwelling sequence, a target will be inactive only if its peak uncertainty is lower (or equal) than that of the active targets. □

This proposition gives a necessary condition for the optimality of the dwelling sequence. Moreover, its constructive proof also gives insight on how to locally optimize the dwelling sequence for a given visiting sequence. However, in general, this property is not sufficient for determining an optimal dwelling sequence. In the next section, we will restrict ourselves to a specific set of visiting sequences Ξ where the optimality condition in Proposition 6 can indeed be exploited to optimize the dwelling times at each target.

3.3 Optimal Dwelling Sequence on a Constrained Visiting Sequence

While in the previous section we discussed a necessary condition for optimal dwelling time allocation, this condition alone is not sufficient to fully determine a *globally optimal* trajectory, since when a target is visited multiple times, this proposition does not give any insight on the values of the peak uncertainties for all the peaks, except the one with worse uncertainty. In this section we restrict ourselves to consider only visiting sequences Ξ where *each target is visited at most once during a cycle* (called "constrained" visiting sequences) and then extend that algorithm to more

general settings in the next section. Under this assumption, which applies throughout this section, we develop a practical algorithm that optimizes the dwelling sequence corresponding to a given Ξ. In some specific setups, we show that this approach gives the globally optimal dwelling sequence. Later on in this section, we discuss the process to obtain an optimal visiting sequence Ξ'. As a side note, in this section we will omit the upper index of $\overline{P}_i, t_{\text{on},i}$ and $t_{\text{off},i}$, since $N_i \leq 1, \forall i \in \Xi$.

The main idea behind our approach is to exploit the property that all peak uncertainties ($\overline{P}_i$) of active targets must coincide in an optimal solution. In particular, we develop an iterative scheme that balances the dwell-times ($t_{\text{on},i}$) such that the peak uncertainties coincide upon convergence of the algorithm. The update law used to update the dwell-times $t_{\text{on},i}$ of the active targets in this iterative scheme is:

$$t_{on,i}[k+1] = t_{on,i}[k] + k_p \log\left(\frac{g_i\left(\left\|\overline{P}_i\right\|\right)}{g_{\text{avg}}}\right), \tag{3.28}$$

where $g_{\text{avg}} = \left(\prod_{j\in\mathcal{A}} g_j\left(\left\|\overline{P}_j\right\|\right)\right)^{\frac{1}{|\mathcal{A}|}}$ and k_p is a small positive constant and $|\mathcal{A}|$ is the cardinality of $\mathcal{A}$. Equation (3.28) can be interpreted as aiming to achieve "consensus" regarding the peak uncertainties $\overline{P}_i$ among the active targets, and thus its structure is very similar to geometric mean consensus algorithms (Bullo, 2020). The expression (3.28) does not require the computation of gradients, which makes it computationally much less demanding than gradient-based approaches, such as those found in Chapter 2.

Remark 1. *At each iteration of* (3.28)*, it is necessary to compute* $\overline{P}_i$ *(given by the steady state solution of the Riccati equation) and* g_{avg}*. For computing* $\overline{P}_i$*, we use the Structure Preserving Doubling algorithm described in (Chu et al., 2004), which converges quadratically and is numerically stable. On the other hand, we note that* g_{avg} *can be computed distributively through a consensus protocol if* $\overline{P}_i$ *is computed locally by each target (Bullo, 2020).*

In order to simplify the convergence analysis of this update law, we abstract it

with the following differential equation:

$$\frac{d}{dr}t_{\text{on},i} = \begin{cases} 0, & \text{if } i \notin \mathcal{A},\ \frac{g_i(\left\|\overline{P}_i\right\|)}{g_{\text{avg}}} \leq 1, \\ k_p \log\left(\frac{g_i(\left\|\overline{P}_i\right\|)}{g_{\text{avg}}}\right), & \text{otherwise.} \end{cases} \tag{3.29}$$

Note that this version of the update law considers continuous parameter variation, i.e., the auxiliary variable r should be understood as the continuous time equivalent of "iteration index" and does not carry any actual "time" interpretation. Also, note that the hybrid structure of this update law allows for active targets become active and for inactive targets to become inactive, based on their $t_{\text{on},i}$ and $g(\left\|\overline{P}_i\right\|)$ values.

Proposition 7. *Under the update law* (3.29)*, the function* $\max_{i,j\in\mathcal{A}} |g_i(\left\|\overline{P}_i\right\|) - g_j(\left\|\overline{P}_j\right\|)|$ *is asymptotically stable.*

Proof. Note that $\frac{d\overline{P}_i}{dr} = \frac{\partial \overline{P}_i}{\partial t_{\text{on},i}}\frac{dt_{\text{on},i}}{dr} + \frac{\partial \overline{P}_i}{\partial t_{\text{off},i}}\frac{dt_{\text{off},i}}{dr}$ and

$$\sum_{i=1}^{M} \frac{dt_{\text{on},i}}{dr} = k_p \log\left(\frac{\prod_{j\in\mathcal{A}} g_j\left(\left\|\overline{P}_j\right\|\right)}{g_{\text{avg}}}\right) = k_p|\mathcal{A}| \log\frac{g_{\text{avg}}}{g_{\text{avg}}} = 0,$$

Therefore, under this control law, the period is constant. Thus, $\frac{dt_{\text{on},i}}{dr} = -\frac{dt_{\text{off},i}}{dr}$. Hence, for the active targets, $\frac{d\overline{P}_i}{dr} = (\frac{\partial \overline{P}_i}{\partial t_{\text{on},i}} - \frac{\partial \overline{P}_i}{\partial t_{\text{off},i}})k_p \log\frac{g_i(\left\|\overline{P}_i\right\|)}{g_{\text{avg}}}$. Consequently, if $g_i(\left\|\overline{P}_i\right\|) > g_{\text{avg}}$, $\frac{d\overline{P}_i}{dr} \prec 0$. Conversely, if $g_i(\left\|\overline{P}_i\right\|) < g_{\text{avg}}$, then $\frac{d\overline{P}_i}{dr} \succ 0$.

Since g_{avg} is the geometric mean, it is guaranteed to be between the maximum and minimum values of $g_j(\left\|\overline{P}_j\right\|))$. Thus, if $\max_{i\in\mathcal{A}} g_i(\left\|\overline{P}_i\right\|) \neq g_{\text{avg}}$, then $\frac{d}{dr}\max_{i\in\mathcal{A}} g_i(\left\|\overline{P}_i\right\|) < 0$ and $\frac{d}{dr}\min_{j\in\mathcal{A}} g_j(\left\|\overline{P}_j\right\|) > 0$.

Therefore, since both

$$\max_{i\in\mathcal{A}} g_i(\left\|\overline{P}_i\right\|) \text{ and } \min_{j\in\mathcal{A}} g_j(\left\|\overline{P}_j\right\|)$$

are monotonic and bounded, they converge. Now suppose they do not converge to the same value. Then, there exists r^* such that for $r > r^*$,

$$\log\left(\max_{i\in\mathcal{A}} g_i(\left\|\overline{P}_i\right\|)/\min_{j\in\mathcal{A}} g_j(\left\|\overline{P}_j\right\|\right) \geq \alpha > 0.$$

Then, for the target j with minimum peak uncertainty, $dt_{on,j}/dr \leq -\alpha/|\mathcal{A}|$, which means that the target j will eventually become inactive, since its observation time will reach zero within a finite increase of r. Therefore, a target cannot belong in the active set as $r \to \infty$ unless its peak uncertainty converges to $\max_{i\in\mathcal{A}} g_i(\|\overline{P}_i\|)$, which proves the proposition.

□

Remark 2. *In the proof of Proposition 7, we see that* $\frac{d}{dr}\max_i g_i(\|\overline{P}_i\|) < 0$. *Therefore, the update law* (3.29) *always reduces the cost defined in* (3.2). *This also implies that if not all the targets have the same peak value* $g_i\left(\|\overline{P}_i\|\right)$, *then the cost can be reduced by that update law.*

Remark 3. *The* log *in* (3.29) *is only one among many options of the update law. The essential feature is that* (3.29) *preserves the total dwelling time among different targets. Proposition 7 would still hold if we used other update laws that preserved this property. One of the possible update laws is* $\frac{d}{dr}t_{on,i} = g_i(\|\overline{P}_i\|)/\sum_j g_i(\|\overline{P}_j\|)$ *when* $i \in A$. *Here we use the* log *structure as our simulations indicate it yields good convergence rates.*

Now we show that the value achieved by update law (3.29) is unique, i.e. it does not depend on the observation time distribution at $r = 0$.

Lemma 4. *For a given cycle period* T *and a fixed constrained visiting sequence* Ξ, *there is a unique observation time distribution such that* $g_i(\|\overline{P}_i\|) = g_j(\|\overline{P}_j\|)$, $\forall\, i, j \in \Xi$.

Proof. Note that $g_i(\|\overline{P}_i\|)$ is a function only of $t_{\text{on},i}$, since the period T is fixed (i.e. $t_{\text{off},i} = T - t_{\text{on},i}$). Additionally, Prop. 5 asserts that $g_i(\|\overline{P}_i\|)$ is strictly decreasing with $t_{\text{on},i}$. Suppose there are two different sets of dwelling times ($t_{\text{on},i}$ and $t'_{\text{on},i}$), and consequently different costs g_{con} and g'_{con} such that all targets have the same peak value. Without loss of generality, we assume $g_{con} < g'_{con}$, which implies that $t_{\text{on},i} > t'_{\text{on},i}$. However, since the period is the same, we must have $\sum_i t_{\text{on},i} = \sum_i t'_{\text{on},i}$, which yields a contradiction. □

Finally, we give a specialization of Proposition 6 to the particular case discussed in this section.

Proposition 8. *For a fixed constrained visiting sequence Ξ and a given cycle period T, the dwelling sequence under the update law* (3.29) *converges to the optimal dwelling sequence (i.e., to $\mathcal{T}'$) that minimizes the cost function $J(\Xi, \mathcal{T})$ in* (3.2)*.*

Proof. For any dwelling sequence such that the period is T, the update law (3.29) always reduces the cost while maintaining T constant if $g_i(\left\|\overline{P}_i\right\|)$ is not the same for all the targets. Since there is a unique way such that every target has the same peak (and the update law (3.29) ensures convergence to it), the dwelling sequence after convergence of (3.29) has to be optimal, otherwise (3.29) would be able to improve the cost. □

For a fixed visiting sequence Ξ, we so far have shown a simple way to optimize the dwelling sequence given a fixed cycle period T. However, we have not addressed the problem of optimizing the value of the cycle period T. For this task, we use golden ratio search (Kiefer, 1953) that finds the global optimum in a unimodal function and local optima in a generic single variable function. The complete golden ratio search procedure is given in Alg. 3. The function $g_{con}(T)$ corresponds to running the update law in (3.28) until convergence and the value $g_{con}(T)$ is the value of g_{avg} at the final iteration and $r = (1 + \sqrt{5})/2$ (i.e., the search intervals are divided according to the golden ratio).

Algorithm 3 Search for the Optimal Cycle Period

Input: $T_{\min}$, $T_{\max}$.
$T_1 \leftarrow T_{\max} - (T_{\max} - T_{\min})/r$
$T_2 \leftarrow T_{\min} + (T_{\max} - T_{\min})/r$
while $|g_{con}(T_2) - g_{con}(T_1)| < \epsilon$ **do**
    if $f(T_2) > f(T_1)$ **then**
        $T_{\max} \leftarrow T_2$
    else $T_{\min} \leftarrow T_1$
    $T_1 \leftarrow T_{\max} - (T_{\max} - T_{\min})/r$
    $T_2 \leftarrow T_{\min} + (T_{\max} - T_{\min})/r$
Return: $(T_1 + T_2)/2$

Particular case: scalar state. Here we show that when the internal target state ϕ_i is a scalar (i.e. $L_i = 1$), the optimal peak uncertainty is a unimodal function of the cycle period T, in which case an *globally optimal* allocation of dwelling times can be achieved. First, we define a function $\beta_i(\rho, T)$ that returns the dwell-time $t_{\text{on},i}$ for a given target to achive a peak uncertainty of ρ with cycle period T. Note that, by definition, $t_{\text{off},i} = T - \beta_i(\rho, T)$, and it defines a peak uncertainty level as $g_i(||\overline{P}_i(\beta_i(\rho, T), T - \beta_i(\rho, T))||) = \rho$.

Note that the function $\beta(\rho, T)$ is well defined for any ρ such that $g_i(||\Omega_{i,ss}||) \leq \rho \leq g_i(||\Omega_i^\infty||)$ and $T > 0$. This is due to the fact that $\rho = g_i(||\Omega_{i,ss}||)$ will give $t_{\text{on},i} = \beta(\rho, T) = T$ and $\rho = g_i(||\Omega_i^\infty||)$ will give $t_{\text{on},i} = \beta(\rho, T) = 0$. Since the peak uncertainty of target i varies continuously with $t_{\text{on},i}$, we have that there will always be a $t_{\text{on},i}$ that yields a given value of ρ, if $g_i(||\Omega_{i,ss}||) \leq \rho \leq g_i(||\Omega_i^\infty||)$ and $T > 0$.

Now, we make the following technical assumption on the smoothness of the function $\beta(\rho, T)$, that is used on the proof of the following proposition. Note that while this assumption can be proved to hold, such a proof is outside the scope of this dissertation.

Assumption 3. *The function $\beta(\rho, T)$ is differentiable with respect to T whenever $\Omega_{i,ss} < \rho < \Omega_i^\infty$.*

Lemma 5. *When the state ϕ_i is a scalar, the function $\frac{\partial \beta_i(\rho,T)}{\partial T}$ is strictly positive and increasing.*

Proof. When computing $\frac{\partial \beta(\rho,T)}{\partial T}$, we leave the upper peak $\overline{P}_{i,k}(t_{\text{on},i}, t_{\text{off},i})$ constant and vary the period T. However, the lower peak $\underline{P}_{i,k}(t_{\text{on},i}, t_{\text{off},i})$ does not remain constant. Indeed, the variation of the lower peak can be computed as:

$$\frac{\partial \underline{P}_{i,k}}{\partial T} = \dot{\bar{\Omega}}_i(\underline{\tau}_i^-)\frac{\partial \beta_i(\rho, T)}{\partial T} = -\dot{\bar{\Omega}}_i(\underline{\tau}_i^+)\left(1 - \frac{\partial \beta_i(\rho, T)}{\partial T}\right)$$

Therefore, $\frac{\partial \beta_i(\rho,T)}{\partial T} = \frac{2A_i\underline{P}_i + Q_i}{\underline{P}_i^2 G_i} > 0$. Note that since $\underline{P}_i > \Omega_{i,ss}$, $0 < \frac{\partial \beta_i(\rho,T)}{\partial T} < 1$. Therefore, $t_{\text{off},i}$ also increases as T increases. This implies that $\underline{P}_i$ decreases with the increase of T, since $\underline{P}_i = \exp(-2A_i t_{\text{off},i})(\rho + Q_i) - Q_i$.

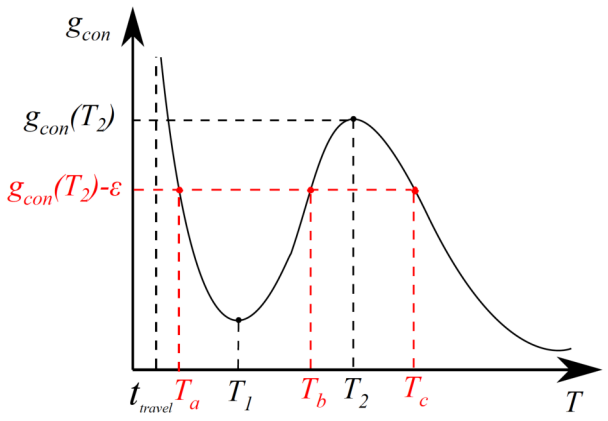


Figure 3·2: Illustration of the proof of Proposition 9.

Computing the variation of $\frac{\partial \beta_i(\rho,T)}{\partial T}$ with respect to P_i, we get that $\frac{\partial}{\partial P_i}\left(\frac{\partial \beta_i(\rho,T)}{\partial T}\right) = -\frac{2A_iP_i+Q_i}{P_i^3G_i}$, which is negative for any positive value of P_i. Since $\frac{\partial \beta_i(\rho,T)}{\partial T}$ is strictly decreasing with P_i and P_i is strictly decreasing with respect to T, we get that $\frac{\partial \beta_i(\rho,T)}{\partial T}$ is strictly increasing with the increase of T. □

Lemma 6. *Given a visiting sequence Ξ, where each target is only visited once, the equation $\sum_{i\in\Xi}\beta_i(\rho,T) = T - t_{travel}$ has a unique solution in T with ρ fixed.*

Proof. Denoting $\Gamma(T) = \sum_{i\in\Xi}\beta_i(\rho,T) - T + t_{\text{travel}}$ and using Lemma 5, we know that $\frac{\partial\Gamma}{\partial T}$ is a strictly increasing function. Now suppose that $\Gamma(T)$ has 3 or more distinct roots and we pick three of them $T_1 < T_2 < T_3$. The mean value theorem tells us that there is $\theta_1 \in (T_1, T_2)$ such that $\frac{\partial\Gamma}{\partial T}(\theta_1) = 0$ and $\theta_2 \in (T_2, T_3)$ such that $\frac{\partial\Gamma}{\partial T}(\theta_2) = 0$. This is a contradiction since $\theta_2 > \theta_1$ and $\frac{\partial\Gamma}{\partial T}$ is strictly increasing. □

Proposition 9. *When the state ϕ_i is a scalar, for a given visiting sequence Ξ, the optimal peak is a unimodal function of T.*

Proof. We prove this proposition by contradiction, supposing that there are at least two extremum points when considering the optimal peak as a function of T and showing that this contradicts Lemma 6.

We now show that, if there are two extremum points, then there is at least one peak value that could be generated by three different values of T. First, note that when $T \to t_{travel}$ the upper peak tends to $\max_i g_i\left(\|\Omega_i^\infty\|\right)$, since in that case no target is observed. In any non-degenerate dwell-times distribution, we have $P_i \prec \Omega_i^\infty$. Moreover, since there are at least two extremum points, there must exist a minimum

(at $T = T_1$) and a maximum (at $T = T_2$). Note that since the consensus peak for $T = t_{travel}$ is a maximum, we have $t_{travel} < T_1 < T_2$.

Let us denote $g_{con}(T)$ as the consensus peak for a given period T and take $\epsilon > 0$ such that $g_{con}(T_2) - \epsilon > g(T_1)$ and $g_{con}(T_2) - \epsilon = g_{con}(T_b) = g_{con}(T_c)$, for some $T_b,\ T_c$ such that $T_1 < T_b < T_2$ and $T_c > T_2$. Note that, since T_2 is a point of maximum and $g_{\text{con}}(\cdot)$ is continuous, such T_b, T_c and ϵ exist. Additionally, we note that there exists a $T_a \in [t_{travel}, T_1]$ such that $g_{con}(T_a) = g_{con}(T_2) - \epsilon$, since $g_{con}(T)$ is assumed continuous and $g_{con}(t_{travel}) > g_{con}(T_2) - \epsilon > g_{con}(T_1)$. These times, along with their respective g_{con} are illustrated in Fig. 3·2.

Therefore, if there are two extremum points, then $g_{con}(T_a) = g_{con}(T_b) = g_{con}(T_c)$, with $T_a \neq T_b \neq T_c$, which contradicts Lemma 6. Therefore, when an extremum point exists, it is a unique global minimum and hence the optimal peak is a unimodal function of T. □

Optimal Visiting Sequence We now focus on determining the optimal constrained visiting sequence (i.e., Ξ'). We will show that the optimal visiting sequence that visits every target is the minimum length tour that visits all the nodes in the network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

Proposition 10. *Among all the constrained visiting sequences where all targets are visited, for any dwelling sequence $\mathcal{T}$, the visiting sequence given by the TSP solution ($\Xi = \Xi_{TSP}$) has the lowest cost $J(\Xi, \mathcal{T})$ in* (3.2).

Proof. Recall that $t_{\text{off},i}$ is the sum of all the entries in the dwelling sequence $\mathcal{T}$ (omitting $t_{\text{on},i}$) and the sequence of travel-times corresponding to the visiting sequence Ξ. Since Ξ is a constrained visiting sequence, every target is visited only once. Hence, the visiting sequence given by the TSP solution Ξ_{TSP} is the one with the least amount of total travel-time and, according to Prop. 5, $g_i(\left\|\overline{P}_i\right\|)$ strictly increases with $t_{\text{off},i}$. Since for any dwelling sequence $\mathcal{T}$, the values of $t_{\text{off},i},\ \forall i$, are minimized if $\Xi = \Xi_{TSP}$ we conclude that the visiting sequence $\Xi = \Xi_{TSP}$ yields the lowest cost. □

Remark 4. *The problem of computing the optimal TSP cycle is NP-hard. However, efficient sub-optimal solutions are available (see e.g. (Tang et al., 2000)). The approach we discussed for optimizing the dwelling sequence does not rely on having the optimal TSP cycle and indeed can handle any cycle as long as every target is sensed*

(with a non-zero dwell-time) exactly once. Therefore, if finding the TSP cycle is computationally infeasible, we still can use a sub-optimal cycle (visiting sequence).

Note that in an optimal visiting sequence, inactive targets do not necessarily have to be a part of the agent's tour (and, even if they are, the agent will not have to dwell on them). Thus, when searching for the optimal visiting sequence, we can omit inactive targets, as long as the cost after the exclusion is lower than the cost of moving through that target without dwelling.

We now describe a procedure to obtain an optimal visiting sequence, without the restriction that every target must be visited (but still constrained to at most one visit per target). First, we order the targets according to their steady state uncertainty in the case where they are never observed, $g_i(\|\Omega_i^\infty\|)$. This quantity can be either finite or unbounded, and those targets which have infinite $g_i(\|\Omega_i^\infty\|)$ must necessarily be active. Among the targets with stable dynamics, the optimal action may be to never visit them, if $g_i(\|\Omega_i^\infty\|)$ is low enough. In order to determine the optimal sequence, we then progressively exclude the targets with lowest unobserved steady state uncertainty, until the overall cost stops to benefit from such exclusion.

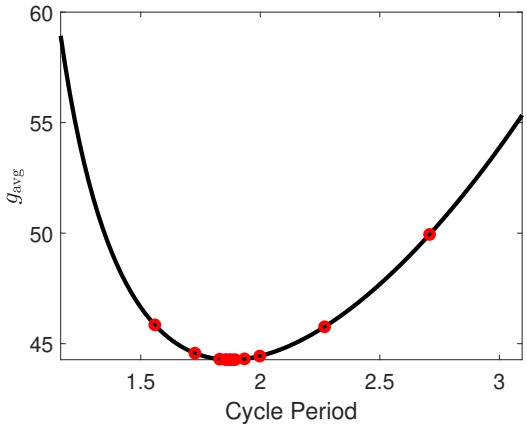
More formally, we assume that targets $1, ..., M$ follow an increasing order according to $g_i(\|\Omega_i^\infty\|)$. Then, the goal is to find the index m^* that minimizes the cost

$$C(m^*) = \max(g_{m^*}(\|\Omega_{m^*}^\infty\|), \text{Peak-Uncertainty}(\Xi^*(m^*)),$$

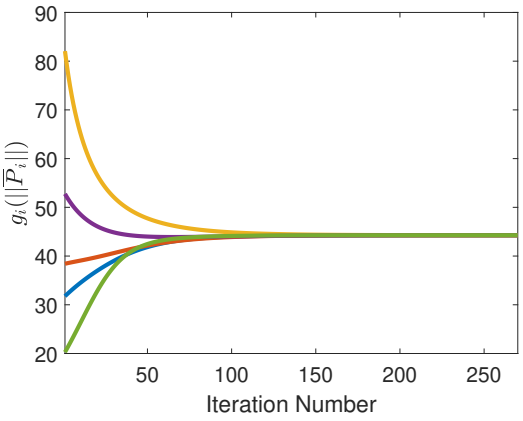
where $\Xi^*(m^*)$ is the shortest visiting sequence that visits all targets $m^*+1, ..., M$ and Peak-Uncertainty(Ξ) gives the peak uncertainty, given optimal dwelling sequence for the constrained visiting sequence Ξ computed using the iterative procedure given in (3.29). The cost $C(m^*)$ coincides with the overall PM (3.2) cost, considering $\Xi = \Xi^*(m^*)$ and optimal dwelling times. A simple way to solve this optimization is by sequentially evaluating the cost for $m^* = 0, 1, 2, ..., M$, however the optimal value

of m^* can also be determined more efficiently using Fibonacci search (Overholt, 1973), that only requires evaluating the cost $\mathcal{O}(\log M)$ times.

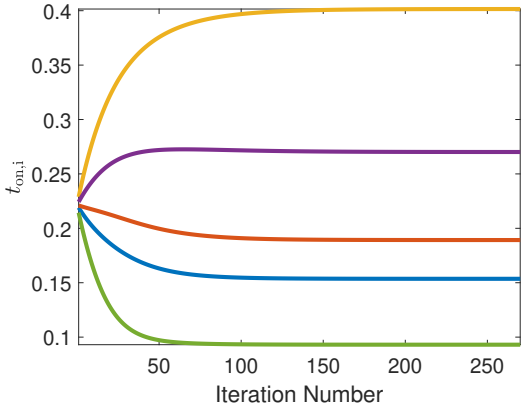
Note that the peak uncertainty of targets that remain active will never get worse by excluding a target from the set of active targets $\mathcal{A}$. Therefore, since the described procedure always excludes all the targets with unobserved steady state uncertainties lower than $g_{m^*}(\|\Omega^{\infty}_{m^*}\|)$, it is guaranteed to find the optimal visiting sequence.



(a) Peak value after balance among targets, as a function of cycle period.



(b) Peak uncertainty at the optimal period.



(c) Dwell-time at the optimal period.

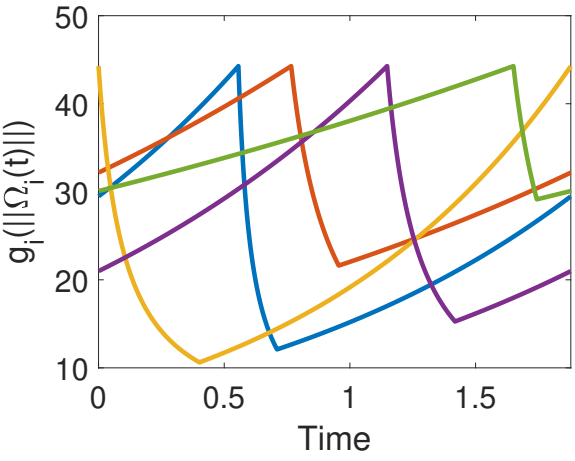
Figure 3·3: Results of simulating Algorithm 3. In (a), the balanced peak uncertainty, as a function of the cycle period. The red dots mark the values of T that were explored by the golden ratio search. In (b)-(c), we show the evolution of the peak uncertainty and the dwell-time for each target.

Some Simulation Results We have implemented the model described in (2.1) and (2.3), with parameters indicated in Table 3.1. Note that targets are also assigned colors that will be used to identify each target in figures containing the simulation results. For simplicity, the internal states of the targets were assumed to be scalars. Each target's location was drawn from a uniform distribution in $[0, 0.5] \times [0, 0.5]$. The target locations are displayed in Fig. 3·4b and the graph was assumed to be fully connected, with edge costs being the distance between two targets. Moreover, for the definition of the optimization goal as in (3.2), we used $g_i(\xi) = \xi$, $\forall i$, and $\|\Gamma\| = |\Gamma|$.

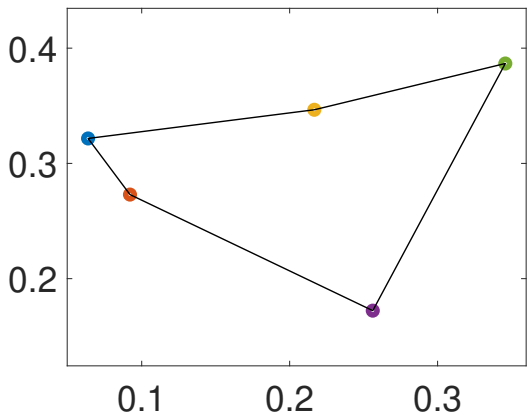
Table 3.1: Parameters used in the simulation.

Target	1	2	3	4	5
Color	blue	red	yellow	purple	green
A_i	0.3487	0.1915	0.4612	0.2951	0.1110
Q_i	1.1924	1.2597	0.8808	1.7925	0.4363
R_i	2.3140	7.1456	4.2031	5.2866	7.5314

For the visiting sequence, we considered the TSP cycle and Alg. 3 was then used to find the corresponding optimal dwelling sequence. The parameter k_p in Eq. (3.28) was chosen to be 10^{-2} and we set $[T_{\min}, T_{\max}] = [0.1t_{\text{travel}}, 3t_{\text{travel}}]$, where t_{travel} is the total travel-time required to complete the cycle.



(a) Covariance over one period.



(b) Agent trajectory (black) and target locations (colored).

Figure 3·4: Results obtained after optimizing the visiting and dwelling sequences.

The results are shown in Figs. 3·3 and 3·4. In particular, Fig. 3·3 shows the evolution of the steady-state covariance matrix norm over one complete period of the agent trajectory, under the optimal dwelling sequence. In Fig. 3·3, details of the optimization process are highlighted. In Fig. 3·3a, we can see how the peak uncertainty behaved as a function of the period (after balancing the dwelling times among targets). Moreover, this figure also highlights that the golden ratio search scheme efficiently converged to a global minimum. Figs. 3·3b and 3·3c show how the dwelling sequence and the peak covariance varied while using the update law (3.29). Initially, all targets are visited for the same amount of time. However, as the iterations go on, the dwell-time spent on some targets becomes larger than that of the others. As expected, in the final iteration, all the peak covariances have converged to the same value (consensus).

3.4 Optimal Dwelling Sequence on an Unconstrained Visiting Sequence

In Sec. 3.3, we only considered situations where each target was visited at most once in every cycle . We designed a procedure that computed the optimal dwelling sequence for a given visiting sequence and made some remarks about selecting the optimal visiting sequence. This section extends these ideas to settings where targets can be visited multiple times in a cycle (referred to as *unconstrained* sequences). In particular, we design an algorithm that aims to obtain a dwelling sequence such that the peak uncertainty (i.e., $g_i(\left\|\bar{P}_i^k\right\|)$) is the same at each visit k for every active target i. In other words, the goal is not only to have every target with the same maximum peak uncertainty, but also that the peak uncertainties within visits to the same target have the same peak value. Since multiple visits to the same targets are allowed, finding an optimal visiting sequence is more challenging than in Sec. 3.3,

as the possibilities for visiting sequences are more extensive and the optimization of the dwelling sequence in this case is not guaranteed to be optimal, even for a fixed period. Thus, in this section, we assume the visiting sequence is pre-determined and the process of determining an optimized (but not necessarily globally optimal) visiting sequence will be discussed in Sec. 3.5.

As we now consider situations where targets can be visited multiple times, to maintain the same notation as before, let

$$t_{\text{on},i} = \sum_{r=1}^{N_i} t^r_{\text{on},i}, \tag{3.30}$$

where the index r refers to the r^{th} visit to the target of interest, in the given visiting sequence Ξ. Thus, we maintain the same update law as in the case with constrained visiting sequences, given in (3.28).

However, only defining the total dwell-time spent at each target is not sufficient to determine the peak uncertainties as one has to determine how long the agent dwells at each of its visits at each target. To this end, we consider an additional optimization step, that takes place for each target for each total dwell-time update step k (of (3.28)). In particular, starting from an arbitrary valid distribution of dwell-times $t^p_{\text{on},i}[1]$ such that $\sum_{p=1}^{N_i} t^p_{\text{on},i}[1] = t_{on,i}[1]$, we propose the update law:

$$t^p_{\text{on},i}[m+1] = t^p_{\text{on},i}[m] + k_p \log\left(\frac{g_i\left(\left\|\overline{P}^p_i\right\|\right)}{g_{\text{avg,i}}}\right), \tag{3.31}$$

where $g_{\text{avg,i}} = \left(\Pi_{p=1}^{N_i} \bar{P}^p_i\right)^{\frac{1}{N_i}}$. The intuition behind this update law is that, similar to how a total dwell-time can be split among different targets to reach the same peak value, it can be split within the same target in order to yield the same peak value (at the beginning of each visit to that target). The complete optimization process is described in Alg. 4. We use the procedure in Alg. 5 (referenced as "MV") that

is responsible for balancing the dwelling time of multiple visits (MV) to the same target. Additionally, we point out that the procedure "computeToff" is responsible for computing the time between subsequent visits to the same target using (3.4b) and "SSPeaks" computes the steady state peak uncertainty values.

Algorithm 4 Computing the optimal dwelling sequence

Input: Visiting sequence Ξ, Cycle period T.
$k \leftarrow 1$;
$t_{\text{on},i}[k] \leftarrow 1/N$;
for $i \in \{1, ..., N\}$ **do**
    for $p \in \{1, ..., N_i\}$ **do**
        $t^p_{\text{on},i}[k] \leftarrow t_{\text{on},i}[k]/N_i$;
$g_{prev} \leftarrow \infty$;
while True **do**
    for $i \in \{1, ..., N\}$ **do**
        $[\bar{P}_i[k], t^p_{\text{on},i}[k]] \leftarrow \text{MV}(i, t_{\text{on},i}[k], \tilde{t}^p_{\text{on},j}[k], T, \Xi)$;
    $g_{\text{avg}} \leftarrow \left(\prod_{j\in\mathcal{A}} g_j\left(\left\|\overline{P}_j\right\|\right)\right)^{\frac{1}{M}}$;
    if $|g_{avg} - g_{prev}| < tol$ **then**
        Break;
    else
        $t_{on,i}[k+1] = t_{on,i}[k] + k_p \log\left(\frac{g_i(\left\|\overline{P}_i\right\|)}{g_{\text{avg}}}\right)$;
        for $i \in \{1, ..., N\}$ **do**
            for $p \in \{1, ..., N_i\}$ **do**
                $\tilde{t}^p_{\text{on},i}[k+1] \leftarrow \frac{t_{\text{on},i}[k+1]}{t_{\text{on},i}[k]} \tilde{t}^p_{\text{on},i}[k]$;
        $k \leftarrow k + 1$;
Return: $\bar{P}^p_i[k], t_{\mathbf{on},i}[k]$

When the visiting sequence is unconstrained, one special difficulty in optimizing the dwelling sequence is: to compute the steady state covariance, a target must know the information about the agent dwell-times at other targets. Recall that when each target was only visited once during a cycle, the number of visits $N_i = 1 = p$ and thus $t^p_{\text{off},i}$ was computed based only on the dwell-time at target i, i.e. $t^p_{\text{off},i} = T - t^p_{\text{on},i}$.

On the other hand, when multiple visits are allowed to each target, to compute the steady state covariance, one has to know the values of $t^p_{\text{on},i}$, $t^p_{\text{off},i}$ as well as $t^p_{\text{on},j}$ for

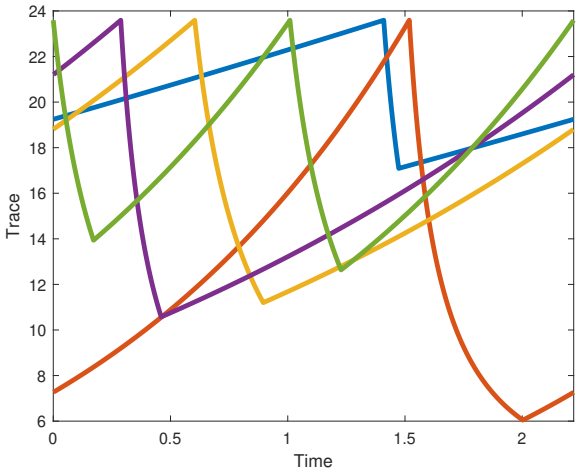
Algorithm 5 Computing dwell-times at a target: Process MV

1: **Input:** t, $t_{\mathbf{on},i}$, $t^p_{\mathbf{on},j\neq i}$, T, Ξ.
2: $t^p_{\text{off},i} \leftarrow \text{computeToff}(t^p_{\text{on},j\neq i}, T, \Xi)$;
3: $g_{prev,i} \leftarrow \infty$;
4: $m \leftarrow 1$;
5: **while** True **do**
6: $\quad [\bar{P}^1_i, ..., \bar{P}^{N_i}_i] = \text{SSPeaks}(t^p_{\text{on},i}, t^p_{\text{off},i})$;
7: $\quad g_{\text{avg},i} \leftarrow \left(\prod_{j\in\mathcal{A}} g_j\left(\|\overline{P}_j\|\right)\right)^{\frac{1}{M}}$;
8: $\quad$ **if** $|g_{\text{avg},i} - g_{prev,i}| < tol$ **then**
9: $\quad\quad$ **Break**;
10: $\quad$ **else**
11: $\quad\quad$ **for** $p \in \{1, ..., N_i\}$ **do**
12: $\quad\quad\quad t^p_{\text{on},i}[m+1] = t^p_{\text{on},i}[m] + k_p \log\left(\frac{g_i(\|\overline{P}^p_i\|)}{g_{\text{avg,i}}}\right)$;
13: $\quad\quad g_{prev,i} \leftarrow g_{\text{avg},i}$;
14: $\quad\quad m \leftarrow m+1$;
15: **Return:** $g_{\mathbf{avg},i}, t^p_{\mathbf{on},i}$;

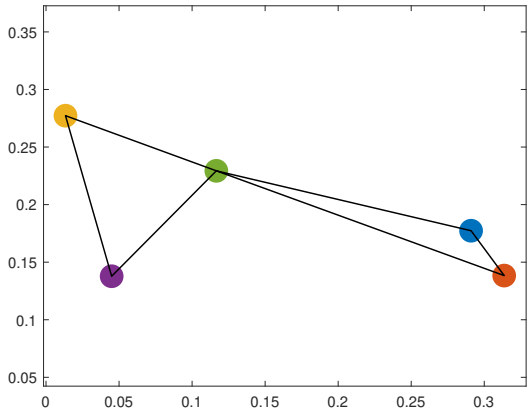
all $j \neq i$ (see (3.4b)). This interdependence does not allow independently optimizing the dwell-times and computing the steady state covariance at each target. Thus, as can be observed in line 18 of Alg. 4, to compute the steady state covariance (and dwelling sequence) at each target, what we propose is to assume each dwell-time holds the same proportional share of the total time as it did in the previous iteration. This assumption, however is heuristic and proving the convergence or optimality of this proposed method remains a topic of current research. Nevertheless, our simulation results lead us to believe that this algorithm (Alg. 4) converges for any positive cycle period T.

Simulation Results. In order to demonstrate the dwelling time allocation algorithm, we simulated it in a setting with 5 targets. The system parameters are given in 3.2. The visiting sequence was set to 5-4-3-5-1-2. In this simulation setting, target 5 (green) was visited twice over the period, and both peak uncertainties have the same

value, after convergence of the dwelling time allocation algorithm.



(a) Covariance over one period.



(b) Agent trajectory (black) and target locations (colored).

Figure 3·5: Results obtained after optimizing the unconstrained visiting and dwelling sequences.

Table 3.2: Parameters used in the simulation of unconstrained visiting sequence.

Target	1	2	3	4	5
Color	blue	red	yellow	purple	green
A_i	0.0307	0.3468	0.1706	0.1621	0.3029
Q_i	1.7739	1.0277	0.7247	1.0540	0.3690
R_i	3.812	2.9165	5.3625	3.0293	4.8251

3.5 A Greedy Solution for Determining an Optimal Visiting Sequence

In previous sections, we explored how to determine the dwelling sequence for a given fixed visiting sequence, constrained or not. In the constrained case, we also characterized the optimal visiting sequence. Naturally, to optimize the cost (3.2), the next step is to determine the optimal unconstrained visiting sequence on the given target network $\mathcal{G}$. For this problem, the visiting sequence given by the corresponding TSP is only a sub-optimal solution unless the visiting sequence is constrained (see

Prop. 10). In this section, we introduce a heuristic approach, specifically designed for the persistent monitoring formulation considered in this chapter, that iteratively refines a visiting sequence.

We first derive a lower bound of the cost of a visiting sequence that does not require computation of dwelling times. The goal of designing this lower bound is to use it to guide the greedy exploration scheme. Note that, in the greedy exploration process, the cost of a proposed visiting sequence needs to be evaluated multiple times. Avoiding the determination of the dwelling times through use of this lower bound proxy will significantly reduce the computational cost.

Our proposed greedy exploration uses a (possibly suboptimal) TSP solution based on the genetic algorithm based TSP solver (see e.g. (Singh and Baghel, 2009)), since it is computationally more efficient than, for example, using mixed-integer programming techniques (Hari et al., 2019). Using this TSP solution, the proposed greedy method *explores* several possible modifications to the current solution and *executes* the most profitable (greedy) modification. Intuitively, this greedy algorithm's computational efficiency depends on two factors: (i) the execution time required for the exploration of a single modified solution and (ii) the number of different modified solutions explored in each iteration. In order to strategically limit the computation time, motivated by (Welikala and Cassandras, 2020), we propose a novel *metric* that can be utilized to efficiently evaluate a solution (i.e., to explore a visiting sequence). To limit the latter, we exploit several structural properties of this PM setup.

3.5.1 The metric used to evaluate a visiting sequence

The optimal cost for a given visiting sequence (say $\bar{\Xi}$) is $J(\bar{\Xi}, \mathcal{T}')$, where J is as in (3.2) and $\mathcal{T}'$ is the optimal dwelling sequence. The optimization process to compute $\bar{\Xi}$ was described in Sec. 3.4. In this section, we establish a lower bound to $J(\bar{\Xi}, \mathcal{T}')$ as $\hat{J}(\bar{\Xi}) \leq J(\bar{\Xi}, \mathcal{T}')$, so that it can be used to efficiently evaluate any visiting sequence

$\bar{\Xi}$.

Let us denote a generic visiting sequence by $\bar{\Xi} = \{p_j\}_{j=1,2,\ldots,N}$ where each $p_j \in \mathcal{V}$. Note that the visiting sequence $\bar{\Xi}$ fully defines a corresponding sequence of edges $\bar{\xi} \subseteq \mathcal{E}$ as $\bar{\xi} = \{(p_{j-1}, p_j)\}_{j=1,2,\ldots,N}$ with $p_0 = p_N$. Since targets are allowed to be visited more than once during a cycle, some elements in $\bar{\Xi}$ may have repeated entries (i.e., there may be $p_j, p_l \in \bar{\Xi}$ such that $p_j = p_l$ even though $j \neq l$). For notational convenience, we define an equivalent cycle to $\bar{\Xi}$ that has unique entries as $\Xi = \{p_j^k\}_{j=1,2,\ldots,N}$ where p_j^k represents the k^{th} instance of the target $p_j \in \bar{\Xi}$. Recall that we previously used N_i to represent the number of times target i is visited in a cycle. Therefore, for each $p_j^k \in \Xi,\ 1 \leq k \leq N_{p_j}$, the corresponding sequence of edges of Ξ is denoted by $\xi = \{(p_{j-1}^l, p_j^k)\}_{j=1,2,\ldots,N}$. For example, if $\mathcal{V} = \{1,2,3\}$ in $\mathcal{G}$, $\bar{\Xi} = \{2,3,1,3,1,3\}$ is an example cycle where its equivalent version would be $\Xi = \{2^1, 3^1, 1^1, 3^2, 1^2, 3^3\}$. In essence, any given visiting sequence can be represented by the corresponding cycle $\bar{\Xi}$ or by any of its equivalent representations $\bar{\xi}$, Ξ or ξ.

Next, let us define the *auxiliary target-pool* of a target $p_j \in \bar{\Xi}$ as $\tau_{p_j} = \{p_j^1, p_j^2, \ldots, p_j^{N_{p_j}}\}$. The *sub-cycle* of a target $p_j^k \in \Xi$ is denoted as $\Xi_{p_j}^k$ and is defined in Ξ starting after p_j^{k-1} and going to p_j^k. For instance, in the previous example, the sub-cycles corresponding to targets 3^2 and 1^1 are $\Xi_3^2 = \{1^1, 3^2\}$ and $\Xi_1^1 = \{3^3, 2^1, 3^1, 1^1\}$ respectively. Similarly, $\xi_{p_j}^k$ is used to denote the sequence of edges of the corresponding sub-cycle $\Xi_{p_j}^k$. We further define $w_{p_j}^k$ as the total sub-cycle travel time required to traverse the edges in $\xi_{p_j}^k$.

We next prove that, for any dwelling sequence $\mathcal{T}$,

$$J(\bar{\Xi}, \mathcal{T}) \geq \max_{i \in \bar{\Xi}} L_i(\bar{t}_i), \tag{3.32}$$

where

$$L_i(\bar{t}_i) = g_i\Big(||\exp(A_i\bar{t}_i)\Omega_{\text{ss},i}\exp(A_i^T\bar{t}_i) + \int_0^{\bar{t}_i}\exp(A_i(\bar{t}_i-\tau))Q_i\exp(A_i^T(\bar{t}_i-\tau))d\tau||\Big), \quad (3.33)$$

with $\Omega_{\text{ss},i}$ being the positive definite solution of the algebraic Riccati Equation

$$A_i\Omega_{\text{ss},i} + \Omega_{\text{ss},i}A_i' + Q_i - \Omega_{\text{ss},i}G_i\Omega_{\text{ss},i} = 0$$

and

$$\bar{t}_i = \max_{k:i^k\in\tau_i} w_i^k. \quad (3.34)$$

Note that w_i^k can be understood as the k^{th} revisit time of the target $i \in \bar{\Xi}$ if no dwell-time was spent at any target in the sub-cycle ξ_i^k.

Remark 5. *The definition of* $\bar{t}_i$ *in* (3.34) *is equivalent to* $\max_{1\le k\le N_i} t_{off,i}^k$ *when* $t_{on,m}^n = 0$ *for all targets* m *and visiting instances* n*. In other words,* $\bar{t}_i$ *is the time spent exclusively traveling between two consecutive visits to target* i *given no dwell-time on any of the targets in the visiting sequence.*

The intuition behind this lower bound is that it computes the peak target covariance (which directly affects the cost $J(\bar{\Xi},\mathcal{T})$ in (3.2)) as if when the target is visited, its covariance instantaneously decreases to the steady state value. This is a key observation as it provides a computationally efficient metric

$$\hat{J}(\bar{\Xi}) = \max_{i\in\bar{\Xi}} L_i(\bar{t}_i), \quad (3.35)$$

to estimate the cost $J(\bar{\Xi},\mathcal{T})$ of a known visiting sequence $\bar{\Xi}$. As stated earlier, we use this $\hat{J}(\cdot)$ metric in our greedy scheme to efficiently evaluate and thus compare the cost of different visiting sequences so as to find the one that yields the lowest proxy for the actual cost.

Proposition 11. *$J(\bar{\Xi}, \mathcal{T}) \geq \max_{i \in \bar{\Xi}} L_i(\bar{t}_i)$ holds for any dwelling sequence $\mathcal{T}$ and visiting sequence $\bar{\Xi}$, with L_i defined as in* (3.35).

Proof. From the general solution of a linear matrix Riccati equation we get that

$$\overline{P}_i^k = \int_0^{t_{\text{off},i}^{k-1}} \exp(A_i(t_{\text{off},i}^{k-1} - \tau)) Q_i \exp(A_i^T(t_{\text{off},i}^{k-1} - \tau)) d\tau + \exp(A_i t_{\text{off},i}^{k-1}) \underline{P}_i^{k-1} \exp(A_i^T t_{\text{off},i}^{k-1}) \quad (3.36)$$

Moreover, we note that $\bar{\Omega}_i(t) \succ \Omega_{\text{ss},i}$, where $\Omega_{\text{ss},i}$ is the steady state covariance matrix of target i when it is permanently observed. Therefore,

$$\overline{P}_i^k \prec \int_0^{t_{\text{off},i}} \exp(A_i(t_{\text{off},i}^{k-1} - \tau)) Q_i \exp(A_i^T(t_{\text{off},i}^{k-1} - \tau)) d\tau + \exp(A_i t_{\text{off},i}^{k-1}) \Omega_{\text{ss},i} \exp(A_i^T t_{\text{off},i}^{k-1}) \quad (3.37)$$

Note that if we replace $t_{\text{off},i}$ by $\bar{t}_i$, we get the definition of $L_i(\bar{t}_i)$ in (3.32). Using Lemma 2, we know that if $\bar{t}_i \leq \max_{1 \leq k \leq N_i} t_{\text{off},i}^k$, then $J(\bar{\Xi}, \mathcal{T}) \geq \max_{i \in \mathcal{V}} L_i(\bar{\Xi})$ □

Note that this metric does not require the computation of dwelling times and thus can be immediately computed for a given trajectory. Additionally, we highlight that this metric for fast approximate cost evaluation can be used in conjunction with any heuristic method for exploration of visiting sequences, and is not exclusively tied to the specific heuristic we consider.

3.5.2 Possible types of modifications for a visiting sequence

As stated earlier, in each greedy iteration, we explore several modified versions of the current visiting sequence. In particular, we use three types of cycle modification operations (CMOs) to obtain modified cycles. Before discussing each of them, we first introduce some notation and a lemma.

Let us denote $\bar{\Xi}$ as the current cycle in a greedy iteration and Ξ as its equivalent

representation with unique entries and denote the respective sequences of edges $\bar{\xi}$ and ξ. We define the *critical target* $i^k \in \Xi$ as $i = \arg\max_{\gamma \in \Xi} \hat{J}(\bar{\Xi})$ and $k = \arg\max_{\alpha \in \tau_i} w_\alpha$ (i.e., the optimal $i \in \bar{\Xi}$ in (3.32) and the optimal $k \in \{1, 2, \ldots, N_i\}$ in (3.34)) as i^{k*}. The corresponding sub-cycle, the sequence of edges in this sub-cycle, and the total sub-cycle travel time are denoted as Ξ_i^{k*}, ξ_i^{k*} and w_i^{k*}, respectively.

Lemma 7. *The metric $\hat{J}(\bar{\Xi})$ can only be reduced (improved) by modifying the sub-cycle Ξ_i^{k*} so that w_i^{k*} is decreased.*

Proof. From (3.32), (3.34) and (3.33), it is clear that $\hat{J}(\bar{\Xi}) = L_i(w_i^{k*})$. According to (3.33), $L_i(\cdot)$ is a monotonically increasing function. Therefore, to reduce the metric $\hat{J}(\bar{\Xi})$, the maximum revisit time w_i^{k*} should be decreased. This can only be achieved if the corresponding sub-cycle Ξ_i^{k*} is modified. □

The above lemma implies that we only need to modify a portion of the complete cycle $\bar{\Xi}$ (the sub-cycle Ξ_i^{k*}) to improve the metric $\hat{J}(\bar{\Xi})$. This result significantly reduces the number of modified cycles that need to be explored in a greedy iteration. We are now ready to introduce the three types of cycle modification operations shown in Fig. 3·6.

CMO Type - I Remove an edge $(l^m, j^n) \in \xi_i^{k*}$ and replace it with the fastest path between targets $l, j \in \bar{\Xi}$.

Clearly, this modification is only effective if the fastest path between targets $l, j \in \bar{\Xi}$ is not the direct path $(l^m, j^n) \in \xi_i^{k*}$ that we remove. In practice, this CMO is useful in early greedy iterations - as we propose to start the greedy process with the TSP solution, where the agent is constrained to visit each target only once. Hence, such a TSP cycle may contain edges with high travel-time values that can be omitted if the agent is allowed to make multiple visits to some targets.

One example where CMO Type - I has helped to reduce the $\hat{J}$ value is illustrated in Fig. 3·7. The *cycle diagrams* in Fig. 3·7(b) and (c) convey the $\{L_i(w_i^k), i^k \in \Xi\}$ values

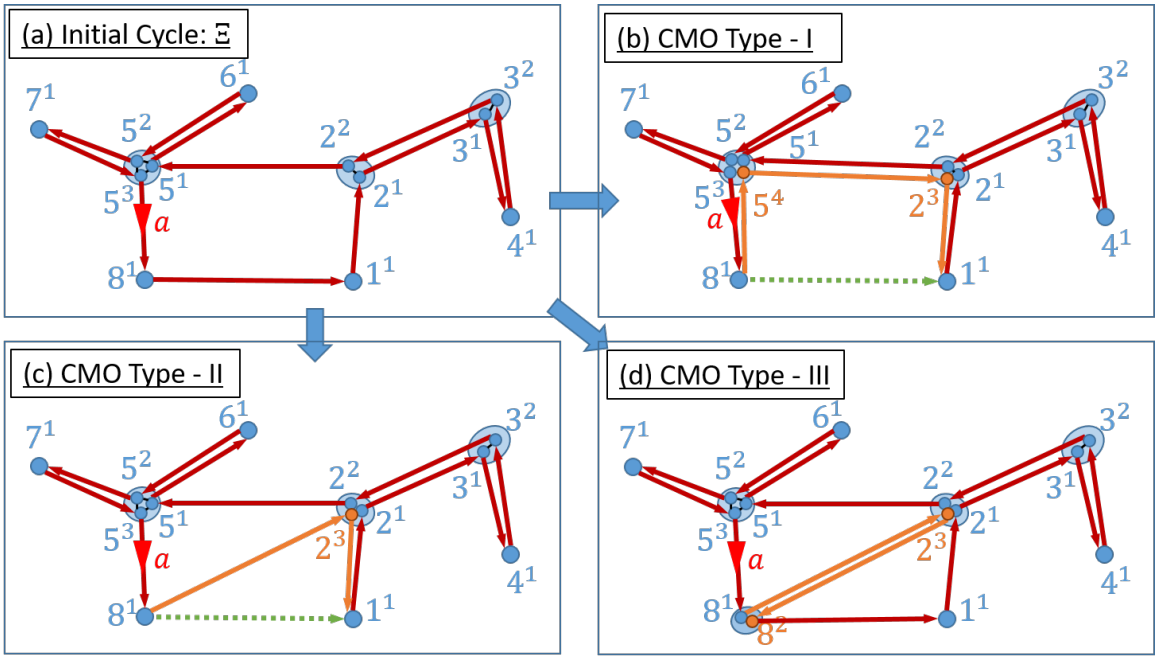


Figure 3·6: Three types of cycle modification operations (CMOs).

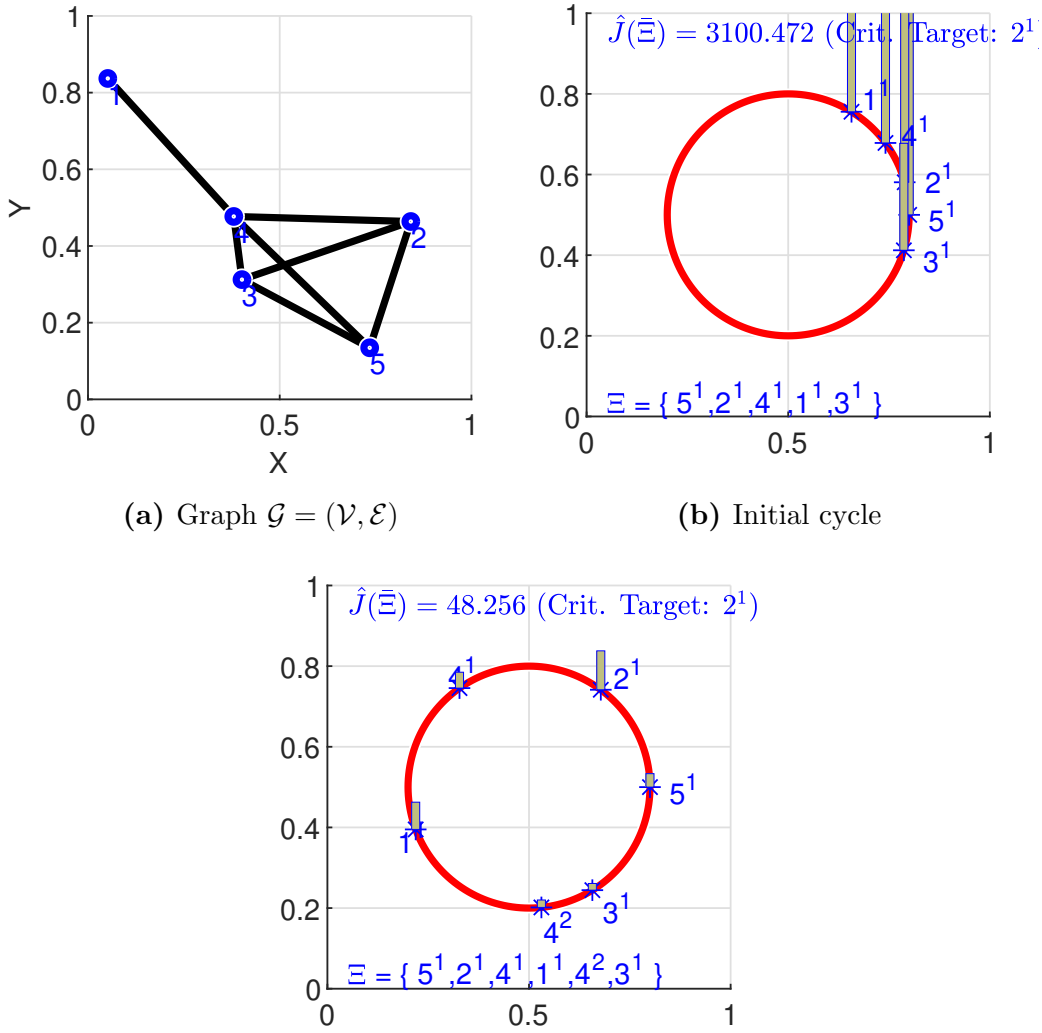
(as vertical gray colored bars) and travel-time values between targets (as circular red colored segments) of the cycle. Each such cycle diagram also indicates: (i) the cycle version with the unique entries Ξ, (ii) the lower bound metric value $\hat{J}(\bar{\Xi})$ and (iii) the critical target i^{k*}.

CMO Type - II Remove an edge $(l^m, j^n) \in \xi_i^{k*}$ such that $l^m \neq i^{k*}$ and $j^n \neq i^{k*}$. Then, replace it with two edges: $(l^m, i^{k*}), (i^{k*}, j^n)$.

The rationale behind this CMO is given in the following lemma that provides a way to evaluate whether this CMO improves $\hat{J}(\bar{\Xi})$, without the need even to compute the lower bound.

Lemma 8. *If the travel-times between targets $l, m, i \in \bar{\Xi}$ are such that $|w_{l,m} - w_{i,m}| < w_{l,m}$, then the CMO Type - II described above improves the $\hat{J}(\bar{\Xi})$ value.*

Proof. The CMO Type - II adds an extra visit to the target i^{k*} by dividing its sub-cycle Ξ_i^{k*} into two: say $\Xi_i^{k_1}$ and $\Xi_i^{k_2}$. Using the triangle inequalities, it can be shown that the corresponding new revisit times: say $w_i^{k_1}$ and $w_i^{k_2}$, will be lower than



(a) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

(b) Initial cycle

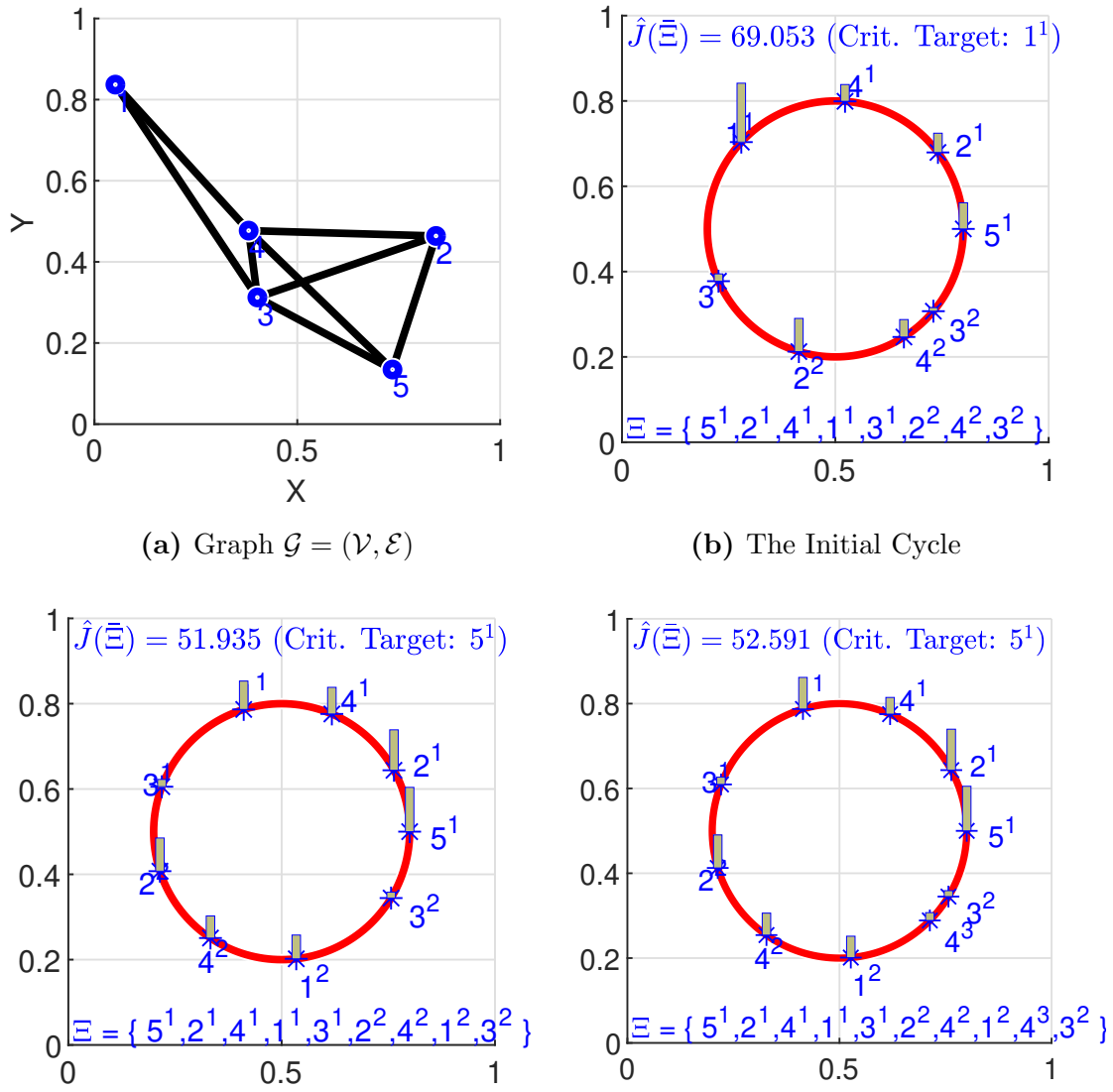
(c) After a CMO Type - I.

Figure 3·7: An example for CMO Type - I. Here, the edge $(1^1, 3^1)$ is replaced by the shortest path $(1^1, 4^2), (4^2, 3^1)$.

the previous (critical) revisit time w_i^{k*}. The proof is complete by observing that $\hat{J}(\bar{\Xi}) = L_i(w_i^{k*})$ (from (3.32), (3.34), (3.33)) and $L_i(\cdot)$ is a monotonically increasing function. □

CMO Type - III Select a target $j^n \in \Xi_i^{k*}$ such that $j^n \neq i^{k*}$ and insert two new target visits: $\{i^{k*}, j^{n+1}\}$ immediately after it.

Similar to the CMO Type - II, this CMO also adds an extra visit to the critical target i^{k*}.Lemma 8indicates under which conditions that this CMO can improve the metric $\hat{J}(\bar{\Xi})$. An illustrative example of CMO types I and II is shown in Fig. 3·8.



(a) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

(b) The Initial Cycle

(c) After a CMO Type - II.

(d) After a CMO Type - III.

Figure 3·8: Examples for CMO Type II and III. In (c), target 1^2 was inserted to the initial cycle (b). In (d), targets 1^2, 4^3 were added to the initial cycle (b).

Under these three types of CMOs, the total number of modified cycles to be

explored in a greedy iteration is strictly less than $3|\Xi_i^{k*}|$ (where $|\cdot|$ is the cardinality operator). Most importantly, as greedy iterations proceed, according to the CMO Type - II and III discussed above, we always break the critical sub-cycle Ξ_i^{k*} into two. Therefore the value of $|\Xi_i^{k*}|$ will effectively decrease with the iterations.

3.5.3 Greedy Algorithm

To efficiently construct a visiting sequence (say $\bar{\Xi}_G$), we propose the greedy scheme given in Alg. 6 (where ϵ is a positive number that can be as small as desired that ensures that the cost improvement at each iteration is not infinitesimal). It starts with the TSP solution and involves two sequential greedy expansion loops. In the first loop, only the CMO Type - I is explored, and in the second loop, both CMO Type II and III are explored in parallel. In a greedy iteration, the gain of executing a CMO defined as

$$\Delta_G = \hat{J}(\bar{\Xi}) - \hat{J}(\bar{\Xi}') \tag{3.38}$$

is explored where $\bar{\Xi}$ is the current visiting sequence and $\bar{\Xi}'$ is the modified visiting sequence.

Lemma 9. *The greedy algorithm given in Alg. 6 terminates after a finite number of iterations.*

Proof. The first greedy loop executes the CMO Type - I iteratively. Since it attempts to replace edges of the TSP solution with alternative fastest paths, it will only run for at most $|\Xi_{TSP}|$ iterations. To prove the termination of the second greedy loop, we use the fact that $\hat{J}(\bar{\Xi})$ is lower bounded: $\hat{J}(\bar{\Xi}) \geq 0$. Note also that each greedy iteration maintains the condition $\Delta_G \geq \epsilon$, which implies that $\hat{J}(\bar{\Xi})$ decreases monotonically over the greedy iterations. Hence, it is clear that $\Delta_G \to 0$ as greedy iterations go on, and thus the second greedy loop will terminate. □

Simulation Results We create random persistent monitoring problems by randomly generating network topologies together with target parameters. Such a PM

Algorithm 6 Greedy Construction of a Visiting Sequence

Input: Network topology: $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
$\bar{\Xi} \leftarrow \bar{\Xi}_{TSP} = \{\text{TSP solution for } \mathcal{G} = (\mathcal{V}, \mathcal{E})\}$;
$\Delta_G \leftarrow \epsilon$;
while $\Delta_G \geq \epsilon$ **do** ▷ First greedy loop.
    $\bar{\Xi}' \leftarrow \arg\max_{\bar{\Xi}'} \Delta\hat{J}(\bar{\Xi}'|\bar{\Xi}, \text{I})$;
    $\Delta_G \leftarrow \hat{J}(\bar{\Xi}) - \hat{J}(\bar{\Xi}')$;
    $\bar{\Xi} \leftarrow \bar{\Xi}'$;
$\Delta_G \leftarrow \epsilon$;
while $\Delta_G \geq \epsilon$ **do** ▷ Second greedy loop.
    $\bar{\Xi}' \leftarrow \arg\max_{\bar{\Xi}', Y \in \{\text{II,III}\}} \Delta\hat{J}(\bar{\Xi}'|\bar{\Xi}, Y)$
    $\Delta_G \leftarrow \hat{J}(\bar{\Xi}) - \hat{J}(\bar{\Xi}')$;
    $\bar{\Xi} \leftarrow \bar{\Xi}'$;
Return: $\bar{\Xi}_G \leftarrow \bar{\Xi}$ ▷ Greedily constructed cycle.

setup is shown in Fig. 3·9(a). Figures 3·9(b)-(e) show the evolution of the greedy visiting sequence $\bar{\Xi}$ and its cost (in terms of the metric $\hat{J}(\bar{\Xi})$) over three consecutive greedy iterations. Another two randomly generated persistent monitoring problems together with their respective high-performing greedily constructed visiting sequences are shown in Figs. 3·10 and 3·11.

3.5.4 Spectral Clustering Based Graph Partitioning

In order to partition the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we use the well-known *spectral clustering* technique (von Luxburg, 2007) mainly due to its advantages of: (i) simple implementation, (ii) efficient solution and (iii) better results compared to conventional partitioning techniques like the k-means algorithm (von Luxburg, 2007). In particular, the spectral clustering method derives the graph partitions based on a set of inter-target *similarity* values $\{s_{ij} \in \mathbb{R}_{\geq 0} : i, j \in \mathcal{V}\}$ so two targets that have high similarity will end up belonging to the same partition and two targets with low similarity to different ones.

Similarity Values In persistent monitoring, a similarity value s_{ij} should represent the effectiveness of covering both targets i and j in $\mathcal{V}$ using a single agent (Welikala and Cassandras, 2020). Specifically, we define s_{ij} using the *Gaussian Similarity Function*

$$s_{ij} = \exp\left(-\frac{|d(i,j)|^2}{2\sigma^2}\right). \tag{3.39}$$

where $d : \mathcal{V} \times \mathcal{V} \to \mathbb{R}_{\geq 0}$ is a *disparity metric* between two targets and σ is a parameter that controls how rapidly the similarity s_{ij} falls off with the disparity $d(i,j)$. According to (3.39), note that the similarity and disparity metrics are inversely related. We next focus on defining an appropriate disparity metric for the considered persistent monitoring paradigm.

Remark 6. *As a candidate for the disparity metric $d(i,j)$, neither using the physical distance nor the shortest path distance (between the targets i and j) provides a reasonable characterization to the underlying persistent monitoring aspects of the problem - as such metrics disregard target parameters as well as agent behaviors when monitoring targets.*

Considering the above remark, we propose a novel disparity metric named *covering cycle cost* (CCC):

$$d(i,j) = \min_{\bar{\Xi}:\, i,j \in \bar{\Xi}} \hat{J}(\bar{\Xi}), \tag{3.40}$$

with $\hat{J}(\bar{\Xi})$ defined as in (3.35). The intuition behind this metric is that, given two targets, the similarity is given by the lowest value of the lower bound (that serves as a proxy for the actual post) among all the closed path that contains these two targets. We also name the arg min of (3.40) as the *optimal covering cycle* (OCC) and denote it as $\bar{\Xi}^*_{ij}$. Simply, the OCC $\bar{\Xi}^*_{ij}$ is the best way to cover targets $i, j \in \mathcal{V}$ in a single cycle so that the corresponding cycle metric $\hat{J}(\cdot)$ is minimized. Therefore, if the CCC value is higher for a certain target pair, it implies that it is not effective to cover both those targets in a cycle (in other words, by a single agent). Thus, it is clear that the disparity metric $d(i,j)$ defined in (3.40) provides a good characterization of the

underlying persistent monitoring aspects of the problem.

Due to the combinatorial nature of the computation of the metric in (3.40), we use a greedy technique to obtain a sub-optimal solution for it. In particular, given a target pair $i, j \in \mathcal{V}$, we start with a candidate cycle $\bar{\Xi} = \{i\}$ which is then *iteratively expanded* (by adding external targets in $\mathcal{V} \backslash \bar{\Xi}$) in a greedy manner (using the greedy expansion approach similar to the ones described in Sec. 3.5) until it includes the target j. The terminal state of the candidate cycle $\bar{\Xi}$ is then considered as the OCC $\bar{\Xi}^*_{ij}$. The same procedure is next used to greedily construct the OCC $\bar{\Xi}^*_{ji}$. Finally, the corresponding CCC value, i.e., the disparity metric $d(i, j)$ in (3.40) is estimated as: $d(i, j) = \frac{1}{2}(\hat{J}(\bar{\Xi}^*_{ij}) + \hat{J}(\bar{\Xi}^*_{ji}))$.

Note, however, that for the computation of the similarity metric, the set of targets that will be part of Ξ is not defined a priori. Thus, the greedy expansion process considered here is slightly different from the one in Sec. 3.5. Therefore, for the sake of completeness, we now provide the details of the iterative greedy cycle expansion mechanism considered in this section. Take $\bar{\Xi}$ as the current version of the candidate cycle in a certain greedy iteration. Here $\bar{\xi}$ (similarly to ξ) represents the corresponding sequence of edges. Note however that, unlike ξ, the sequence $\bar{\Xi}$ may not contain all the targets in $\mathcal{V}$ (i.e., $|\mathcal{V} \backslash \bar{\Xi}| > 0$). There are three types of cycle expansion operations (CEOs) as shown in Fig. 3·12 that can be used to expand the current cycle $\bar{\Xi}$ so that it includes an external target $i \in \mathcal{V} \backslash \bar{\Xi}$.

CEO Type - I Replace an edge $(l, j) \in \bar{\xi}$ with two edges: $\{(l, i), (i, j)\}$.

CEO Type - II Select a target $j \in \bar{\Xi}$ and inset two target $\{i, j\}$ following j in the sequence Ξ.

CEO Type - III Select two targets $l, j \in \bar{\Xi}$ such that removing all the intermediate targets between them will not reduce the number of distinct targets in $\bar{\Xi}$.Then replace those intermediate targets with the external target i.

In this greedy algorithm, the gain of executing a CEO Type - Y where $Y \in$ {I, II, III} defined as

$$\Delta\hat{J}(\bar{\Xi}',\, j|\bar{\Xi}, Y) = \hat{J}(\bar{\Xi}) - \hat{J}(\bar{\Xi}') \tag{3.41}$$

is exploited to chose the most suitable: (i) CEO type Y, (ii) external target j and (iii) expanded cycle $\bar{\Xi}'$ (here, $\bar{\Xi}$ stands for the current cycle). The exact form of the greedy algorithm is provided in Alg. 7. In all, we use Alg. 7 together with (3.39) to compute the similarity values: s_{ij}, $\forall i, j \in \mathcal{V}$ between different targets in the network $\mathcal{G}$.

Algorithm 7 The greedy cycle expansion algorithm for the computation of disparity values $\{d(i,j) : i, j \in \mathcal{V}\}$

Input: Network of targets $\mathcal{G} = (\mathcal{V}, \mathcal{E})$
$d(i,j) \leftarrow 0$, $\forall i, j \in \mathcal{V}$;
for $i \in \mathcal{V}$ **do** ▷ For each start node.
  $\bar{\Xi} \leftarrow \{i\}$; ▷ Initial cycle.
  while $|\mathcal{V} \backslash \bar{\Xi}| > 0$ **do** ▷ Add all external targets.
    $[\bar{\Xi}',\, j] \leftarrow \underset{\bar{\Xi}',\, j \in \mathcal{V}\backslash\bar{\Xi},\, Y \in \{\mathrm{I, II, III}\}}{\arg\max} \Delta\hat{J}(\bar{\Xi}',\, j|\bar{\Xi}, Y)$; ▷ The expanded cycle and the added external target.
    $\bar{\Xi}' \leftarrow \underset{\bar{\Xi}, Y \in \{\mathrm{II, III}\}}{\arg\max} \Delta\hat{J}(\bar{\Xi}|\bar{\Xi}', Y)$ ▷ An optional further refinement for $\bar{\Xi}'$ step based on CMOs.
    $d(i,j) = d(i,j) + \frac{1}{2}\hat{J}(\bar{\Xi}')$;
    $\bar{\Xi} \leftarrow \bar{\Xi}'$ ▷ Update the current cycle.
Return $\{d(i,j) :\ i, j \in \mathcal{V}\}$;

3.6 Extension to Multi-Agent Problems

In this section, we extend the developed single-agent persistent monitoring solution to handle multi-agent systems. Let us denote the set of agents as $\mathcal{A} = \{1, 2, \ldots, N\}$. The

key idea here is to partition the target network $\mathcal{G}$ into N sub-graphs and then assign individual agents to each of those sub-graphs. This "divide and conquer" approach was motivated by two main reasons: (i) to uphold the "no target sharing" assumption made early on (due to the fact that sharing is ineffective (Welikala and Cassandras, 2021a; Welikala and Cassandras, 2020)) and (ii) to maintain the applicability of the developed single-agent persistent monitoring solution in previous sections. The main steps of the proposed multi-agent persistent monitoring solution are outlined in Alg. 8. In it, to execute Steps 3 and 5, we respectively use the techniques developed in Sections 3.5 and 3.4 (i.e., Alg. 6 and 4). The details of (the remaining) Steps 2 and 4 are provided in the following two subsections.

Algorithm 8 The multi-agent persistent monitoring solution.

1: **Inputs**: The network of targets $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and agents $\mathcal{A}$.
2: Partition the given graph $\mathcal{G}$ into N sub-graphs $\{\mathcal{G}^a\}_{a\in\mathcal{A}}$.
3: Find a high-performing cycle $\bar{\Xi}^a$ on each sub-graph $\mathcal{G}^a$.
4: Refine the sub-graphs and respective cycles: $\{\mathcal{G}^a, \bar{\Xi}^a\}_{a\in\mathcal{A}}$.
5: Compute the optimal dwelling sequence $\mathcal{T}^a$ corresponding to each visiting sequence $\bar{\Xi}^a$.

Spectral Clustering Algorithm In spectral clustering, the *Weighted Adjacency Matrix* W, the *Degree Matrix* D, and the *Laplacian Matrix* L plays a main role (von Luxburg, 2007). In our case, $W = S = [s_{ij}]_{(i,j)\in\mathcal{V}}$ (i.e., the *Similarity Matrix*), $D = diag(d_1, d_2, \cdots, d_M)$ where $d_i = \sum_{j=1}^{M} s_{ij}$, and $L = D - W$. Similarly to (Welikala and Cassandras, 2020), we specifically use the normalized spectral clustering technique proposed in (Jianbo Shi and Malik, 2000) where the normalized Laplacian $L_{rw} = D^{-1}L$ is used instead of L.

Algorithm 9 outlines the used normalized spectral clustering method. It gives the target clusters $\mathcal{V}^1, \mathcal{V}^2, \ldots, \mathcal{V}^N$ where each $\mathcal{V}^a$, $a \in \mathcal{A}$ is then used to form a *sub-graph* $\mathcal{G}^a = (\mathcal{V}^a, \mathcal{E}^a)$ out of the given graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ by selecting $\mathcal{E}^a \subseteq \mathcal{E}$ as the

set of intra-cluster edges taken from $\mathcal{E}$. Note that the set of inter-cluster edges (i.e., $\mathcal{E} \setminus \cup_{a\in\mathcal{A}} \mathcal{E}^a$) are now not included in any one of these sub-graphs.

Algorithm 9 Normalized Spectral Clustering Algorithm (Jianbo Shi and Malik, 2000)

1: **Input**: Normalized Laplacian L_{rw}.
2: Compute first N eigenvectors of L_{rw} as: $u_1, u_2, \cdots, u_N$.
3: Form $U \in \mathbb{R}^{M\times N}$ using $u_1, u_2, \cdots, u_N$ as its columns.
4: For $i = 1, \cdots, M$, let $y_i \in \mathbb{R}^N$ be the i^{th} row of U.
5: Cluster the data points $\{y_i\}_{i=1,\cdots,M}$ using the k-means algorithm into N clusters as: $C_1, C_2, \cdots, C_N$.
6: **Return** $\{\mathcal{V}^1, \mathcal{V}^2, \ldots, \mathcal{V}^N\}$ with each $\mathcal{V}^i = \{j : y_j \in C_i\}$

Once the sub-graphs are formed, we follow Step 4 of Alg. 8 by executing the greedy cycle construction procedure (i.e., Alg. 6) for each sub-graph. The resulting visiting sequence on a sub-graph $\mathcal{G}^a$ is denoted as $\bar{\Xi}^a$ and is assumed to be assigned (arbitrarily) to an agent $a \in \mathcal{A}$.

Simulation Results Figure 3·13 shows a few intermediate results obtained when executing Alg. 7 (to compute disparity values according to (3.40)) for an example target network.

Figure 3·14(a) shows an example target network with 15 targets. Assuming that three agents $\mathcal{A} = \{1, 2, 3\}$ are to be deployed to monitor these targets, Fig. 3·14(b) shows the generated sub-graphs when Alg. 9 is used. The yellow contours indicate the cycles constructed within each sub-graph (using Alg. 6) for each agent to traverse. Figs. 3·14(c)-(e) provide details of each agent's cycle. In these figures, notice that $\hat{J}(\bar{\Xi}^1) = 9.739$, $\hat{J}(\bar{\Xi}^2) = 10.113$ and $\hat{J}(\bar{\Xi}^1) = 8.951$, which implies that the worst L_i (3.32) value over the network is 10.113 (by sub-cycle $\bar{\Xi}^1_7$ of target instant 7^1 on agents 2's cycle).

We next use the same problem setup in Fig. 3·14(a) to highlight the importance of the proposed disparity metric in (3.40). Figure 3·15(a) shows the clustering result

obtained when the similarity values required in Alg. 9 are computed using a shortest path distance based disparity metric (instead of (3.40)). Observing the cycle assigned for the Agent 1 shown in Fig. 3·15(b), it is evident that now the worst L_i (3.32) value over the network is 21.079 (i.e., a 108.4% degradation from the use of (3.41)).

3.6.1 Target-exchange scheme (TES) used to refine sub-graphs

Practically, it is preferred to have the persistent monitoring load *balanced* across all the deployed agents. In other words, we prefer to have sub-graphs $\{\mathcal{G}^a\}_{a\in\mathcal{A}}$ that have a numerically closer set of visiting sequence metrics $\{\hat{J}(\bar{\Xi}^a)\}_{a\in\mathcal{A}}$. As a result of the used disparity metric in (3.40), the spectral clustering algorithm (Alg. 9) often directly provides such a balanced set of sub-graphs (e.g., like in Fig. 3·14(b) as opposed to Fig. 3·15(a)).

However, to further enforce this requirement, we propose a target exchange scheme (TES) between sub-graphs that attempts to balance the sub-graphs by minimizing the metric:

$$\hat{J}(\mathcal{G}) = \max_{a\in\mathcal{A}} \hat{J}(\bar{\Xi}^a). \tag{3.42}$$

Note that (3.42) can be simplified using (3.35), (3.34) and (3.32) into:

$$\hat{J}(\mathcal{G}) = \max_{a\in\mathcal{A}} \max_{i\in\mathcal{V}^a} L_i\Big(\max_{k:i^k\in\tau_i} w_i^k\Big). \tag{3.43}$$

According to (3.43), note that $\hat{J}(\mathcal{G})$ is determined by a specific (critical) target $i^k \in \Xi^a$ where the critical i, k, a are the optimal arguments of the three optimization problems involved in the R.H.S. of (3.43). Analogous to Lemma 7, it is clear that the only way to decrease $\hat{J}(\mathcal{G})$ is by modifying the sub-cycle corresponding to this particular critical target instant $i^k \in \bar{\Xi}^a$. Let us denote this critical sub-cycle as $\Xi_i^{k,a}$ and note that it is a segment of the cycle $\bar{\Xi}^a$ constructed on the sub-graph $\mathcal{G}^a$. A such feasible sub-cycle modification is to remove a target $j \in \Xi_i^{k,a}$ from agent a's trajectory

(i.e., from sub-graph $\mathcal{G}^a$). Clearly, one of the neighboring agents $b \in \mathcal{A}\backslash\{a\}$ will have to annex this removed target j into its cycle $\bar{\Xi}^b$.

Upon such a target exchange between sub-graphs $\mathcal{G}^a$ and $\mathcal{G}^b$, typically, both cycles $\bar{\Xi}^a$ and $\bar{\Xi}^b$ should be re-computed (using Alg. 6) respectively on the updated sub-graphs $\mathcal{G}^a$ and $\mathcal{G}^b$. Since Alg. 6 involves solving a TSP problem, we only use it when the optimal target exchange, i.e., the optimal target $j^* \in \Xi_i^{k,a}$ to remove and the optimal neighbor $b^* \in \mathcal{A}\backslash\{a\}$ to receive, is known. To find this optimal b^* and j^*, we again use a computationally efficient greedy scheme as follows.

First, we estimate the gain of annexing an external target j to a cycle $\bar{\Xi}^b$ as $\Delta\hat{J}_A(j, \bar{\Xi}^b)$ where

$$\Delta\hat{J}_A(j, \bar{\Xi}^b) = \max_{\bar{\Xi}^{b'},\ Y\in\{\mathrm{I,\,II,\,III}\}} \Delta\hat{J}(\bar{\Xi}^{b'}, j|\bar{\Xi}^b, Y), \tag{3.44}$$

where $\Delta\hat{J}(\bar{\Xi}^{b'}, j|\bar{\Xi}^b, Y)$ is given in (3.41) and $\mathrm{I, II, III}\}$ refer to the three CEOs considered in this chapter.

Next, we estimate the gain of removing a target j from a cycle $\bar{\Xi}^a$ as $\Delta\hat{J}_R(j, \bar{\Xi}^a)$ where

$$\Delta\hat{J}_R(j, \bar{\Xi}^a) = \hat{J}(\bar{\Xi}^a) - \hat{J}(\bar{\Xi}^{a\#}), \tag{3.45}$$

where $\bar{\Xi}^{a\#}$ represents the contracted version of the cycle $\bar{\Xi}^a$. This contracted version is obtained by following the steps: (i) remove the entries of j from $\bar{\Xi}^a$, (ii) bridge the gaps created by this removal using corresponding fastest paths and (iii) refine the obtained cycle (say $\bar{\Xi}^{a1}$) using CMOs (3.38). In particular,

$$\bar{\Xi}^{a\#} = \underset{\bar{\Xi}^{a'},\, Y\in\{\mathrm{II,III}\}}{\arg\min} \hat{J}(\bar{\Xi}^{a'}|\bar{\Xi}^{a1}, Y). \tag{3.46}$$

The final step of determining the optimal target exchange: b^*, j^* exploits $\Delta\hat{J}_A(j, \bar{\Xi}^b)$ and $\Delta\hat{J}_R(j, \bar{\Xi}^a)$ functions defined respectively in (3.44) and (3.45). In

particular, b^* and j^* are:

$$[b^*, j^*] = \underset{b \in \mathcal{A}\setminus\{a\},\ j \in \Xi_i^{a,k}}{\arg\max} \Delta\hat{J}_A(j, \bar{\Xi}^b) + \Delta\hat{J}_R(j, \bar{\Xi}^a). \tag{3.47}$$

Continuing the previous discussion, upon finding this optimal target exchange, we modify the sub-graphs $\mathcal{G}^a$ and $\mathcal{G}^{b^*}$ appropriately and use Alg. 6 to re-compute the respective cycles $\bar{\Xi}^a$ and $\bar{\Xi}^{b^*}$ on them. However, we only commit to this target exchange if the resulting $\hat{J}(\mathcal{G})$ in (3.42) is better (i.e., smaller) than before. Clearly, this target exchange process can be iteratively executed until there is no feasible target exchange that results in an improved $\hat{J}(\mathcal{G})$ metric.

The proposed TES above is distributed as it only involves one agent (sub-graph) and its neighbors at any iteration. Further, the proposed TES is convergent due to the same arguments made in the proof of Lemma 9. Finally, apart from the fact that this TES leads to a balanced set of sub-graphs, it also makes the final set of sub-graphs independent of the clustering parameter σ in (3.39) that we have to choose.

Simulation Results Figures 3·16 and 3·17 illustrate the operation and the advantages of the proposed TES for two different persistent monitoring problems.

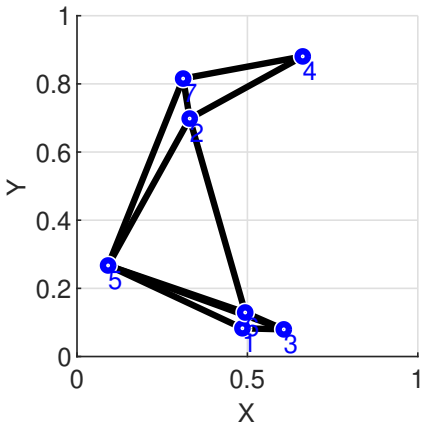
Finally, we use the randomly generated persistent monitoring problems shown in Fig. 3·18 to compare the performance (in terms of the cost J in (3.2)) of persistent monitoring solutions: (i) the minimax approach proposed in this chapter (labeled "Minimax"), (ii) the event-driven receding horizon control approach proposed in (Welikala and Cassandras, 2021a) (labeled "RHC") and (iii) the threshold-based basic distributed control approach proposed in (Welikala and Cassandras, 2021a) (labeled "BDC"). It should be highlighted that, compared to the minimax solution, both RHC and BDC solutions: (i) are distributed on-line, (ii) can only handle one-dimensional target state dynamics and (iii) have the objective of minimizing the average overall

target error covariance observed over a finite period. Despite these structural differences, it is important to note that all three solutions have the same ultimate goal of maintaining the target error covariances as low as possible.

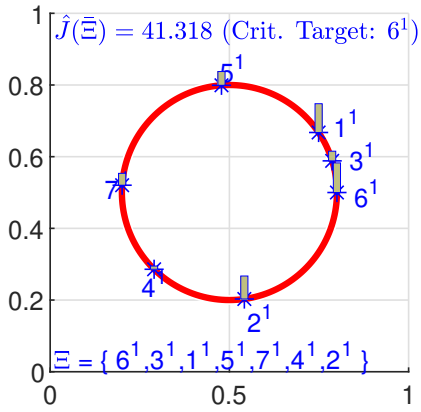
In particular, Fig. 3·18(a) and 3·18(b) show the initial conditions (i.e., at $t = 0$) of the two persistent monitoring problem setups considered. After executing the agent controls given by Minimax, RHC and BDC approaches until $t = 500$, the cost value J in (3.2) was evaluated as the maximum target error covariance value observed in the period $t \in [450, 500]$. The observed cost values are summarized in Tab. 3.3. The superiority of the Minimax approach proposed in this chapter is evident from the reported simulation results in Tab. 3.3. We note however that the other methods in Tab. 3.3 were not designed for the specific cost metric we consider in this chapter and, to the best of the authors knowledge, there is no algorithm in the literature designed specifically for the cost function we consider there.

Table 3.3: Performance comparison of different agent control methods under different persistent monitoring problem setups

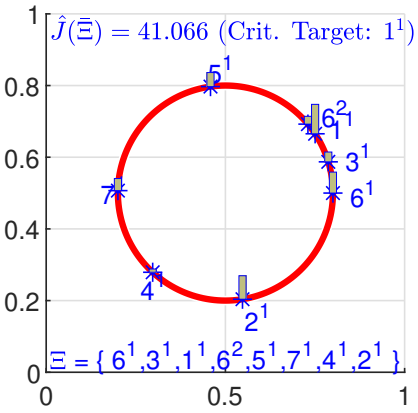
Cost J in (3.2)		Agent Control Method		
		Minimax	BDC (Welikala and Cassandras, 2021a)	RHC (Welikala and Cassandras, 2021a)
Problem Configuration	1	**40.23**	54.47	56.65
	2	**46.17**	456.55	110.87



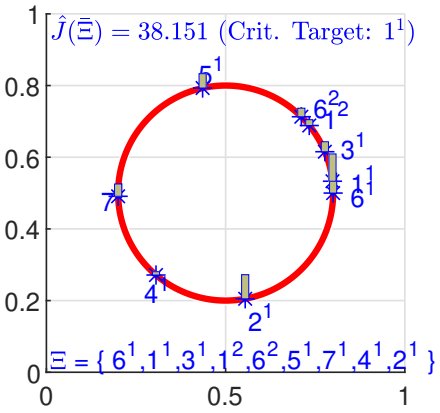
(a) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$



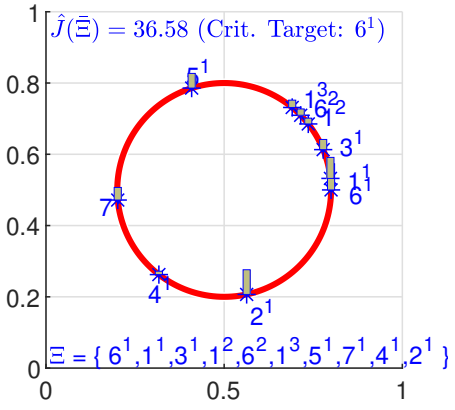
(b) Initial Cycle: $\bar{\Xi}_{TSP}$



(c) Intermediate Cycle 1

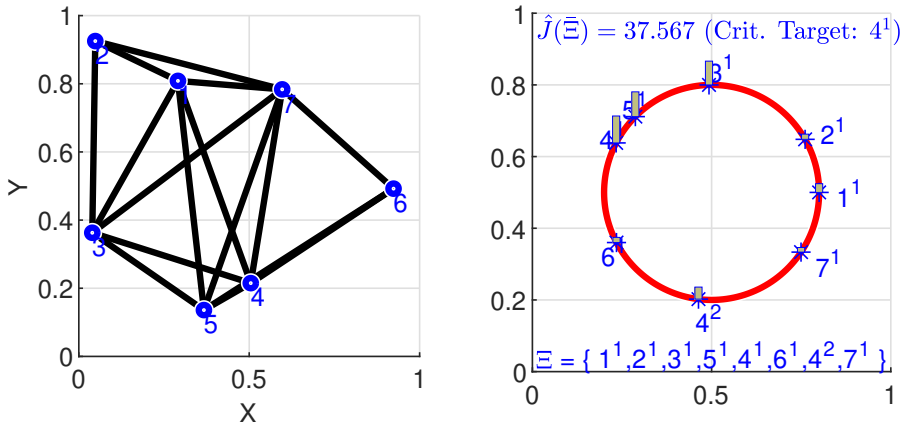


(d) Intermediate Cycle 2



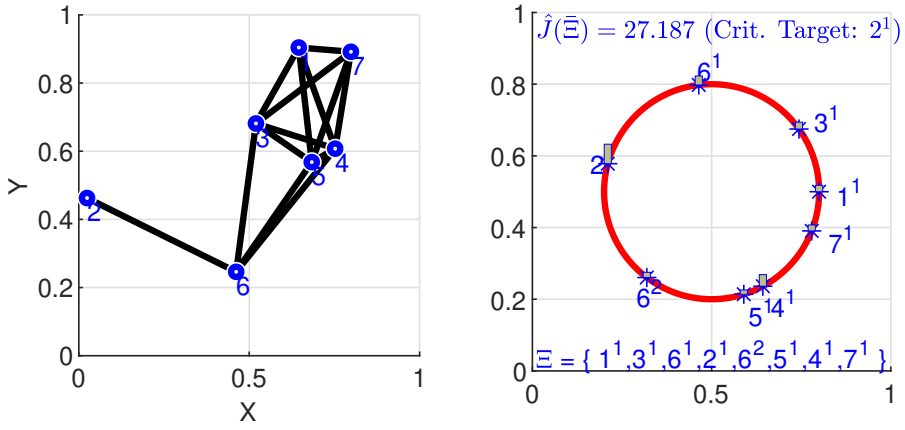
(e) Greedy Cycle: $\bar{\Xi}_G$

Figure 3·9: Example 1: The Greedy Cycle Construction Process.



(a) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ **(b)** Greedy Cycle: $\bar{\Xi}_G$

Figure 3·10: Example 2: A Constructed Greedy Cycle.



(a) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ **(b)** Greedy Cycle: $\bar{\Xi}_G$

Figure 3·11: Example 3: A Constructed Greedy Cycle.

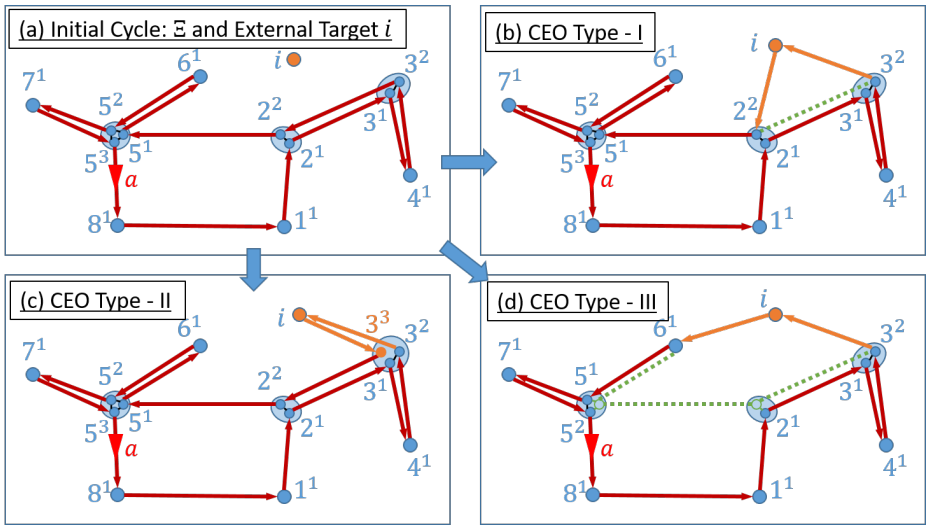
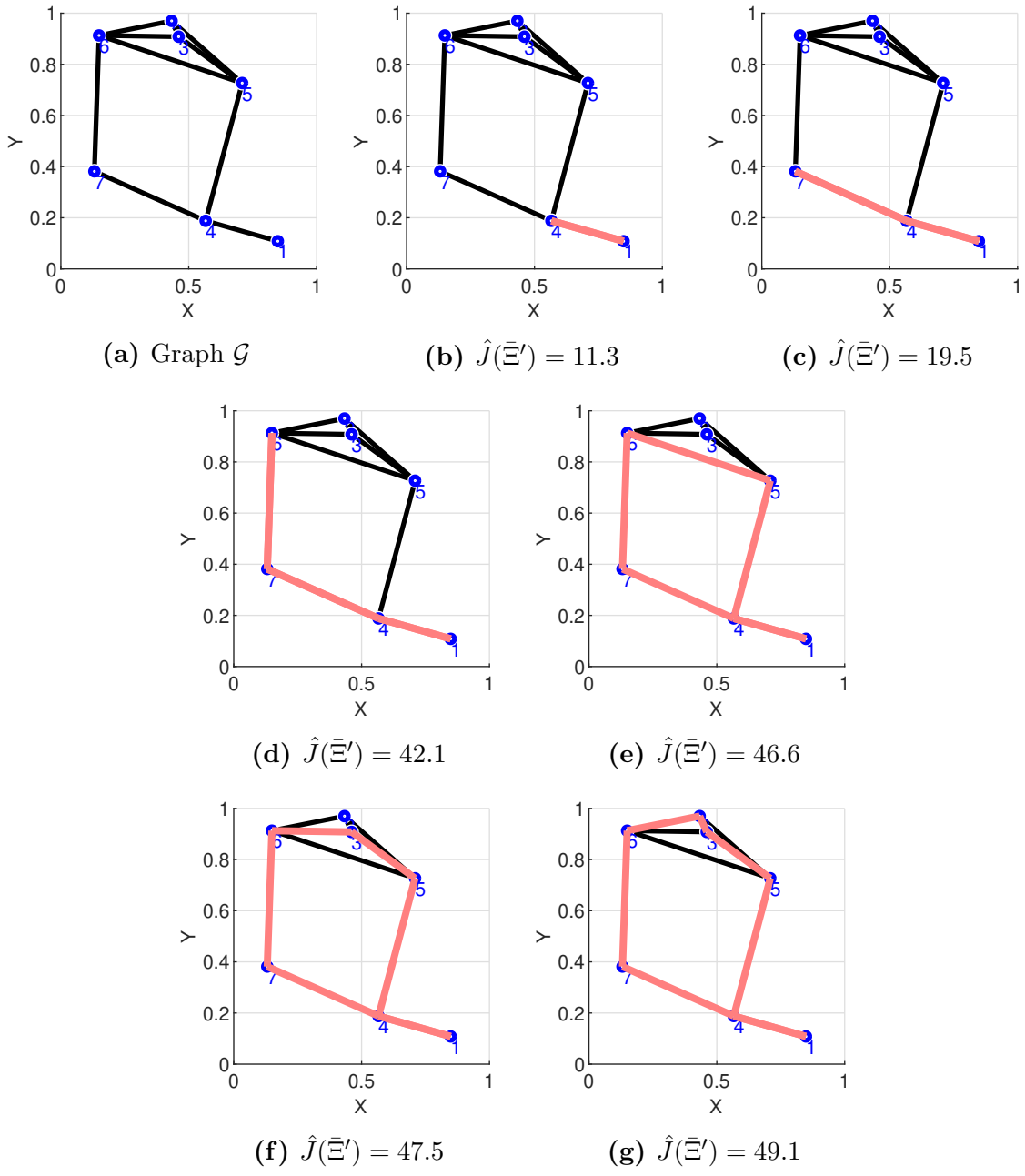


Figure 3·12: Three types of cycle expansion operations (CEOs).

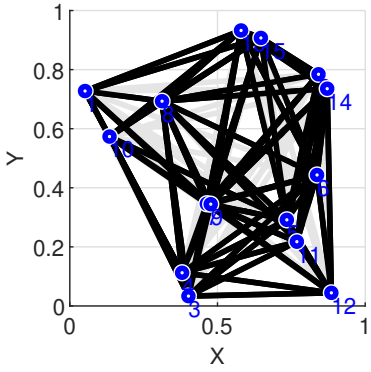


(a) Graph $\mathcal{G}$ **(b)** $\hat{J}(\bar{\Xi}') = 11.3$ **(c)** $\hat{J}(\bar{\Xi}') = 19.5$

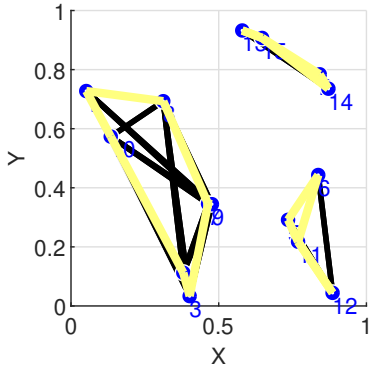
(d) $\hat{J}(\bar{\Xi}') = 42.1$ **(e)** $\hat{J}(\bar{\Xi}') = 46.6$

(f) $\hat{J}(\bar{\Xi}') = 47.5$ **(g)** $\hat{J}(\bar{\Xi}') = 49.1$

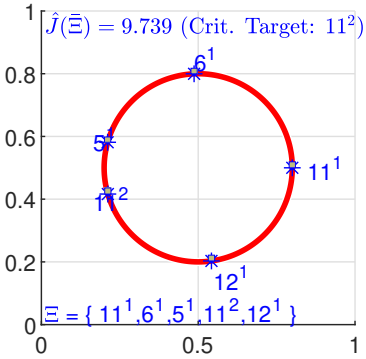
Figure 3·13: Greedy cycle expansion process for the computation of disparity values w.r.t. target 1: $\{d(1,j) : j \in \mathcal{V}\}$ using Alg. 7. The red contours in (b)-(g) show the expanded cycle $\bar{\Xi}'$ after executing Steps 5-9 of Alg. 7 with $i = 1$.



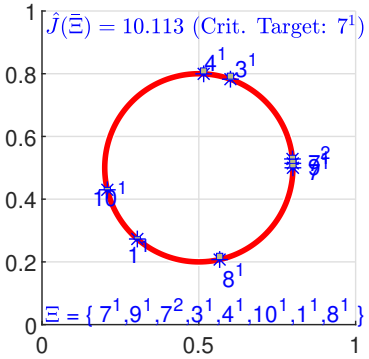
(a) Graph $\mathcal{G}$



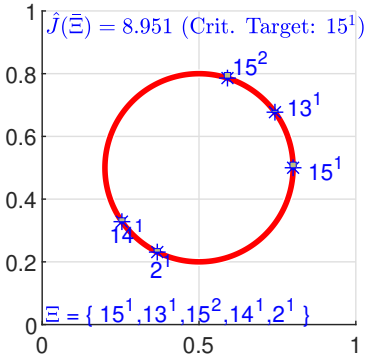
(b) $(\mathcal{G}^a, \bar{\Xi}^a) : a \in \mathcal{A}$



(c) Agent 1: $\bar{\Xi}^1$

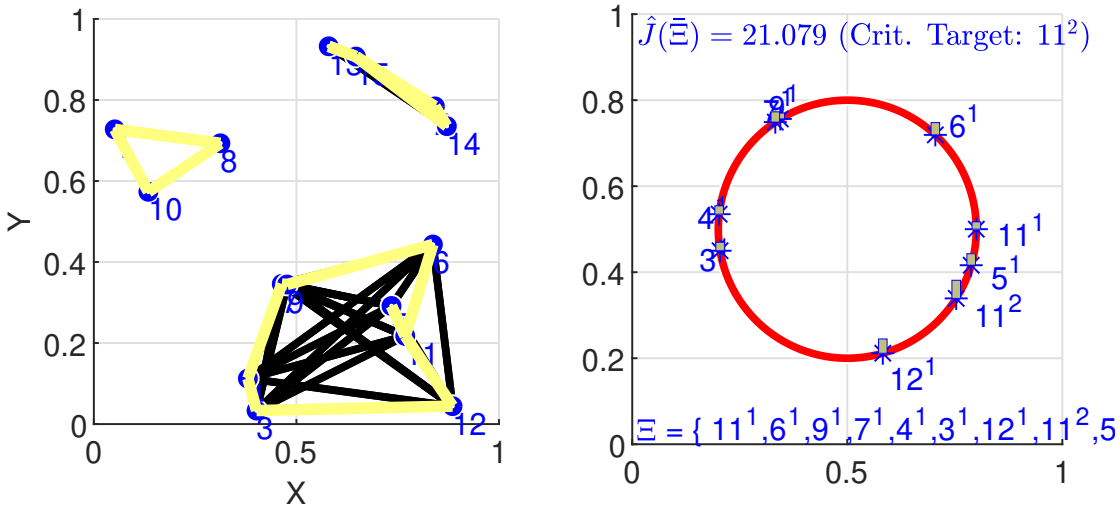


(d) Agent 2: $\bar{\Xi}^2$



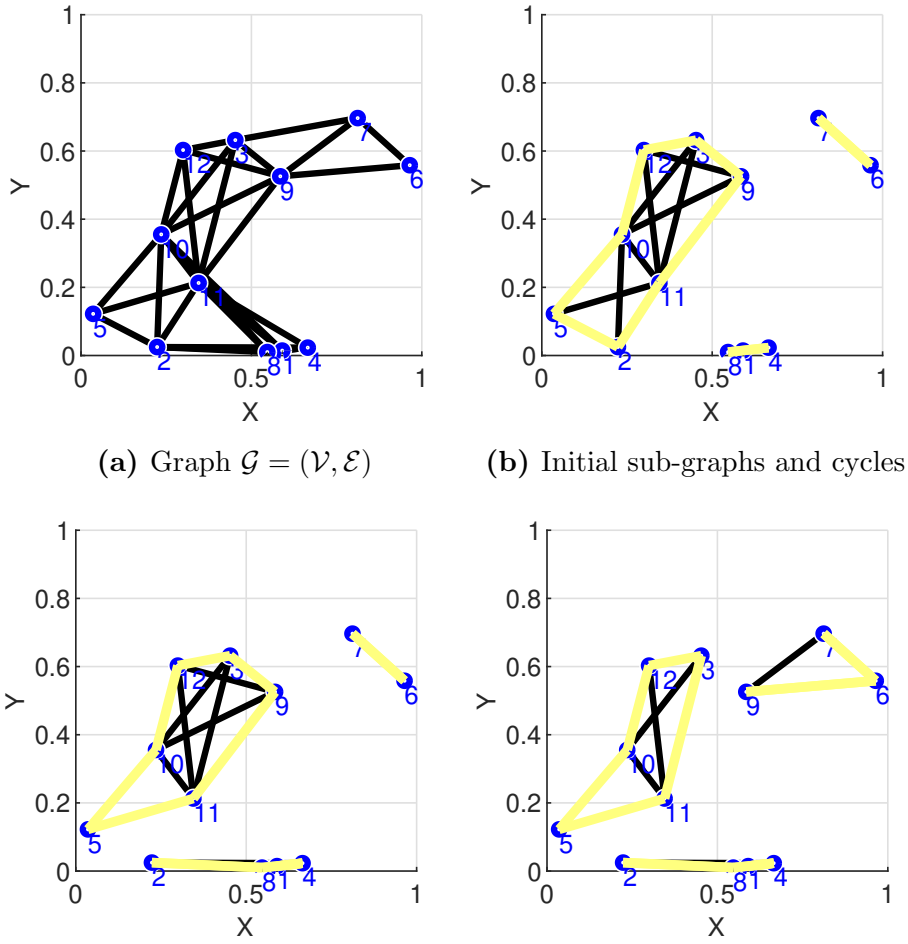
(e) Agent 3: $\bar{\Xi}^3$

Figure 3·14: Clustering results and the greedy cycles constructed in each sub-graph for individual agents.



(a) $(\mathcal{G}^a, \bar{\Xi}^a) : a \in \mathcal{A}$ **(b)** Agent 1: $\bar{\Xi}^1$

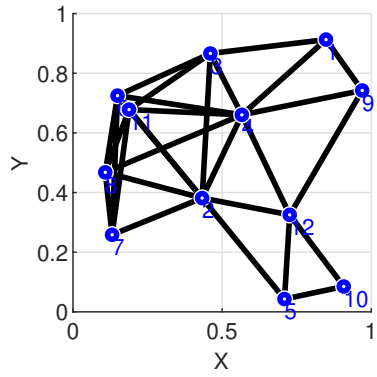
Figure 3·15: Clustering results (for the graph in Fig. 3·14(a)) when the shortest path distance is used as the disparity metric.



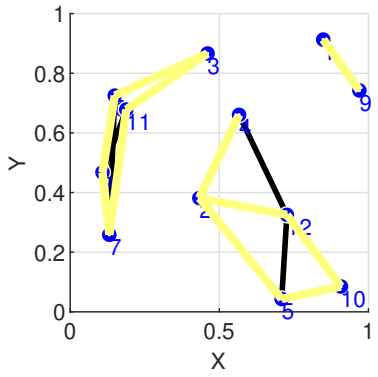
(a) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ **(b)** Initial sub-graphs and cycles

(c) After target exchange 1 **(d)** After target exchange 2

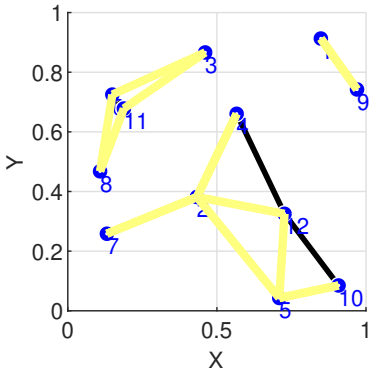
Figure 3·16: Target Exchange Scheme (TES) Example 1. Initial set of sub-graphs (in (b)): $\hat{J}(\mathcal{G}) = 16.3 = \max\{16.3, 6.9, 9.7\}$. Final set of sub-graphs (in (d)): $\hat{J}(\mathcal{G}) = 11.6 = \max\{11.2, 11.6, 10.9\}$ (Balanced and improved by 28.8%).



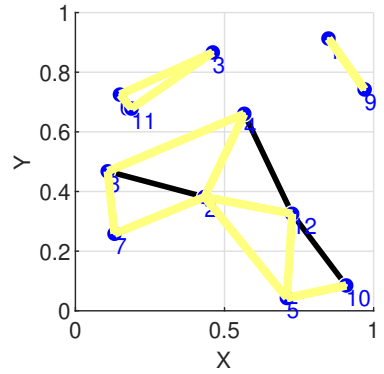
(a) Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$



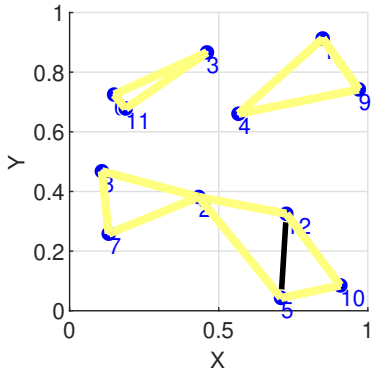
(b) Initial sub-graphs



(c) After target exchange 1

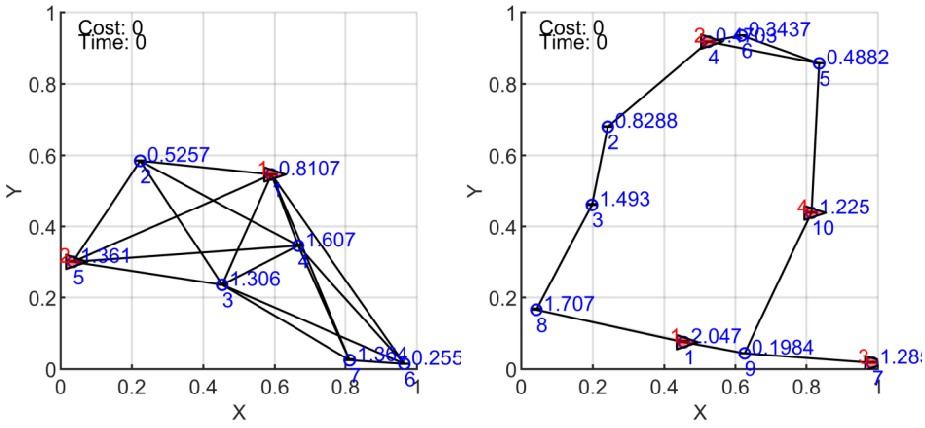


(d) After target exchange 2



(e) After target exchange 3

Figure 3·17: Target Exchange Scheme (TES) Example 2. Initial set of sub-graphs (in (b)): $\hat{J}(\mathcal{G}) = 12.3 = \max\{7.2, 12.3, 4.4\}$. Final set of sub-graphs (in (e)): $\hat{J}(\mathcal{G}) = 8.3 = \max\{7.7, 8.3, 5.6\}$ (Balanced and improved by 32.5%).



(a) Problem Configuration 1 **(b)** Problem Configuration 2

Figure 3·18: The persistent monitoring problem setups used in the performance comparison (at the initial condition).

Chapter 4

Discrete Time Formulation

In this chapter, we approach the problem from a discrete time perspective. This different perspective complements the work presented in the previous chapters, since it allows us to approach a richer class of problem formulations. Note, however, that the fact that the problem is in discrete time does not allow for fine design of dwelling times and, thus, we cannot take advantage of properties of optimal policies like we did in the previous chapter. Instead, we again approach the PM problem as an optimization problem and solve it directly. However, instead of basing on gradient-descent approaches, we formulate the Persistent Monitoring roblem as a semidefinite program (SDP). Thus, we can take advantage of state of the art SDP solvers, that are usually based on interior point methods that achieve polynomial-time convergence (MOSEK ApS, 2019). This formulation is a bit more flexible than the one presented in Chapter 3, allowing for sensing when the target-agent distance is not zero and maintaining a continuous environment. Additionally, while we use a less flexible model than the one presented in Chapter 2, our proposed solution in this chapter is able to go beyond the local optimality discussed in that chapter. In terms of computational effort, the algorithm discussed in this chapter is a middle ground between the (relatively) cheap one in Chapter 3 and the expensive gradient-based solution in Chapter 2. The results in this chapter have previously been published in (Pinto et al., 2021a).

4.1 Discrete Time Model

In this chapter, we use the following discrete time model for the targets' internal states:

$$\phi_{\Delta,i}(k+1) = A_{\Delta,i}\phi_{\Delta,i}(k) + w_{\Delta,i}(k), \tag{4.1a}$$

$$z_{\Delta,i}(k) = H_{\Delta,i}(k)\phi_{\Delta,i}(k) + v_{\Delta,i}(k), \tag{4.1b}$$

where the letter index Δ indicates the discrete time nature of that variable. As a natural extension of the continous time version, we assume that $w_{\Delta,i}(k)$ and $v_{\Delta,i}(k)$ are zero mean, mutually independent white Gaussian processes with constant covariance matrices $Q_{\Delta,i}$ and $R_{\Delta,i}$, respectively.

In this chapter again we assume that only one agent is available. Note that the divide and conquer approaches in Chap. 3 could be easily extended to also take into consideration this discrete time model. Moreover, we assume the following dynamics for the agent:

$$s_\Delta = s_\Delta(k) + u_{\Delta,k},\ \|u_{\Delta,k}\| \leq u_{\max}, \tag{4.2}$$

and the observation model

$$H_{\Delta,i}(k) = \sqrt{\gamma_{\Delta,i}(k)}H_{\Delta,i,\max}, \qquad \gamma_{\Delta,i}(k) = \begin{cases} \left(1 - \frac{(s_\Delta(k)-x_i)^2}{r_i^2}\right), & \|s_\Delta(k) - x_i\| \leq r_i, \\ 0, & \text{otherwise.} \end{cases} \tag{4.3}$$

This particular structure of $H_{\Delta,i}(k)$ captures the fact that the power of the signal decays as the agent moves farther from the target, while the noise power stays constant. If the distance between agent and target is larger than r_j, the intensity of the signal is zero, which is equivalent to not sensing at all. The particular quadratic decay was chosen due to the fact that it can be easily incorporated into an SDP.

Analogously to the continuous time case, the optimal estimator is a Kalman filter

and we restrict ourselves to periodic trajectories. The goal is to minimize the infinite horizon uncertainty, and we expect that the covariance matrix will converge to a unique (positive definite) limit cycle k as goes to infinity. In order to ensure that this convergence will indeed take place for trajectories where all the targets are visited, we make the following natural assumptions:

Assumption 4. *The pair* $(A_{\Delta,i}, H_{\Delta,i,\max})$ *is observable, for every* $i \in \{1, ..., M\}$.

Assumption 5. $Q_{\Delta,i}$, $R_{\Delta,i}$ *and the initial covariance matrix* $\Sigma_i(0)$ *are positive definite, for every* $i \in \{1, ..., N\}$.

Denoting $\Sigma_i(k)$ as the covariance matrix of the Kalman filter estimator, the $\mathcal{L}$-2 cost is then defined as:

$$C = \lim_{M\to\infty} \frac{1}{M} \sum_{m=1}^{M} \sum_{i=1}^{N} \mathrm{tr}(\Sigma_i(m)) = \frac{1}{\tau} \sum_{k=1}^{\tau} \sum_{i=1}^{N} \mathrm{tr}(\bar{\Sigma}_i(k)), \tag{4.4}$$

where τ is the period of the discrete time trajectory and $\bar{\Sigma}_i$ is the steady state covariance matrix.

4.2 An Optimization Approach for Computing the Infinite Horizon Cost

In order to jointly optimize the trajectory and compute the infinite horizon cost using SDPs, we write the Kalman filter in a different format, known as the information filter. We first briefly recall the relation between the Kalman filter and the information filter equations and then we show how to compute the infinite horizon cost of a schedule (i.e. periodic trajectory) using an SDP that solves the information filter equations.

Information Filter

Recal that the Kalman Filter equations can be written in two steps (prediction and update) (Thrun, 2002). The covariance update in the prediction is given by

$$\Sigma_i(k|k-1) = A_{\Delta,i}\Sigma_i(k-1|k-1){A_{\Delta,i}}^T + Q_{\Delta,i} \tag{4.5}$$

and in the update step by

$$\begin{aligned}\Sigma_i(k|k) = \Sigma_i(k|k-1) - \Sigma_i(k|k-1)H_{\Delta,i}^T(k) \\ \times (H_{\Delta,i}(k)\Sigma_i(k|k-1)H_{\Delta,i}^T(k) + R_{\Delta,i})^{-1}H_{\Delta,i}(k)\Sigma_i(k|k-1)\end{aligned} \tag{4.6}$$

where $\Sigma_i(k|k) = \Sigma_i(k)$ and $\Sigma_i(k|k-1)$ are the covariances at time k using information up to time k and $k-1$ respectively. For details, see, e.g. (Anderson and Moore, 2012; Thrun, 2002).

Under Assumption 5, we know that the covariance matrices $\Sigma_i(k|k-1)$ and $\Sigma_i(k|k)$ are positive definite. Let us define $P_i(k|k-1) = \Sigma_i^{-1}(k|k-1)$ and $P_i(k|k) = \Sigma_i^{-1}(k|k)$. Using the matrix inversion lemma (Thrun, 2002) in the prediction step (Thrun, 2002), we know that

$$P_i(k|k-1) = Q_i^{-1} - Q_i^{-1}A_{\Delta,i}(Q_{\Delta,i}^{-1} + A_{\Delta,i}^T P_i(k-1|k-1)A_{\Delta,i})^{-1}A_{\Delta,i}^T Q_{\Delta,i}^{-1}. \tag{4.7}$$

Analogously, using the matrix inversion lemma on the update step leads to

$$P_i(k|k) = P_i(k|k-1) + H_{\Delta,i}^T(k)R_{\Delta,i}H_{\Delta,i}i(k). \tag{4.8}$$

Merging both steps, we get the following recursion:

$$\begin{aligned}P_i(k|k) = Q_{\Delta,i}^{-1} + H_{\Delta,i}^T(k)R_{\Delta,i}H_{\Delta,i}(k) \\ - Q_{\Delta,i}^{-1}A_{\Delta,i}(Q_{\Delta,i}^{-1} + A_{\Delta,i}^T P_i(k-1|k-1)A_{\Delta,i})^{-1}A_{\Delta,i}^T Q_{\Delta,i}^{-1},\end{aligned} \tag{4.9}$$

which is the well known information filter recursion (Anderson and Moore, 2012) that we will use from this point forward in the chapter. The information filter is optimal, since it is simply a rearrangement of the Kalman filter equations, where instead of propagating directly the covariance, the inverse of the covariance is propagated.

Cyclic Schedules and the algebraic Riccati equation

In this subsection, we discuss one method for computing the steady state covariance matrix $\bar{\Sigma}_i(k)$. We pick this specific method due to the fact that it can be easily integrated in the SDP framework that we will explore in the next subsection. We take an approach similar to (Fujimoto et al., 2016), where the steady state covariance is computed using a single augmented algebraic Riccati equation (ARE). We point out that we cannot directly use the results in (Fujimoto et al., 2016) because they use the Kalman filter in its standard form and not the information version. In order to move to the ARE formulation, we define the following augmented covariance $\tilde{P}_{i,k} = diag(\bar{P}_i(k), ..., \bar{P}_i(k+\tau-1))$ and augmented parameters $\tilde{\Lambda}_i = \text{diag}(A_{\Delta,i}, ..., A_{\Delta,i})$, $\tilde{\Psi}_i = \text{diag}(Q_{\Delta,i}, ..., Q_{\Delta,i})$, $\tilde{H}_{i,k} = \text{diag}(H_{\Delta,i}(k), ..., H_{\Delta,i}(k+\tau-1))$, $\tilde{R}_i = \text{diag}(R_{\Delta,i}, ..., R_{\Delta,i})$. The recursion in (4.9) can be rewritten in terms of the augmented states as:

$$\tilde{P}_{i,k} = \tilde{\Psi}_i^{-1} + \tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} - \tilde{\Psi}_i^{-1}\tilde{\Lambda}_i(\tilde{\Psi}_i^{-1} + \tilde{\Lambda}_i^T \tilde{P}_{i,k-1}\tilde{\Lambda}_i)^{-1}\tilde{\Lambda}_i^T\tilde{\Psi}_i^{-1}. \tag{4.10}$$

For ensuring periodicity, we require that $P_i(k+\tau) = P_i(k)$, therefore,

$$P_{i,k+1} = JP_{i,k}J^T,\ J = \begin{bmatrix} 0_{(N-1)L_i \times L_i} & I_{L_i \times L_i} \\ I_{(N-1)L_i \times (N-1)L_i} & 0_{P \times (N-1)L_i} \end{bmatrix}. \tag{4.11}$$

Defining $\tilde{Q}_i^{-1} = (J^T diag(Q_{\Delta,i}, ..., Q_{\Delta,i})J)^{-1}$ and $\tilde{A}_i = J^{-T} diag(A_{\Delta,i}, ..., A_{\Delta,i})$ and rearraging (4.10) we get the following algebraic Riccati equation for computing $\tilde{P}_{i,k}$ for each of the targets i:

$$\tilde{Q}_i^{-1} - \tilde{P}_{i,k} + \tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} - \tilde{Q}_i^{-1}\tilde{A}_i(\tilde{P}_{i,k} + \tilde{A}_i^T\tilde{Q}_i^{-1}\tilde{A}_i)^{-1}\tilde{A}_i^T\tilde{Q}_i^{-1} = 0. \tag{4.12}$$

Solving the ARE as an SDP

Even though the most efficient methods for solving AREs do not rely on SDPs, in the path for jointly solving the ARE and optimizing the trajectory, we first describe how to cast the solution of the ARE as an SDP. This will allow us to benefit from the efficient solvers available for SDPs and from their convexity properties in order to efficiently approach the persistent monitoring problem. Moving in this direction, we first introduce a relaxed version of (4.12), where equality is replaced by inequality, with the goal that, in the optimal solution of the optimization, the constraint will be tight and equality will hold:

$$\tilde{Q}_i^{-1} - \Pi_i + \tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} - \tilde{Q}_i^{-1}\tilde{A}_i(\Pi_i + \tilde{A}_i^T\tilde{Q}_i^{-1}\tilde{A}_i)^{-1}\tilde{A}_i^T\tilde{Q}_i^{-1} \succcurlyeq 0, \tag{4.13}$$

where $\succcurlyeq 0$ denotes that the matrix is positive semi-definite and Π_i is a variable for which we want $\Pi_i = \tilde{P}_{i,k}$ as in (4.12) in the optimal solution of the optimization. Using the Schur complement (Balakrishnan and Vandenberghe, 1995), this inequality can be written as:

$$\begin{bmatrix} \tilde{Q}_i^{-1} - \Pi_i + \tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} & \tilde{Q}_i^{-1}\tilde{A}_i \\ \tilde{A}_i^T\tilde{Q}_i^{-1} & \Pi_i + \tilde{A}_i^T\tilde{Q}_i^{-1}\tilde{A}_i \end{bmatrix} \succcurlyeq 0. \tag{4.14}$$

We also define an upper bound Γ_i on the covariance matrix $\Gamma_i \succcurlyeq \Pi_i$, which in the optimal solution will coincide with the covariance matrix. Using Schur's complement, this upper bound can be expressed as:

$$\begin{bmatrix} \Gamma_i & I \\ I & \Pi_i \end{bmatrix} \succcurlyeq 0. \tag{4.15}$$

Now, we show that using an SDP, we can compute the exact solution of the information filter and that the relaxations we proposed will indeed be tight in an optimal solution. Moreover, we show that the cost function of the optimization is

equal to the trace of the augmented steady state covariance matrix. Inspired by (Balakrishnan and Vandenberghe, 1995), where it is shown that the LQR Riccati equation can be solved as an SDP, we give the following proposition:

Proposition 12. *If the pair $(\tilde{A}_i, \tilde{H}_{i,k})$ is observable and Q_i and R_i are positive definite, then the optimal solution of the SDP*

$$\begin{aligned} &\underset{\Gamma_i,\Pi_i}{min} \quad tr(\Gamma_i) \\ &s.t. \quad (4.14), \qquad (4.15), \qquad \Gamma_i,\ \Pi_i \succcurlyeq 0. \end{aligned} \tag{4.16}$$

is such that $\Pi_i^* = \tilde{P}_{i,k}$ is a solution of the ARE (4.12) and $\Gamma_i^* = (\Pi_i^*)^{-1}$.

Proof. First we show that Slater's condition (Balakrishnan and Vandenberghe, 1995) is satisfied on the dual problem. The dual of this SDP is:

$$\begin{aligned} \underset{Z,\,Y}{\text{minimize}} \quad & \operatorname{tr}\left(Z\mathcal{M} + Y \begin{bmatrix} \Gamma_i & 0 \\ 0 & \Pi_i \end{bmatrix}\right) \\ \text{subject to} \quad & -Z_{11} + Z_{22} + Y_{22} = 0, \\ & Y_{11} = I, \\ & Z,\ Y \succcurlyeq 0, \end{aligned} \tag{4.17}$$

where

$$Z = \begin{bmatrix} Z_{11} & Z_{12} \\ Z_{12}^T & Z_{22} \end{bmatrix},\ Y = \begin{bmatrix} Y_{11} & Y_{12} \\ Y_{12}^T & Y_{22} \end{bmatrix},$$

$$\mathcal{M} = \begin{bmatrix} \tilde{Q}_i^{-1} + \tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} & \tilde{Q}_i^{-1}\tilde{A}_i \\ \tilde{A}_i^T \tilde{Q}_i^{-1} & \tilde{A}_i^T \tilde{Q}_i^{-1} \tilde{A}_i \end{bmatrix}.$$

If we pick $Y_{12} = 0$, $Y_{11} = 0$ and $Z_{11} \succ Z_{22} \succ 0$, then $Z \succ 0$ and $Y \succ 0$ (i.e. strictly positive definite), and Y and Z are feasible, therefore the dual is strictly feasible. On top of that, given that the system is observable and Q_i and R_i are full rank, then (4.12) has a unique positive definite solution (Bittanti et al., 2012). Therefore, the primal is feasible since $\Pi_i = \tilde{P}_{i,k}^{-1}$ is a solution of the primal. Thus, strong duality and complementary slackness hold. Now, using complementary slackness, we know that in an optimal solution,

$$\begin{bmatrix} Y_{11}^* & Y_{12}^* \\ (Y_{12}^*)^T & Y_{22} \end{bmatrix} \begin{bmatrix} \Gamma_i^* & I \\ I & \Pi_i^* \end{bmatrix} = 0, \tag{4.18}$$

which implies that, given that $Y_{11}^* = I$, $\Gamma_i^* = (\Pi_i^*)^{-1} \succ 0$ and that $Y_{22}^* = (\Gamma_i^*)^T \Gamma_i^*$.

Therefore, since $Z_{11} = Z_{22} + Y_{22}$, we know that $Z_{11}^* \succ 0$. Without loss of generality, Z can be expressed as:

$$Z = \begin{bmatrix} I \\ K^T \end{bmatrix} Z_{11} \begin{bmatrix} I & K \end{bmatrix}$$

and, given that $Z_{11}^* \succ 0$ complementary slackness on (4.14) also implies that

$$\begin{bmatrix} I & K^* \end{bmatrix} \begin{bmatrix} \tilde{Q}_i^{-1} - \Pi_i^* + \tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} & \tilde{Q}_i^{-1} \tilde{A}_i \\ \tilde{A}_i^T \tilde{Q}_i^{-1} & \Pi_i^* + \tilde{A}_i^T \tilde{Q}_i^{-1} \tilde{A}_i \end{bmatrix} = 0 \tag{4.19}$$

which yields

$$\tilde{Q}_i^{-1} - \Pi_i^* + \tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} - \tilde{Q}_i^{-1} \tilde{A}_i (\Pi_{i,k}^* + \tilde{A}_i^T \tilde{Q}_i^{-1} \tilde{A}_i)^{-1} \tilde{A}_i^T \tilde{Q}_i^{-1} = 0. \tag{4.20}$$

□

In the optimal solution of the SDP, Γ_i^* is the augmented covariance matrix, i.e. $\Gamma_i^* = \tilde{P}_{i,k}^{-1}$. Therefore, minimizing $\mathrm{tr}(\Gamma_i)$ for all the targets is equivalent to minimizing $\tau^{-1} \sum_{i=1}^{\tau} \mathrm{tr}(\bar{\Sigma}_i(k))$, which is the optimization objective in the persistent monitoring problem, as expressed in (4.4). Thus, solving (4.16) gives us the squared estimation error of a single target, given an agent trajectory. Although Prop. 12 is not directly used to solve the Persistent Monitoring problem, it gives important insight into Prop. 13, which is the main result of this chapter.

4.3 Optimization of Persistent Monitoring Schedules

In order to obtain an efficient persistent monitoring schedule, one has to design an agent trajectory that will lead to low uncertainty in the estimation. Therefore, in this section, we give a procedure to jointly optimize the steady state uncertainty (4.4) and the trajectory of the agent. If we knew in advance when $\|s(k) - x_i\|$ was larger than r_i, then (4.3) would be linear with $d_i^2(k)$ at every time for every i, and the problem would be an SDP. However, since we do not know whether or not $\|s(k) - x_i\|$ is larger than r_i, a set of SDPs needs to be solved in order obtain the optimal trajectory with

that period. Therefore, we propose in this section a two-step procedure, where in the higher level, an algorithm produces sequences of targets to be visited by the agent and determines which of the modes of (4.3) is active in each time step. The lower level, on the other hand, assumes a fixed mode in (4.3) given by the higher level algorithm and through the solution of an SDP produces a trajectory that minimizes the steady state uncertainty.

Lower Level Problem

Recalling (4.3), in order to simplify notation on the rest of this subsection, we define $G_i = H_{\Delta,i,\max}^T R_{\Delta,i} H_{\Delta,i,\max}$ and its augmented version $\tilde{G}_i = diag(G_i, ..., G_i)$. Note that $H_{\Delta,i,k}^T R_{\Delta,i} H_{\Delta,i,k} = \gamma_{\Delta,i}(k) G_i$ and in the optimization $\gamma_{\Delta,i}(k)$ will be treated as an optimization variable and G_i as a constant. Therefore,

$$\tilde{H}_{i,k}^T \tilde{R}_i \tilde{H}_{i,k} = \underbrace{\begin{bmatrix} \gamma_{\Delta,i}(k) I_{L_i \times L_i} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \gamma_{\Delta,i}(k+\tau) I_{L_i \times L_i} \end{bmatrix}}_{\tilde{\gamma}_{i,k}} \tilde{G}. \tag{4.21}$$

Moreover, we create variables d_i^2 such that $d_i^2(k) \geq \|s_\Delta(k) - x_i\|^2$. The underlying goal of creating this variable is that the constraint will be binding in an optimal solution, i.e., $d_i^2(k) = \|s_\Delta(k) - x_i\|^2$, therefore we can compute $\gamma_{\Delta,i}(k)$ using (4.3) once $d_i^2(k)$ is fixed. Finally, we define logical variables $b_{i,k} \in \{0, 1\}$ (that will be fixed pre-defined variables in the optimization) as

$$b_{i,j} = \begin{cases} 0, & \|s_\Delta(k) - x_i\| > r_i, \\ 1, & \|s_\Delta(k) - x_i\| \leq r_i. \end{cases} \tag{4.22}$$

These logical variables represent whether or not the agent visits a given target i (i.e., the target within the agent's sensing range) on time step k of the cycle and thus

define the mode in (4.3). With that in mind, we state Prop. 13.

Proposition 13. *For fixed values of cycle length τ and logical variables $b_{i,k}$, the solution of the optimization* (4.23)*, when it exists, minimizes the cost in* (4.4)*, and the optimal trajectory $s^*_\Delta(\cdot)$ satisfies dynamic constraints* (4.2)*.*

$$
\begin{aligned}
\underset{\Gamma_i,\,\Pi_i,\,s_\Delta(\cdot)}{\text{minimize}} \quad & \frac{1}{\tau}\sum_{i=1}^{N} tr(\Gamma_i) \\
\text{subject to} \quad & \begin{bmatrix} \tilde{Q}_i^{-1} - \Pi_i + \tilde{\gamma}_{i,k}\tilde{G}_i & \tilde{Q}_i^{-1}\tilde{A}_i \\ \tilde{A}_i^T\tilde{Q}_i^{-1} & P_i + \tilde{A}_i^T\tilde{Q}_i^{-1}\tilde{A}_i \end{bmatrix} \succcurlyeq 0, \\
& \begin{bmatrix} \Gamma_i & I \\ I & P_i \end{bmatrix} \succcurlyeq 0, \\
& \Gamma_i,\ P_i \succcurlyeq 0, \\
& \|s_\Delta(k+1) - s_\Delta(k)\|^2 \le u_{\max}^2, \\
& \|s_\Delta(\tau) - s_\Delta(1)\|^2 \le u_{\max}^2, \\
& d_i^2(k) \ge r_i^2, && \text{if } b_{i,k} = 0, \\
& \gamma_i(k) = 0, && \text{if } b_{i,k} = 0, \\
& d_i^2(k) \le r_i^2, && \text{if } b_{i,k} = 1, \\
& \gamma_{\Delta,i}(k) = 1 - \frac{d_i^2(k)}{r_i^2}, && \text{if } b_{i,k} = 1, \\
& \forall i \in \{1,...,N\},\ \forall k \in \{1,...,\tau\}.
\end{aligned}
\tag{4.23}
$$

Proof. We only provide a sketch of the proof due to its similarity to Prop. 12. Using complementary slackness, we conclude that, if the optimal solution of SDP (4.23) is bounded, then when $b_{i,k} = 1$, in the optimal solution $d_i^2(k) = |s(k) - x_i|_2^2$ and therefore $\gamma_i(k) = 1 - (|s(k) - x_i|_2^2)/r_i^2$. Moreover, using the same arguments as in Prop. 12, we conclude that Π_i in an optimal solution of each feasible subproblem is a solution of the information filter Riccati equation (4.12) and that $\Gamma_i = \Pi_i^{-1}$, which means that Γ_i is the covariance matrix. Note also that the optimization objective ensures minimization of the infinite horizon cost defined as in (4.4) and the constraints ensure that feasible trajectories are periodic and satisfy dynamics constraints (4.2). □

Some brief insights on (4.23) are that the first three constraints are used in the solution of the ARE, similar to Prop. 12. The following two constraints impose the agent movement constraints (4.2) and periodicity. The next one is used to compute

the distance between the agent and the target and the last four constraints compute γ_i based on the relative position of the agent and the target.

Higher Level Problem

Using the optimization problem (4.23), we have a procedure such that, for each cycle period τ, we can solve an exponentially growing ($2^{N\times\tau}$) number of SDPs, representing different variations of $b_{i,k}$, and obtain the optimal solution for that cycle period. This approach is very inefficient due to the exponential scaling. We thus propose a graph-based scheme that explores different combinations of τ and $b_{i,k}$ by exploring how targets are spatially distributed and evaluates each combination using the low level optimization (4.23).

Remark 7. *While the idea of developing a higher level scheme that handles the exploration of visiting sequences was already explored in Chap. 3, this discrete time formulation has a fundamental difference. There, one assumed that, given a visiting sequence, we had an algorithm that could compute the dwell times in a non-combinatorial manner. Note that in the discrete-time case, one could think of visiting the same target for multiple consecutive time steps as the equivalent to dwelling at a target. However, even computing the optimal number of steps to dwell is a combinatorial problem here. Therefore, heuristics needs to be redesigned in order to correctly approach this problem.*

As a motivation for the higher level algorithm we propose, consider the case where two targets are far enough apart that the agent seeing one target at a time instant cannot see the other one in the following time step due to the constraints in the dynamics. In a "blind" exploration of variables $b_{i,k}$, one could encode the possibility of the agent visiting these targets at consecutive time steps and such choice of $b_{i,k}$ renders an infeasible solution of (4.23). The goal of the algorithm we introduce in this section is to evaluate only sets of τ and $b_{i,k}$ that can produce feasible trajectories according to the dynamics and the constraints as in (4.2). This algorithm is a "brute

force" approach and the exploration of more efficient exploration schemes are left to future work (e.g. extend the greedy exploration scheme in Chap. 3 for more general scenarios). We abstract the targets as nodes in a graph $\mathcal{G}$. The goal is to find a sequence of nodes to be visited. The cost $\xi(i,h)$ of each edge (i,h) in the graph $\mathcal{G}$ is the minimum number of time steps necessary for the agent to transition between visiting these two targets, i. e.,

$$\xi(i,h) = \max\left(1, \left\lceil \frac{\|x_i - x_h\| - r_i - r_h}{u_{\max}} \right\rceil\right). \tag{4.24}$$

Note that if an agent is visiting a given target, it can visit the same target in the following time step. Therefore, the self-transition cost is such that $\xi(i,i) = 1$, for any target. Given this structure, we can directly translate any sequence of visited nodes $\mathcal{S} = \{n_1, ..., n_F\}$ to the number of time steps in a cycle, τ, and to $b_{i,j}$, where

$$\tau(\mathcal{S}) = \xi(n_F, n_1) + \sum_{j=1}^{F-1} \xi(n_j, n_{j+1}). \tag{4.25}$$

For example, given two nodes in a 2D environment at positions $x_1 = (0,0)$ and $x_2(0,1)$ with $u_{\max} = 0.3$ and $r_j = 0.3$, then $\xi(1,2) = \xi(2,1) = 2$ and $\xi(1,1) = \xi(2,2) = 1$. A visiting sequence $\mathcal{S} = \{1,1,2\}$ would correspond to $\tau(\mathcal{S}) = \xi_{2,1} + \xi_{1,1} + \xi_{1,2} = 5$ and $(b_{1,1}, ..., b_{1,5}) = (1,1,0,0,0)$ and $(b_{2,1}, ..., b_{2,5}) = (0,0,0,1,0)$.

We then propose Algorithm 10, which combines both stages and we name this *SDP-PM*. The intuition behind this algorithm is that, initially, all the cycles in which each target is visited for exactly one time step (i.e. cyclic permutations of the targets) are added to a list and ordered according to the number of time steps in that particular cycle. Then, these cycles are explored in order. In the exploration, the cost of that particular cycle is evaluated and all the possibilities of visiting one new (or the same) targets in that cycle are added to the list $\mathcal{L}$. One thing to note is that the first cycle to be explored will always be the traveling salesman

optimal solution and lower length cycles will always be explored first. As an illus-

Algorithm 10 SDP-PM

1: $OptCost \leftarrow \infty$
2: $OptCycle \leftarrow \emptyset$
3: $\mathcal{L} \leftarrow \emptyset$
4: **for** $\mathcal{S} \in permutation(1, ...N)$ **do**
5: Add $(S, \tau(\mathcal{S}))$ to $\mathcal{L}$.
6: **for** $i \in (1, ..., N_{iter})$ **do**
7: $\mathcal{S} \leftarrow removeFirst(\mathcal{L})$
8: $cost = lowerLevelOptimization(\mathcal{S})$
9: **if** $cost < OptCost$ **then**
10: $OptCost \leftarrow cost$
11: $OptCycle \leftarrow \mathcal{S}$
12: **for** $newVertex \in (1, ..., N)$ **do**
13: **for** $p \in (2, ...N_{\mathcal{S}})$ **do**
14: $\mathcal{S}_{new} \leftarrow \{\mathcal{S}_{1:p-1}, newVertex, \mathcal{S}_{p:F_{\mathcal{S}}}\}$
15: Add $(S_{new}, \tau(\mathcal{S}_{new}))$ to $\mathcal{L}$
16: **return** $OptCost, OptCycle$

tration of Alg. 10, consider the very simple graph shown in Fig. 4·1. The list is initialized with all the possible cyclic permutations (lines 3-4) of the targets, i.e., $\mathcal{L} = \{(\{1, 2, 3\}, 12), (\{1, 3, 2\}, 12)\}$, where each element of the list is composed by a sequence of targets and the cycle length in that sequence, computed as in (4.25). Then, the first node on the list $(\{1, 2, 3\}, 12)$ is deleted from the list, its cost is evaluated, and all the cycles that can be constructed from this element by adding one extra visit are added to $\mathcal{L}$. Therefore, when exploring this first element, the cycles added to the list are $(\{1, 1, 2, 3\}, 13)$, $(\{1, 2, 2, 3\}, 13)$, $(\{1, 3, 2, 3\}, 18)$, $(\{1, 2, 1, 3\}, 14)$, $(\{1, 2, 2, 3\}, 13)$, $(\{1, 2, 3, 3\}, 13)$, $(\{1, 2, 3, 1\}, 13)$, $(\{1, 2, 3, 2\}, 16)$, and $(\{1, 2, 3, 3\}, 13)$. However, the list $\mathcal{L}$ is ordered according to cycle length, which means that the next cycle to be explored would be the one with length 12, followed by any cycle with length 13.

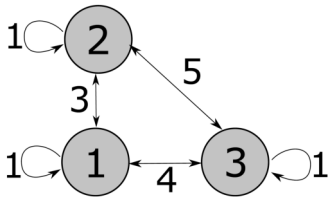


Figure 4·1: Example graph for illustrating Alg. 10.

4.4 Simulation Results

We implemented the SDP-PM Alg. 10 with dynamics as in (4.1) with parameters

$$A_{\Delta,i} = \text{diag}(1.1, 1.1), \quad Q_{\Delta,i} = \text{diag}(0.1, 0.1), \tag{4.26}$$

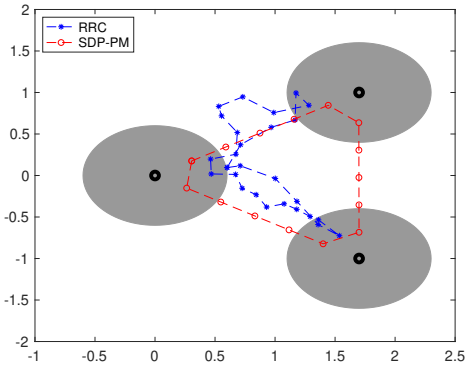
and observation model as in (4.3) with parameters

$$H_{\max} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}, \quad R_{\Delta,i} = \text{diag}(1, 1), \quad r_i = 0.6. \tag{4.27}$$

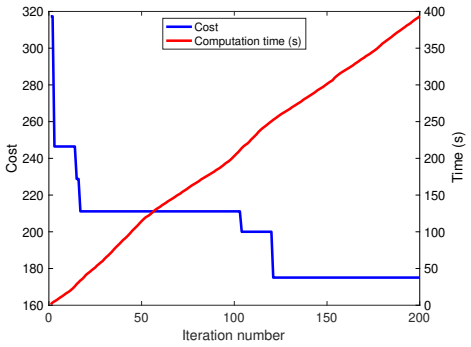
The agent maximum displacement in one time step is bounded by $u_{\max} = 0.33$. The SDP optimization was implemented in MATLAB using YALMIP (Lofberg, 2004) and solved using MOSEK (MOSEK ApS, 2019).

To the best of our knowledge, the only approaches proposed in the scientific literature similar enough to be used as a comparison are *RRC* and its variant *RRC** (Lan and Schwager, 2016). We implemented *RRC* and compared it to our approach (SDP-PM) in a simple environment, The results are shown in Fig. 4·2. Due to the random nature of RRCs, we ran it five independent times and we show its best, worst and average performances.

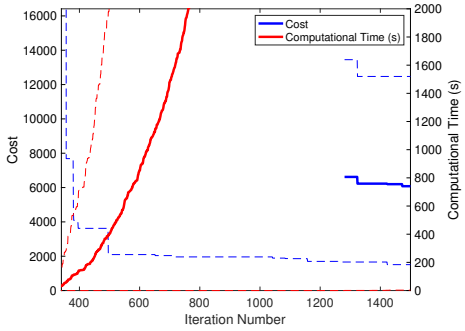
In fig. 4·2a, one can see that the trajectory produced by *SDP-PM* travels between targets in a straight line, while *RRC* does not. Also, when the agent visits a target in the *SDP-PM* trajectory, it always moves as close as possible to the center of the target (given total time steps and speed limitations), which does not happen in *RRC*. The



(a) Trajectories using the SDP-PM and RRC algorithms.



(b) Cost and computation time for SDP-PM.



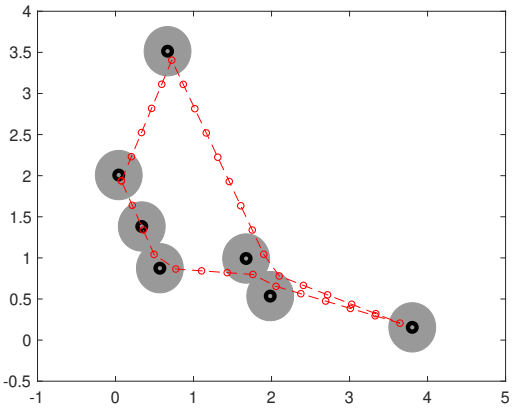
(c) Cost and computation time for RRC.

Figure 4·2: Results of the simulation with three targets. (a) Comparison of the trajectories generated by the *RRC* and the *SDP-PM* approaches. The trajectory displayed for RRC is the one with lowest cost among 5 independent runs of the algorithm. The presented trajectory was obtained after 200 iterations of the SDP-PM algorithm and 1500 iterations of *RRC*. The grey area represents the positions for which the agent can sense a given target. (b) Cost and cumulative computation time as a function of the iteration number for *SDP-PM*. (c) Cost and cumulative computation time as a function of the iterations of *RRC*. The solid lines represent average among 5 runs and the dashed lines are the observed maximum and minimum of the cost and computational time. None of the 5 instances of *SDP-PM* found a feasible solution before 345 iterations.

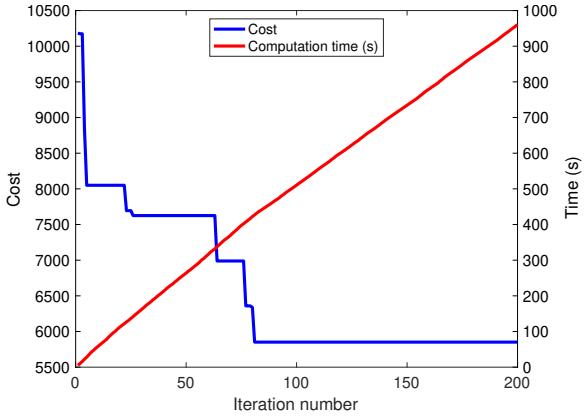
reason is that, for fixed τ and logical variables $b_{i,k}$, the trajectory generated by *SDP-PM* is optimal, while *RRC* only has an asymptotic probabilistic notion of optimality, with no deterministic guarantees for a finite number of iterations. Moreover, Figs. 4·2b and 4·2c show that, for reasonable computation times, *SDP-PM* produces much better solutions in terms of cost. The solution found at the first iteration of *SDP-PM* has the cost equal to 16.8% of the cost of the best (in terms of cost) of the 5 runs of *RRC* after 1500 iteration. Comparing our approach after 200 iterations and *RRC* after 1500, *SDP-PM* reports a cost of only 9% of the best solution of *RRC*. We also note that *SDP-PM* produced a solution with bounded cost in its first iteration, while *RRC* took between 345 and 1305 iterations to find its first feasible solution, i.e. where the target covariances are bounded.

In order to illustrate the performance of our approach in a more complex setup, we ran *SDP-PM* in an environment with 7 targets with their centers x_i randomly picked using a uniform distribution in $[0, 4] \times [0, 4]$. The systems parameters were the same as in the previous case, except that $r_{i,j}$ was set to 0.3. The results are displayed in Figs. 4·3a and 4·3b. We note that we ran *RRC* 5 times in this environment, with 10^4 iterations in each trial, and in none of these did *RRC* find a feasible solution. By analyzing *SDP-PM* results in this more complex environment, we can see that similar to the simpler environment, *SDP-PM* finds a feasible solution very fast and refines it within the first few iterations. One interesting aspect of the trajectory generated is that the agent visits some targets for non-consecutive times. This highlights the fact that the approach we propose here does not only locally search around the initial trajectory we select (in this case, initially the TSP solution is the first to be evaluated, where each target is visited once), but also is able to explore trajectories that have major changes in the visiting order compared to the initial exploration schedule. This gives rise to more complex behaviors, such as some targets being explored once, other

targets at multiple consecutive time steps, and also targets being visited multiple times but at non consecutive instants.



(a) Trajectory of the agent after 200 iterations of SDP-PM at more complex scenario. The circular red marks represent the positions where the agent was at the discrete time steps.



(b) Evolution of optimization cost and computational time using SDP-PM in the scenario with 7 agents.

Figure 4·3: Simulation results in the setting containing 7 targets

4.4.1 Discussion

In this chapter, we studied the discrete time version of the persistent monitoring problem. Our approach to solving it consisted of separating the problem in its combinatorial and its convex parts, and then approaching them separately, using a lower level and a higher level control optimization scheme. Similar ideas were already explored in Chap. 3, where the problems of computing visiting and dwelling sequences were approached almost independently.

Note, however, as stressed already in Remark 7, there are substantial differences between this discrete time approach and the continuous time one in Chap. 3. The main one is that the lower level optimization in this Chapter cannot decide what is the optimal number of time steps that the agent should spend visiting a given target. This makes the combinatorial exploration problem much harder compared to

the one in Chap.˜3. Additionally, although our approach was able to outperform the algorithm proposed in (Lan and Schwager, 2016), its lower level controller still faces significant challenges. First, even using state of the art SDP solvers (that employ polynomial-time interior point algorithms in their solutions), its computational cost is much higher than computing an optimal dwelling time allocation using algorithm 5, considering a similar number of targets. Moreover, this SDP approach presents numerical issues for even medium sized-problems (i.e. around 100 time steps and 10 targets). Meanwhile, the algorithm in 5 has been robust to the number of targets in all the simulations setting we have explored.

With this in mind, it is natural to believe that in most scenarios (especially when the sensing rate is fast compared to the movement dynamics) the approach in Chap. 3 is more adequate for modelling the problem. The hybrid nature of the model allows for exploiting properties that handle a much more robust and efficient scheme for dwelling sequence optimization, and makes it preferable in the vast majority of scenarios.

Chapter 5

Application to Multiple Particle Tracking

While the previous chapters were aimed at obtaining efficient algorithms for solving the persistent monitoring problem, in the present chapter the goal is to use (and adapt) those algorithms for a concrete real-world application: tracking multiple particles using a feedback-driven confocal microscope. The persistent monitoring approach used in this chapter is the one described in Chapter 3, since the fast dynamics of the systems (and the need for online replanning) make all the other approaches we described computationally prohibitive. The results in this section have previously appeared in (Pinto et al., 2021b).

5.1 Brief Background on Multiple Particles Tracking

In the study of cellular biological systems, it is necessary to track and identify components of the cell such as enzymes, RNA, molecular machinery, and viral pathogens. One generically refers to these as "particles". One set of methods for studying biology at these length scales are a group of techniques collectively referred to as single particle tracking (SPT) (Manzo and Garcia-Parajo, 2015; Shen et al., 2017). In these methods, particles of interest are smaller than the diffraction limit of light but can be visualized by labeling them with a fluorescent tag, making them visible to a fluorescence microscope (Mondal and Diaspro, 2013). By tracking a particle over time, one can understand the particle's motion model and the value of its motion parameters. Additionally one can directly observe the particle's behavior and interactions within

the cellular environment.

The application of SPT to the study of cellular biology has led to many breakthroughs in our understanding of subcellular processes. A few specific examples include measuring the behavior and roles of molecular motors (Kural et al., 2005), observing protein complex behavior in membrane localized processes (Aguet et al., 2016), and discovering viral infection pathways (Brandenburg and Zhuang, 2007). In each of these cases, the transport of the target particles can only be interpreted with reference to the cellular context. In many cases, this context includes observing the transport of multiple particles simultaneously.

Tracking multiple particles in a fluorescence microscope can be achieved in a variety of ways. One common method is to acquire images with a laser scanning microscope, such as a confocal fluorescence microscope or two-photon fluorescence microscope. These microscopes work by scanning a laser through a raster pattern to create an image. Unfortunately, the time to acquire a single image grows with the number of pixels in the image, making large area or high resolution images too slow for imaging dynamic particles. A paradigm change happened when feedback began to be used to control the laser position to track a single particle continuously without spending time away from the particle (Enderlein, 2000). Previous work by one of the authors implemented this concept using an extremum seeking feedback formulation (Andersson, 2011). While there are benefits in terms of speed and resolution, the main limitation of feedback methods is that only one particle at a time can be tracked.

The need to track multiple particles motivated the development of more efficient (non-raster scanned) multiple particle tracking (MPT) methods. Earlier work extended the single particle feedback methods by tracking each particle individually and switching between particles at a constant rate (Shen and Andersson, 2009). However, issues still remain such as the lack of a process to design efficient switching rules, the

inability to simultaneously handle particles with different diffusion coefficients, and the challenge of collecting intensity signals that optimize the collected information in order to improve the particle position estimation performance. The contribution of this chapter is to demonstrate a multiple particle tracking method that addresses these issues by combining a feedback driven tracking method with ideas from Chapter 3. The feedback scheme is drawn from the extremum seeking (ES) approach, introduced in (Ashley et al., 2016), where the trajectory of the laser is adapted based on the detected fluorescence. In particular, as will be discussed later in this chapter, the ES controller can converge to a maximally informative trajectory, i.e., a limit cycle where the acquired information optimal in terms of contributing to the estimation process.

By formulating this as a persistent monitoring problem, the system is able to autonomously decide which particle should be tracked at each instant in a way that minimizes the overall estimation uncertainty. We integrate PM to the context of multiple particle tracking, where the SPT algorithm plays the role of "data collection" and PM is responsible for deciding the optimal time to switch from tracking one particle to another.

5.2 Problem Statement

In this section, we formally define the problem of tracking multiple particles with a laser scanning microscope. The goal of tracking these particles is to estimate their positions over time in the cellular context so that one can identify their motion model, as well as the values of the parameters that define the motion model accurately and precisely. For simplicity of exposition, we assume the dynamics of the particles of interest are given by a Brownian motion process, described in continuous time as

$$\dot{X}_i[k] = W_i[k], \qquad W_i[t] \sim \mathcal{N}(0, Q_i). \tag{5.1}$$

In this equation, X_i is a random variable taking values in $\mathbb{R}^2$ which represents the x and y location of particle i, where $i = 1, ..., N$, and N is the number of tracked particles. $W_i[k]$ is a zero mean white noise with covariance matrix $Q_i = \text{diag}(2D_i, 2D_i)$, where D_i is the i-th particle diffusion coefficient,.

Photon detection is a Poisson random process called shot noise. The mean detected photon rate, I_i, for the fluorescent signal of a single particle at X_i and excited by a tightly focused leaser beam centered at X_l is given by:

$$I_i = I_{0,i} \exp\left(-2\frac{||X_i - X_l||^2}{b^2}\right), \tag{5.2}$$

where b is the laser beam width and $I_{0,i}$ is particle i's peak mean detected photon rate, i.e. it is the mean intensity when the laser is positioned exactly above particle i. Given a sampling period T_s, the total mean detected intensity I for an integration time is the sum of each particle's contribution, given by

$$I = \sum_{i=1}^{N} I_i T_s. \tag{5.3}$$

We assume that can directly control the laser position velocity (i.e., $\dot{X}_l$ is the control input), and the laser maximum velocity is upper bounded by $v_{\max}$. Our goal is to control the laser position X_l such that the detected intensity signals can be used to efficiently estimate the particle trajectories. In other words, we want to define an online control strategy where the laser position is updated using some feedback law that aims to minimize the estimation error of each particle i, given by

$$\sum_{k=1}^{N_s} E\left[||\hat{X}_i(kT_s) - X_i(kT_s)||^2\right], \tag{5.4}$$

where N_s is the total number of samples collected, and $\hat{X}_i$ is estimated position of particle i generated using an offline estimator. Note that even though we use feedback

while capturing the intensity data, the estimates of the particle positions are not necessarily computed simultaneously with the data acquisition process. Estimation can then be done offline. As a result there are no strict computational time constraints and the estimation can benefit from the entire dataset (as opposed to online methods, where the estimator must be causal). While the goal of this chapter is the efficient *acquisition* of informative measurements, we do apply an offline estimator in our simulation results in Sec. 5.5 to help illustrate the results. However, a detailed study of estimator design is out of the scope of this dissertation. Interested readers should see e.g. (Lin and Andersson, 2019; Ashley et al., 2016).

5.2.1 Proposed Solution

Our approach to this problem is to implement a two level control scheme. This allows us to divide the problem into two distinct parts, *measurement and tracking* of single particles (low level control), and *planning* which particle to measure and the duration of the measurement (high level control), which uses the PM algorithm. One key assumption that enables this scheme is that the particles are separated enough so that detected photons comes from a single particle chosen by the high level controller. This assumption, while not always true in practice, is a common one in SPT and it allows us to approach the multiple particle tracking problem as being constituted of tracking individual particles sequentially and then cycling the laser between them. Extensions to denser collections of particles is left to future work. The low level control will be discussed in Sec. 5.3 and the higher level one in Sec. 5.4. Algorithm 11 describes precisely how the integration between the two controllers is done.

In Alg. 11, $\hat{X}_i^{\text{on}}$ is an online estimate of the position of particle i, which means we need an online estimator for the interface between the lower level and the higher level controllers. We highlight that this online estimate needs to be computationally cheap and causal, and is usually not the same estimate that will be used offline (i.e.

after all the data has been collected) to estimate the particle position with a high accuracy. In Alg. 11 the procedure ScheduleNextObservedTarget$(\cdot,\cdot)$ is what we call the "high level algorithm" and is based on PM.

Algorithm 11 Multiple Particle Tracking

1: **while** Experiment is Running **do**
2: $[\tau, j] \leftarrow$ ScheduleNextObservedTarget$(\hat{X}_i^{\text{on}}, i)$
3: Move laser to $\hat{X}_j^{\text{on}}$.
4: Run lower level control for τ seconds.
5: Update $\hat{X}_j^{\text{on}}$.
6: $i \leftarrow j$.

5.3 Extremum Seeking Single Particle Tracking

In this section, we discuss the lower level controller, responsible for tracking a single particle for some duration. Considering the goal of minimizing the estimation error in (5.4), one would like to design this controller such that the laser collects a photon signal that is maximally sensitive to small changes in particle position. In this context, we first analyze where is the best region to place the laser. We consider the random observation model with mean given by (5.2), and from it derive the Fisher Information Matrix (FIM) for estimating the particle position under the assumption of a fixed particle position. The FIM for a Poisson temporal random process is

$$\text{FIM}(X_i - X_l) = \int_{t_0}^{t_1} \frac{1}{I_i(X_i - X_l)} \frac{\partial I_i(X_i - X_l)}{\partial X_l} \frac{\partial I_i(X_i - X_l)}{\partial X_l}^T \text{dt}, \tag{5.5}$$

where the expected intensity I_i is given by (5.2) and $[t_0, t_1]$ is the time interval when particle i is being tracked (Snyder and Miller, 2012). We next apply the trace (T-Optimality) criteria to the FIM (Pukelsheim, 2006) to get the cost function

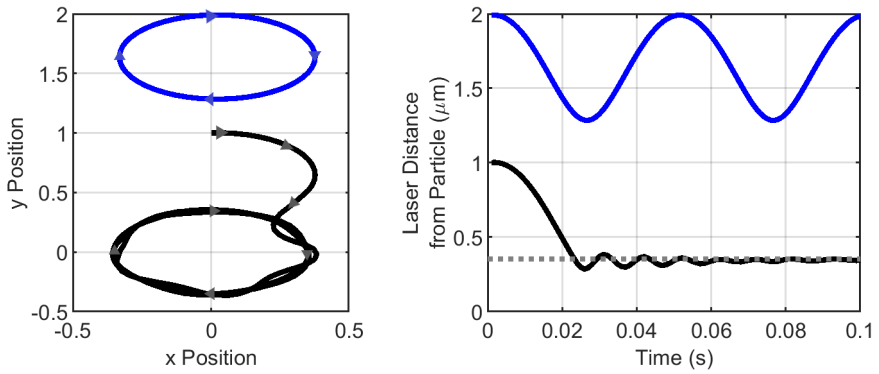
$$J(X_i - X_l) = \text{tr}\left(\text{FIM}(X_i - X_l)\right) = -\frac{16}{b^4} \int_{t_0}^{t_1} ||X_i - X_l||^2 I_i(X_i - X_l)\text{dt}. \tag{5.6}$$

Optimizing (5.6) gives that the laser positions that minimize the trace of the FIM are given by a circle centered on the particle position with a radius of $\frac{b}{\sqrt{2}}$ (Gallatin and Berglund, 2012). We denote this set of positions as the information optimal orbit. In practice, the assumption we made about the particle position being fixed is not true. However, this information optimal orbit provides a near optimal result as long as the speed of the laser is fast relative to the particle motion.

The next step is to determine how to move the laser to the information optimal orbit when the particle position is not known exactly. The search for a practical controller to track a single particle leads us to select an extremum seeking controller (ESC). Extremum seeking is a model free method that locally explores a scalar field and drives the system state towards an extremal point in the field. In the case of SPT, the scalar field is the expected amount of detected photons which has a maximum centered on the particle's location. The use of ESC allows for tracking using only the collected intensity data, without the need of an online estimation scheme. (Note that our complete tracking scheme described in Sec. 5.2.1 only needs an online estimate for the high level controller and, as will be discussed in Sec. 5.4, ESC can be used to provide this estimate.) A particular implementation of ESC that converges to an orbit around a field extremum and that has been shown to work well in SPT applications is that of a non-holonomic, reactive ESC (Andersson, 2011; Ashley and Andersson, 2016; Ashley and Andersson, 2017) given by

$$\begin{aligned} \Delta I &= I[k] - I[k-1], \\ \theta[k] &= \theta_0 + 2\pi f T - \frac{K_p}{T_s}\Delta I, \\ x_l[k] &= x_l[k-1] + 2\pi R f T_s \cos\theta[k-1], \\ y_l[k] &= y_l[k-1] + 2\pi R f T_s \sin\theta[k-1]. \end{aligned} \tag{5.7}$$

In these equations, f is its oscillation frequency, $\omega = 2\pi f$ is its angular frequency,



(a) Laser trajectory **(b)** Laser distance from particle

Figure 5·1: Simulation of Extremum Seeking Controller trajectories starting at two different positions showing failure to converge (blue), and convergence (black). The arrows indicate the movement direction.

T_s is the controller time step, and R is a positive constant. When $\Delta I = 0$, this controller imposes a circular orbit with radius R. The feedback term is responsible for guiding the system to an orbit centered at the extremum and radius R (Ashley and Andersson, 2017). Therefore, when we set $R = b/\sqrt{2}$, ESC gives us a practical method for converging to the information optimal orbit and enables us to adapt the orbit as the particle moves.

We characterized the behavior of this lower level controller empirically. Fig. 5·1 shows the behavior of the ESC from two initial conditions with the particle being tracked at the origin. If the initial conditions are too far away, the photon rate is not high enough to drive the ESC to the target on any practical timescale. If the initial conditions are close enough, the ESC converges to a cycle around the target. The convergence behavior of the particular extremum seeker controller (in the presence of source movement and observation disturbance) that we use here was formally analyzed in (Pinto and Andersson, 2021).

To illustrate the behavior of the convergence rate, we picked $K_p = 0.6$ and $\omega = 60$ Hz and plotted the number of cycles until convergence as a function of the starting

position within the trackable region. The results are shown in Fig. 5·2.

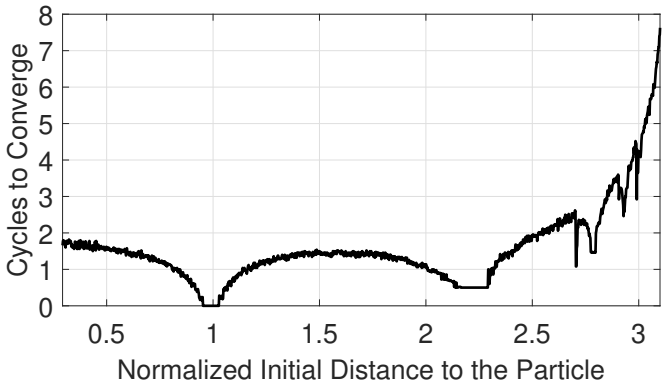


Figure 5·2: Number of cycles to convergence as a function of the initial relative position of the laser and particle. The initial distance is normalized by the radius R.

The ESC is defined by three parameters, ω, K_p, and R. The radius is determined by the information optimal orbit while the other two can be tuned to minimize the tracking error. Using Monte Carlo simulations (that consider shot noise and used a diffusion coefficient of $D = 0.1$), we picked $f = 60$ Hz. The mean squared tracking error as a function of K_p is given in Fig. 5·3. It is important to keep in mind that the specifics will depend strongly on the experimental conditions such as the diffusion coefficient and background noise.

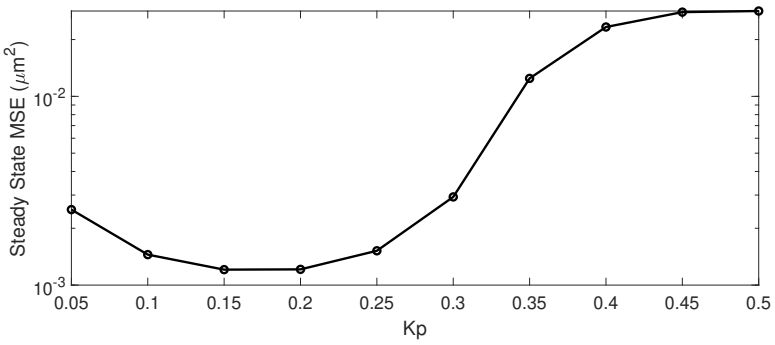


Figure 5·3: Mean squared tracking error as a function of K_p, for $f = 60$ Hz and particle diffusion coefficient $D_x = D_y = 0.1$ μm^2/s.

5.4 Scheduling Multiple Particle Tracking

With the low level controller of Sec. 5.3 in place, we now turn to the specifics of the persistent monitoring formulation we use. As described in Chapter 2, in the PM problem, we consider that the targets' internal state (in this case, the particle position) evolve with linear, time invariant stochastic dynamics. This assumption is true for the Brownian motion model used here. It it was also assumed that when the agent dwelt at a target, the target state could be observed with the linear, stochastic observation model given by (2.3).

Therefore, to deploy the PM algorithm given in Chapter 3, we need a simple online estimate that fits our estimation model (2.3). This is provided by ESC since, after convergence to the radius R, it produces unbiased observations Z according to the following relation:

$$Z = X_l + \begin{bmatrix} -R\sin\theta \\ R\cos\theta \end{bmatrix} = X_i + \tilde{V}_i, \tag{5.8}$$

where $\tilde{V}_i$ is a noise term.

With this estimator, the PM algorithm determines the sequence for visiting the particles and the time to spend at each particle. Intuitively, PM seeks to balance the time spent at each of the particles, trading off estimation accuracy at any given particle for performance over the entire collection of particles.

In the MPT setting, when the laser transitions to visit a given particle, it moves to the particle's last estimated position (as indicated in Alg. 11), as this is the most likely particle position and the expected intensity signal is higher when the particle is closer to the laser. The cost function (3.2) aims to minimize the worst case uncertainty on the particles' estimated position and this maximizes the chances that the observations acquire enough photons for ESC to converge to an orbit centered in the particle.

Finally, while in our PM formulation the targets (particles) were assumed to have a fixed position, here the particles move according to a Brownian motion model.

Table 5.1: Summary of differences between the MPT setting and the PM model, and the assumptions being used in order to apply PM to the MPT model.

MPT Setting	PM Model	Assumption
Moving particle.	Fixed target.	Particle movement is slow compared to the laser speed.
Non-linear observation model.	Linear observation model.	Extremum seeker provides online, linear, observable sensing model.
Time-varying graph structure (due to particle movement).	Fixed graph structure.	Online replanning assume fixed graph structure at each cycle.
Measurements can combine photons coming from different particles.	Targets are sensed independently.	Particles are spread away enough so that photons come from a single particle.

Therefore, here we rely on the assumption that their movement is slow compared to the laser speed and thus the PM algorithm produces near optimal schedules. However, this brings the need of replanning the PM schedule, to adapt it as particles move, leading to the structure of Algorihm 11.

The differences between the MPT setting and the PM model are highlighted in Table 5.1. There we also state the assumptions we made in order to justify applying PM techniques to this problem.

5.5 Simulation and Results

In this section, we provide a set of simulations (with three particles in each) with the goal of illustrating the performance of our proposed approach for tracking multiple particles. For these simulations, we used the Brownian motion model in (5.1) and the laser observation model in (5.2), with the parameters: $D_i = 0.1\ \mu\text{m}^2/\text{s}$, $I_{0,i} = 5 \times$

10^4 photons/s, $b = 0.5$ μm. The maximum laser speed was limited to $v_{\max} = 300$ μm/s and the simulation time-step was set to $T = 10^{-4}$ s. The extremum seeking oscillation frequency was set to $f = 60$ Hz, its gain to $k_p = 0.2$ and the radius to $R = b/\sqrt{2}$. The value of the covariance of the noise $\tilde{V}_i$ in the observation model was obtained using the simulated mean squared tracking error, given in Fig. 5·3. The particles' initial positions were drawn from a uniform distribution in $[0\ \mu\text{m}, 10\ \mu\text{m}] \times [0\ \mu\text{m}, 10\ \mu\text{m}]$. In the initialization, we assumed that the controller had access to the approximate initial position of the particles plus some zero mean Gaussian noise (with covariance equals to $\text{diag}(0.02^2, 0.02^2)$ μm^2) . In practical settings, this initial position could be obtained using, for example, a widefield image. The initial position of the laser was $[5\ \mu\text{m}, 5\ \mu\text{m}]$ and the total simulation time was 1 s.

To characterize the performance of the tracking algorithm, we analyzed the rate of collected photons. Note that although the number of collected photons does not directly translate into estimation performance (in particular the position where the photons were collected is also important), it is still a good proxy for evaluating tracking performance, since in general, increasing the number of collected photons increases the estimation performance.

The trajectory and the collected photons per sample for typical run of the simulation are shown in Figs. 5·4 and 5·5, respectively. The colors in the intensity figure match the particle from which those photons came from. Note that since the particles were widely spaced relative to the width of the laser, all photons collected at each time step were from a single particle. In this run, the mean collected photons rate (normalized by I_0) was 0.2905/sec.

To get a sense of the average performance of the tracking scheme, we ran 100 simulations with the same parameters, but with different initial positions and Brownian motion realizations. The average detected photons rate over these runs was

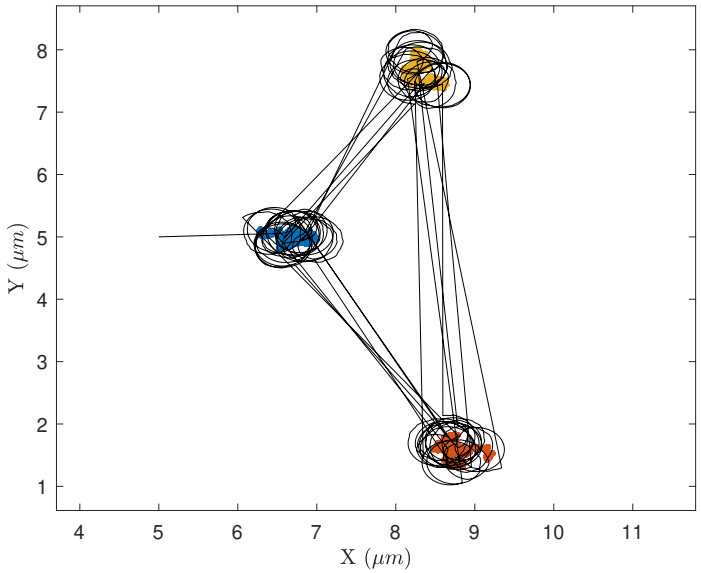


Figure 5·4: Particle and laser trajectories, while tracking three particles in the first simulation scenario. The laser trajectory is in black and the particles are in blue, yellow and red.

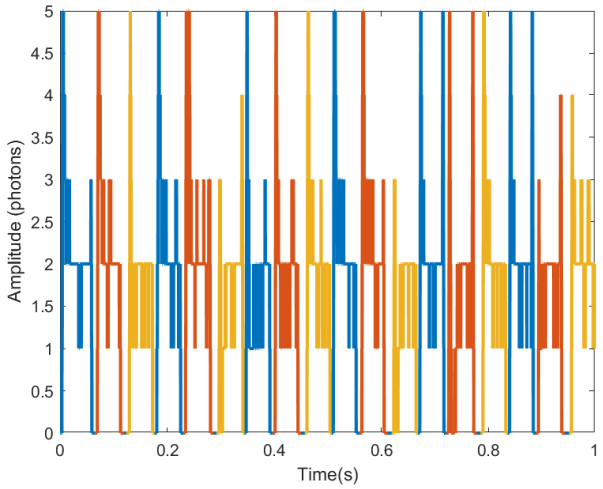


Figure 5·5: Photons collected at each time step ($T_s = 10^{-4}$ s) in the first simulation scenario. The colors in the plot indicate which particle emitted those photon.

$0.2958 \times I_0$.

While the ESC does not use any model information, the high-level PM planner uses prior knowledge of the process and observation noises. In practice, of course, these terms are at best known only approximately. In order to illustrate the performance of our tracking scheme to perturbations in the system parameters, we ran a set of simulation with almost the same setup as in the previously described scenarios, except that the values of the peak intensity and diffusion coefficient were modified to $I_0 = 4 \times 10^4$ photons/s and $D = 0.11\ \mu\text{m}^2/s$. The controller was not adjusted. In these simulations, the average detected photons rate per second was $0.2703 \times I_0$, indicating some drop in performance but some robustness to model uncertainty.

Finally, in order to give a sense of how our approach compares to a simple raster scan, we also used the intensity signal for a raster scan trajectory. In this setup, the laser moved along a zig-zag (raster) pattern with constant speed, equal to $v_{\max}$. The raster scan trajectory is shown in Fig. 5·6. Note that while the raster scan images a large region without any particles, this is normal to raster scanning as the region is set in open loop fashion. This simulation run yielded a normalized average photon rate of 0.0132.

Table 5.2 summarizes the results simulation using the setups above mentioned. Each setup was run 100 times, with random initial positions and diffusion noise realizations. The rate of acquired photons was consistent among the different simulation runs using our tracking method and much higher than when using a raster scan.

Table 5.2: Mean number of collected photons per second normalized by $I_{0,i}$ for 100 runs of each of the simulation setups.

Nominal params.	Perturbed params	Raster scan
0.2958 ± 0.0235	0.2703 ± 0.0795	0.0111 ± 0.0033

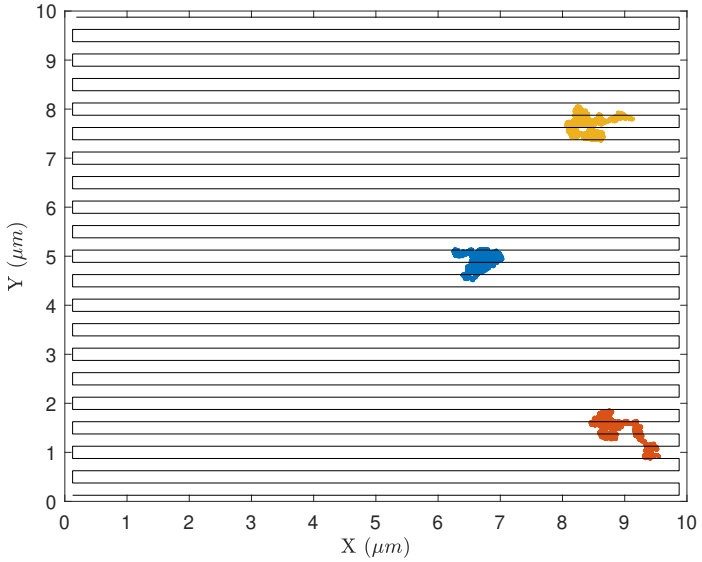


Figure 5·6: Illustration of a raster scanning trajectory considering a similar simulation setup. The agent trajectory is in black, while the particle trajectories are colored.

5.5.1 Trajectory estimation from photon data

Although this work focuses on data collection, the overall goal of MPT is to accurately estimate the particles' positions *offline*. To illustrate how this can be done using the intensity data from our simulations, we estimated the particles' positions by applying a Particle Filter and Rauch-Tung-Striebel Smoother (see e.g. (Lin and Andersson, 2019)). We note that we have not extensively explored different offline estimators and likely other approaches could yield estimation with lower errors. The mean estimation error over time in the first scenario is shown in Fig. 5·7. The RMSE was calculated using

$$RMSE = ||(\hat{X}_i - X_i)||. \tag{5.9}$$

RMSE results for the third simulation setting (with perturbed parameters) is shown in Fig. 5·8. In these plots, the shaded regions indicate times when an individual particle is being tracked. The RMSE of all runs and considering all simulation setups is given in Table 5.3. Our estimation algorithm was able to keep the average error at

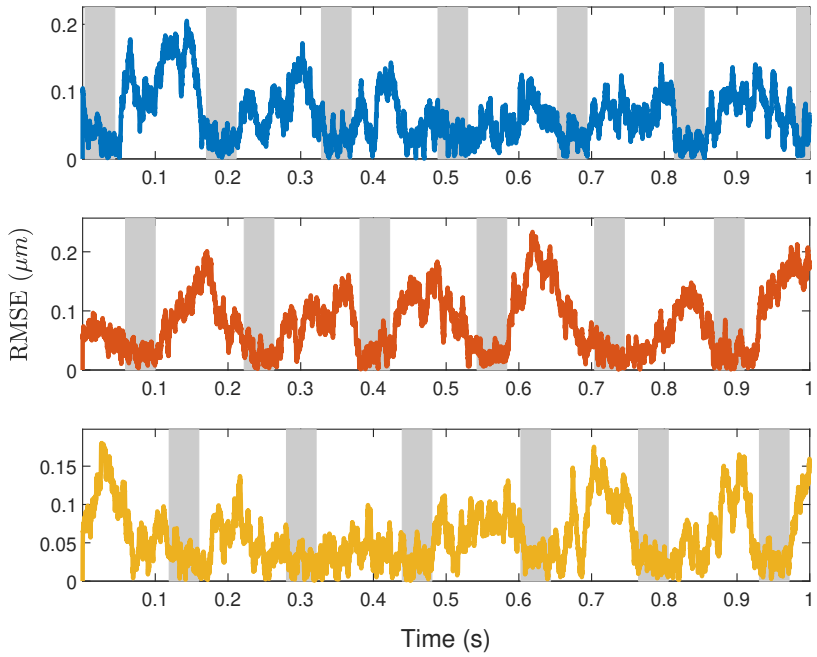


Figure 5·7: Estimation error over time using the offline estimator. The colors of the plots match the colors of the particle in Fig. 5·4. The shaded areas mark when the laser was orbiting around each particle.

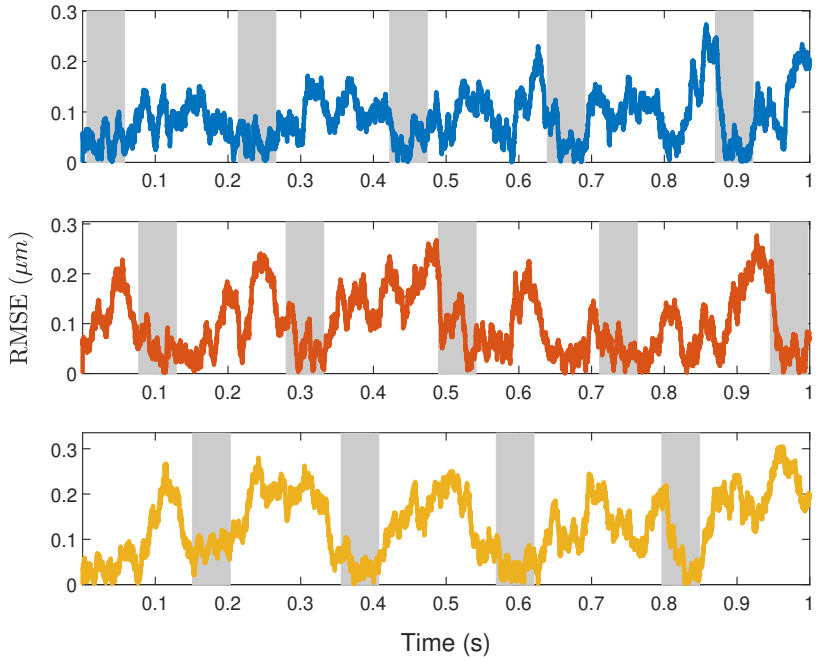


Figure 5·8: Estimation error over time with perturbed parameters. The shaded areas mark when the laser was orbiting around each particle.

Table 5.3: RMSE Estimation error of 100 simulation runs.

Nominal params.	Perturbed params
100.3 ± 38.1 nm	138.4 ± 66.3 nm

around 100 nm when the nominal parameters were used in this simulation. However, the mismatch in D_i $I_{0,i}$, generated a higher estimation error. In future work we plan to also estimate the model parameters along with the particles' positions, aiming to improve our estimation performance and robustness to modeling imperfections.

Chapter 6

Conclusions and Future Work

6.1 Conclusion

In this dissertation, we studied the problem of persistent monitoring of uncertain targets. In particular, our contributions include the creation of scalable and efficient ways to approach the version of the persistent monitoring model that assumes that the state to be tracked evolves with a linear dynamic model and can be observed with a linear observation model. We emphasized especially the advantages of analyzing the problem from a steady-state perspective, assuming periodic trajectories. In such an approach, the persistence of visits over infinite horizons was automatically enforced and the computational complexity of the approaches do not scale with time horizon, making it very suitable to long-term surveillance tasks. We showed that the computation of gradients in such case has a similar computational overhead compared to computing a short-horizon gradient in the transient version of the problem.

However, in many scenarios that demand faster update rates in the planned trajectories, even computing such gradients may be prohibitive. For such scenarios, we considered some simplifications to the problem to allow faster solutions (leading to the *minimax* approach) namely constraining the agent movement space to a graph and using a divide-and-conquer approach, where targets were constrained to only be visited by one agent. In this formulation, the problem reduces to separately computing an optimal visiting and dwelling sequence. We then developed a computationally lightweight algorithm, that exploits properties of an optimal solution rather than

explicitly solving the optimization using gradient-based techniques. By imposing additional assumptions to the problem, global optimality was guaranteed by using such algorithm. This ensures that our approach has some robustness to undesired local minima, unlike most of the previous works in the field.

We then explored the discrete-time version of the problem, which could be framed and solved efficiently using a mathematical-optimization based framework. However, due to its nature, the discrete time formulation cannot take advantage of the properties of an optimal solution, as we did in the *minimax* continuous-time formulation. In practice, this means our approach to the discrete time version cannot be solved as fast and robustly as the continuous time one, since it cannot directly take advantage of the hybrid structure of the graph-based version of the problem, and must resort to combinatorial optimization steps also when searching for the optimal dwelling time a given target.

Finally, we showed that the use of this formulation is indeed compelling in real world applications. By exploring the multiple particle tracking problem, we adapted the current formulation to be able to handle a real world problem, where the dynamics (especially the observation model) do not fully match the original model for which the algorithms were developed. We were able to efficiently plan a laser trajectory that could observe the diffusing particles with the desired rate and accuracy. The advantages of such method compared to simple raster scanning were very evident.

6.1.1 Future Work

Persistent Monitoring for Non-causal Estimation

In the persistent monitoring formulation studied in this dissertation, it was implicitly assumed that estimation was done in a causal manner. However, as demonstrated in the multiple particle tracking application, sometimes the goal is to design an efficient policy for acquiring data, and then only perform estimation offline. In such a

situation, there is no reason for considering only causal estimators.

Non-causal estimation can potentially greatly benefit the overall performance of the estimation process. However, the dynamics of the covariance in the Kalman-Bucy filter equations do not take into account the fact that "future" data can potentially be used for estimation. Thus, one possible direction for future research is to replace the covariance dynamics in order to also accommodate the potential of future data being used in the estimation. This modification might require reevaluating all the propositions related to convergence of the steady state covariance and its gradients computation.

Tracking multiple potentially cluttered particles using feedback driven confocal microscopy

In our study of application of PM to MPT, an important assumption was that particles were separated enough so that they could be individually observed using the extremum seeking controller. However, in a general setting, particles do not maintain a minimum separation, and the distance between particles usually varies substantially over time. Therefore, in order to apply PM in practical MPT settings, it is necessary to augment our approach to be able to handle cluttered particles.

One first idea towards that direction is to use the lower level (extremum seeker) controller to track small clusters of particles instead of individual ones. Towards that, it is necessary to design an algorithm that, based on online measurements, is able to either merge particles into a cluster when they are too close, or separate them into different ones as they move farther apart. With this dynamic clustering process, possibly PM could be used without drastic modifications. However, the question of whether the ESC will be able to track these particle clusters still has to be investigated in more depth. Otherwise, another low level controller has to be designed.

Appendix A

Conditions on the existence of gradients of the steady state covariance matrix

In Chapter 2, we discussed the computation of the steady state partial derivatives of the covariance matrix, however, the computation procedure was conditioned on their existence. In this appendix, we discuss the existence of such gradients. Note that, if in a periodic trajectory $\eta_i(q) = 0 \ \forall q \in [0,1]$ (i.e., target i is never visited), the existence of the steady state covariance matrix is not guaranteed by Prop. 2. Obviously, if the steady state covariance does not exist, its derivative will also not exist. This illustrates the fact that the existence of $\frac{\partial \bar{\Omega}_i}{\partial \theta}$ is not guaranteed. What we show in this appendix is that, under very natural assumptions, the derivative $\frac{\partial \bar{\Omega}_i}{\partial \theta}$ exists for the parameters that belong to the interior of the set of parameters that will lead to convergence of the steady state covariance, except for a set of zero measure.

Since here we analyze the behavior of the steady state covariance with respect to parameter variations, we will use a notation that explicitly shows the dependence of the variables with the parameters. For example, $\bar{\Omega}_i$ is a function of q and of the parameters Θ and, hence, it will be denoted as $\bar{\Omega}_i(q;\Theta)$.

We define the set of parameters for which the steady state covariance is guaranteed to exist as:

$$\vartheta = \{\Theta \mid \eta_i(q,\tilde{\Theta}) > 0 \text{ for some non-degenerate interval } q \in [a,b]\}, \tag{A.1}$$

and Ψ as the interior of the set ϑ.

Our goal is to show that, for any $\Theta \in \Psi$, the partial derivatives $\frac{\partial \bar{\Omega}_i(q;\Theta)}{\partial \theta_d}$ exist locally. From Prop. 3, we know that, when this partial derivative exists, it is equal to $\Sigma(q;\Theta)$. We also know that $\Sigma(q;\Theta)$ is well defined for any $\theta \in \Psi$. We now make the following assumption about the regularity of Σ:

Assumption 6. *$\Sigma(q;\Theta)$ is locally Riemann integrable with respect to Θ for $\Theta \in \Psi$ and $q \in [0,1]$.*

In light of Proposition 3, Assumption 6 means that the parameterizations that we consider do not allow for an infinite number of discontinuities of $\Sigma_h(q;\Theta)$ and $\Sigma_{ZI}(q;\theta)$. Note that, due to the linear nature of their underlying differential equations, $\Sigma_h(q;\Theta)$ and $\Sigma_{ZI}(q;\theta)$ are bounded for any $\Theta \in \Psi$. Therefore, $\Sigma(q;\Theta)$ is also bounded.

Proposition 14. *Under Assumptions 1, 2 and 6, the partial derivative $\frac{\partial \bar{\Omega}_i(q;\Theta)}{\partial \theta_d}$, $q \in [0,1]$ and $\Theta \in \Psi$, exists almost everywhere in $[0,1] \times \Psi$.*

Proof. By construction, we pick two parameter sets Θ_1 and Θ_2, such that any convex combination of Θ_1 and Θ_2 belongs to Ψ. Additionally, since our goal is to compute the partial derivative with respect to θ_d, we pick Θ_2 such that it differs from Θ_1 only in its d-th coordinate. Since the set Ψ is open, if we pick any $\Theta_1 \in \Psi$, we can always find a Θ_2 that fullfills the aforementioned properties.

We define the function $\Upsilon(q;\Theta_2)$ (which later we will show $\Upsilon(q;\Theta_2) = \bar{\Omega}_i(q;\Theta_2)$) as:

$$\Upsilon(q;\Theta_2) = \bar{\Omega}_i(q;\Theta_1) + \int_0^1 \Sigma(q;\Theta_1 + \xi(\Theta_2 - \Theta_1))d\xi. \tag{A.2}$$

Note that, if $\Upsilon(q;\Theta_2) = \bar{\Omega}_i(q;\Theta_2)$ for generic Θ_1, Θ_2, then $\Sigma(q;\Theta) = \frac{\partial \bar{\Omega}_i(q;\Theta)}{\partial \theta_d}$ almost everywhere, since $\Sigma(q;\Theta)$ plays the role of a partial derivative in Eq. (A.2).

$\bar{\Omega}_i(q;\Theta_2)$ is uniquely defined by satisfying the differential equation (2.31) and being periodic with period one. We then show that $\Upsilon(q,\Theta_2)$ also satisfies both of these properties, which imply that indeed $\Upsilon(q,\Theta_2) = \bar{\Omega}_i(q;\Theta_2)$.

First, notice that $\Upsilon(0;\Theta_2) = \Upsilon(1;\Theta_2)$ since $\bar{\Omega}_i(0;\Theta_1) = \bar{\Omega}_i(1;\Theta_1)$ and $\Sigma(0,\Theta) =$

$\Sigma(1,\Theta)$, for any $\Theta \in \Psi$. Also, since $\Sigma(q;\Theta)$ is a solution of (2.32),

$$\int_0^1 \dot{\Sigma}(q;\Theta_1 + \xi(\Theta_2 - \Theta_1))d\xi = \dot{\bar{\Omega}}_i(q,\Theta_2) - \dot{\bar{\Omega}}_i(q,\Theta_1). \tag{A.3}$$

Therefore, taking the derivative of (A.2) with respect to q and substituting (A.3), we get

$$\dot{\Upsilon}(q,\Theta_2) = \dot{\bar{\Omega}}_i(q,\Theta_2). \tag{A.4}$$

Hence we conclude that $\Upsilon(q,\Theta_2) = \bar{\Omega}_i(q;\Theta_2)$, and, as a consequence, $\frac{\partial \bar{\Omega}_i(q;\Theta)}{\partial \theta_d}$ exists almost everywhere in Ψ. Additionally, as already stated in Prop. 3, $\frac{\partial \bar{\Omega}_i(q;\Theta)}{\partial \theta_d} = \Sigma(q,\Theta)$ wherever it exists. □

Note that, as long $\Sigma(q,\Theta)$ is continuous with respect to Θ, the continuity of the derivatives is also guaranteed for $\Theta \in \Psi$ and $q \in [0,1]$.

References

Aguet, F., Upadhyayula, S., Gaudin, R., Chou, Y.-y., Cocucci, E., He, K., Chen, B.-C., Mosaliganti, K., Pasham, M., Skillern, W., et al. (2016). Membrane dynamics of dividing cells imaged by lattice light-sheet microscopy. *Molecular Biology of the Cell*, 27(22):3418–3435.

Alam, T., Reis, G. M., Bobadilla, L., and Smith, R. N. (2018). A Data-Driven Deployment Approach for Persistent Monitoring in Aquatic Environments. In *IEEE International Conference on Robotic Computing*, pages 147–154. 10.1109/JOE.2020.2999695.

Anderson, B. D. and Moore, J. B. (2012). *Optimal filtering*. Courier Corporation.

Andersson, S. B. (2011). A Nonlinear Controller for Three-dimensional Tracking of a Fluorescent Particle in a Confocal Microscope. *Applied Physics B*, 104(1):161–173.

Ashley, T. T. and Andersson, S. B. (2016). A Control Law for Seeking an Extremum of a Three-dimensional Scalar Potential Field. In *2016 American Control Conference*, pages 6103–6108. IEEE. DOI: 10.1109/ACC.2016.7526628.

Ashley, T. T. and Andersson, S. B. (2017). Tracking a Diffusing Three-dimensional Source via Nonholonomic Extremum Seeking. *IEEE Transactions on Automatic Control*, 63(9):2855–2866.

Ashley, T. T., Gan, E. L., Pan, J., and Andersson, S. B. (2016). Tracking Single Fluorescent Particles in Three Dimensions via Extremum Seeking. *Biomedical Optics Express*, 7(9):3355–3376.

Athans, M. and Tse, E. (1967). A Direct Derivation of the Optimal Linear Filter Using the Maximum Principle. *IEEE Transactions on Automatic Control*, 12(6):690–698.

Balakrishnan, V. and Vandenberghe, L. (1995). Connections Between Duality in Control Theory and Convex Optimization. In *1995 American Control Conference*, volume 6, pages 4030–4034. DOI: 10.1109/ACC.1995.532689.

Barraud, A. (1977). A Numerical Algorithm to Solve AT XA-X=Q. In *IEEE Conf. on Decision and Control and 16th Symposium on Adaptive Processes and Special Symposium on Fuzzy Set Theory and Applications*, pages 420–423. DOI: 10.1109/CDC.1977.271607.

Baykal, C., Rosman, G., Kotowick, K., Donahue, M., and Rus, D. (2020). Persistent Surveillance of Events with Unknown Rate Statistics. In *Algorithmic Foundations of Robotics XII*, pages 736–751. Springer.

Bektas, T. (2006). The Multiple Traveling Salesman Problem: an Overview of Formulations and Solution Procedures. *Omega*, 34(3):209 – 219.

Bittanti, S., Colaneri, P., and Guardabassi, G. (1984). Periodic Solutions of Periodic Riccati Equations. *IEEE Transactions on Automatic Control*, 29(7):665–667.

Bittanti, S., Laub, A. J., and Willems, J. C. (2012). *The Riccati Equation*. Springer Science & Business Media.

Brandenburg, B. and Zhuang, X. (2007). Virus trafficking–learning from single-virus tracking. *Nature Reviews Microbiology*, 5(3):197.

Bullo, F. (2020). *Lectures on Network Systems*. Kindle Direct Publishing, 1.4 edition. With contributions by J. Cortes, F. Dorfler, and S. Martinez.

Cassandras, C. G., Lin, X., and Ding, X. (2013). An Optimal Control Approach to the Multi-agent Persistent Monitoring Problem. *IEEE Transactions on Automatic Control*, 58(4):947–961.

Chen, J., Baskaran, A., Zhang, Z., and Tokekar, P. (2020). Multi-agent reinforcement learning for persistent monitoring. *arXiv preprint arXiv:2011.01129*.

Chong, C., Mori, S., Tse, E., and Wishner, R. (1982). Distributed Estimation in Distributed Sensor Networks. In *1982 American Control Conference*, pages 820–826. IEEE. DOI: 10.23919/ACC.1982.4787968.

Chu, E.-W., Fan, H.-Y., Lin, W.-W., and Wang, C.-S. (2004). Structure-preserving Algorithms for Periodic Discrete-time Algebraic Riccati Equations. *International Journal of Control*, 77(8):767–788.

Dieci, L. and Eirola, T. (1996). Preserving Monotonicity in the Numerical Solution of Riccati Differential Equations. *Numerische Mathematik*, 74(1):35–47.

Enderlein, J. (2000). Tracking of fluorescent molecules diffusing within membranes. *Applied Physics B*, 71(5):773–777.

Fujimoto, K., Oji, Y., and Hamamoto, K. (2016). On Periodic Kalman Filters and Multi-rate Estimation. In *2016 IEEE Conference on Control Applications*, pages 934–939. DOI: 10.1109/CCA.2016.7587933.

Gallatin, G. M. and Berglund, A. J. (2012). Optimal laser scan path for localizing a fluorescent particle in two or three dimensions. *Optics Express*, 20(15):16381–16393.

Grocholsky, B., Makarenko, A., and Durrant-Whyte, H. (2003). Information-theoretic Coordinated Control of Multiple Sensor Platforms. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 1, pages 1521–1526. IEEE. DOI: 10.1109/ROBOT.2003.1241807.

Gronwall, T. H. (1919). Note on the Derivatives with Respect to a Parameter of the Solutions of a system of Differential Equations. *Annals of Mathematics*, pages 292–296.

Hari, S. K., Rathinam, S., Darbha, S., Kalyanam, K., Manyam, S. G., and Casbeer, D. (2019). The Generalized Persistent Monitoring Problem. In *2019 American Control Conference*, volume 2019-July, pages 2783–2788. DOI: 10.23919/ACC.2019.8815211.

Hussein, I. I. (2008). Kalman Filtering with Optimal Sensor Motion Planning. In *2008 American Control Conference*, pages 3548–3553. IEEE. DOI: 10.1109/ACC.2008.4587043.

Jianbo Shi and Malik, J. (2000). Normalized Cuts and Image Segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Jones, A., Schwager, M., and Belta, C. (2015). Information-guided Persistent Monitoring Under Temporal Logic Constraints. In *2015 American Control Conference*, pages 1911–1916. DOI: 10.1109/ACC.2015.7171012.

Kiefer, J. (1953). Sequential Minimax Search for a Maximum. *American Mathematical Society*, 4(3):502–506.

Kriegl, A., Michor, P. W., and Rainer, A. (2011). Denjoy–Carleman Differentiable Perturbation of Polynomials and Unbounded Operators. *Integral Equations and Operator Theory*, 71(3):407.

Kural, C., Balci, H., and Selvin, P. R. (2005). Molecular motors one at a time: Fiona to the rescue. *Journal of Physics: Condensed Matter*, 17(47):S3979.

Lan, X. and Schwager, M. (2013). Planning Periodic Persistent Monitoring Trajectories for Sensing Robots in Gaussian Random Fields. In *IEEE Int. Conf. on Robotics and Automation*, pages 2415–2420. IEEE. DOI: 10.1109/ICRA.2013.6630905.

Lan, X. and Schwager, M. (2014). A Variational Approach to Trajectory Planning for Persistent Monitoring of Spatiotemporal Fields. In *2014 American Control Conference*, pages 5627–5632. DOI: 10.1109/ACC.2014.6859098.

Lan, X. and Schwager, M. (2016). Rapidly exploring random cycles: Persistent estimation of spatiotemporal fields with multiple sensing robots. *IEEE Transactions on Robotics*, 32(5):1230–1244.

Lancaster, P. (1964). On Eigenvalues of Matrices Dependent on a Parameter. *Numerische Mathematik*, 6(1):377–387.

Laporte, G. (2009). Fifty years of Vehicle Routing. *Transportation Science*, 43(4):408–416.

Le Ny, J., Feron, E., and Dahleh, M. A. (2010). Scheduling Continuous-time Kalman Filters. *IEEE Transactions on Automatic Control*, 56(6):1381–1394.

Lin, X. and Cassandras, C. G. (2014). An Optimal Control Approach to the Multi-Agent Persistent Monitoring Problem in Two-dimensional Spaces. *IEEE Transactions on Automatic Control*, 60(6):1659–1664.

Lin, Y. and Andersson, S. B. (2019). Simultaneous Localization and Parameter Estimation for Single Particle Tracking via Sigma Points based EM. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 6467–6472. IEEE. DOI: 10.1109/CDC40024.2019.9029251.

Lin, Z., Liu, H. H., and Wotton, M. (2018). Kalman Filter-based Large-scale Wildfire Monitoring with a System of UAVs. *IEEE Transactions on Industrial Electronics*, 66(1):606–615.

Lofberg, J. (2004). Yalmip: A toolbox for modeling and optimization in matlab. In *2004 IEEE International Conference on Robotics and Automation*, pages 284–289. IEEE. DOI: 10.1109/CACSD.2004.1393890.

Manzo, C. and Garcia-Parajo, M. F. (2015). A review of progress in single particle tracking: from methods to biophysical insights. *Reports on Progress in Physics*, 78(12):124601.

Mondal, P. P. and Diaspro, A. (2013). *Fundamentals of Fluorescence Microscopy: Exploring Life with Light*. Springer Science & Business Media.

MOSEK ApS (2019). *The MOSEK optimization toolbox for MATLAB manual. Version 9.0.* Available at: http://docs.mosek.com/9.0/toolbox/index.html.

Nicolao, G. (1992). On the Convergence to the Strong Solution of Periodic Riccati Equations. *International Journal of Control*, 56(1):87–97.

Olfati-Saber, R. (2007). Distributed Kalman Filtering for Sensor Networks. In *2007 46th IEEE Conference on Decision and Control*, pages 5492–5498. IEEE. DOI: 10.1109/CDC.2007.4434303.

Ostertag, M., Atanasov, N., and Rosing, T. (2019). Robust Velocity Control for Minimum Steady State Uncertainty in Persistent Monitoring Applications. In *2019 American Control Conference*, pages 2501–2508. IEEE. DOI: 10.23919/ACC.2019.8814376.

Overholt, K. J. (1973). Efficiency of the Fibonacci Search Method. *BIT Numerical Mathematics*, 13(1):92–96.

Pasqualetti, F., Franchi, A., and Bullo, F. (2012). On Cooperative Patrolling: Optimal Trajectories, Complexity Analysis, and Approximation Algorithms. *IEEE Transactions on Robotics*, 28(3):592–606.

Pinto, S. C. and Andersson, S. B. (2021). Analysis of an Extremum Seeking Controller Under Bounded Disturbance. In *2021 IEEE 60th Conference on Decision and Control (CDC) (to appear)*. IEEE.

Pinto, S. C., Andersson, S. B., Hendrickx, J. M., and Cassandras, C. G. (2019). Optimal Multi-Agent Persistent Monitoring of the Uncertain State of a Finite Set of Targets. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 4280–4285. IEEE. DOI: 10.1109/CDC40024.2019.9029521.

Pinto, S. C., Andersson, S. B., Hendrickx, J. M., and Cassandras, C. G. (2020a). Multi-Agent Infinite Horizon Persistent Monitoring of Targets with Uncertain States in Multi-Dimensional Environments. *IFAC-PapersOnLine*, 53(2):10963–10968.

Pinto, S. C., Andersson, S. B., Hendrickx, J. M., and Cassandras, C. G. (2020b). Multi-agent persistent monitoring of targets with uncertain states. *arXiv preprint arXiv:2004.09647*.

Pinto, S. C., Andersson, S. B., Hendrickx, J. M., and Cassandras, C. G. (2020c). Optimal Periodic Multi-Agent Persistent Monitoring of a Finite Set of Targets with Uncertain States. In *2020 American Control Conference*, pages 5207–5212. IEEE. DOI: 10.23919/ACC45564.2020.9147376.

Pinto, S. C., Andersson, S. B., Hendrickx, J. M., and Cassandras, C. G. (2021a). A Semidefinite Programming Approach to Discrete-time Infinite Horizon Persistent Monitoring. *arXiv preprint arXiv:2104.00166*.

Pinto, S. C., Vickers, N. A., Sharifi, F., and Andersson, S. B. (2021b). Tracking Multiple Diffusing Particles Using Information Optimal Control. In *2021 American Control Conference*, pages 4033–4038. IEEE. DOI: 10.23919/ACC50511.2021.9482619.

Pukelsheim, F. (2006). *Optimal design of experiments*. SIAM.

Shakhatreh, H., Sawalmeh, A. H., Al-Fuqaha, A., Dou, Z., Almaita, E., Khalil, I., Othman, N. S., Khreishah, A., and Guizani, M. (2019). Unmanned Aerial Vehicles (UAVs): A Survey on Civil Applications and Key Research Challenges. *IEE Access*, 7:48572–48634.

Shen, H., Tauzin, L. J., Baiyasi, R., Wang, W., Moringo, N., Shuang, B., and Landes, C. F. (2017). Single particle tracking: from theory to biophysical applications. *Chemical reviews*, 117(11):7331–7376.

Shen, Z. and Andersson, S. B. (2009). LQG-based Tracking of Multiple Fluorescent Particles in Two-dimensions in a Confocal Microscope. In *2009 American Control Conference*, pages 1682–1687. DOI: 10.1109/ACC.2009.5160593.

Singh, A. and Baghel, A. S. (2009). A new grouping genetic algorithm approach to the multiple traveling salesperson problem. *Soft Computing*, 13(1):95–101.

Smith, S. L., Schwager, M., and Rus, D. (2011). Persistent Robotic Tasks: Monitoring and Sweeping in Changing Environments. *IEEE Transactions on Robotics*, 28(2):410–426.

Snyder, D. L. and Miller, M. I. (2012). *Random point processes in time and space*. Springer Science & Business Media.

Sun, C., Welikala, S., and Cassandras, C. G. (2020). Optimal composition of heterogeneous multi-agent teams for coverage problems with performance bound guarantees. *Automatica*, 117:108961.

Tang, L., Liu, J., Rong, A., and Yang, Z. (2000). A Multiple Traveling Salesman Problem Model for Hot Rolling Scheduling in Shanghai Baoshan Iron & Steel Complex. *European Journal of Operational Research*, 124(2):267 – 282.

Thrun, S. (2002). Probabilistic Robotics. *Communications of the ACM*, 45(3):52–57.

Varga, A. (2013). Computational Issues for Linear Periodic Systems: Paradigms, Algorithms, Open Problems. *International Journal of Control*, 86(7):1227–1239.

von Luxburg, U. (2007). A Tutorial on Spectral Clustering. http://arxiv.org/abs/0711.0189.

Wang, B. (2010). *Coverage control in sensor networks*. Springer Science & Business Media.

Wang, Y.-W., Zhao, M.-J., Yang, W., Zhou, N., and Cassandras, C. G. (2019). Collision-free trajectory design for 2-d persistent monitoring using second-order agents. *IEEE Transactions on Control of Network Systems*, 7(2):545–557.

Welikala, S. and Cassandras, C. G. (2020). Asymptotic Analysis for Greedy Initialization of Threshold-Based Distributed Optimization of Persistent Monitoring on Graphs. *IFAC-PapersOnLine*, 53(2):3433–3438.

Welikala, S. and Cassandras, C. G. (2021a). Event-Driven Receding Horizon Control for Distributed Estimation in Network Systems. In *2021 American Control Conference*, pages 1559–1564. IEEE. DOI: 10.23919/ACC50511.2021.9483147.

Welikala, S. and Cassandras, C. G. (2021b). Event-driven receding horizon control of energy-aware dynamic agents for distributed persistent monitoring. *arXiv preprint arXiv:2102.12963*.

Yu, X., Andersson, S. B., Zhou, N., and Cassandras, C. G. (2017). Optimal Dwell Times for Persistent Monitoring of a Finite set of Targets. In *2017 American Control Conference*, pages 5544–5549. IEEE. DOI: 10.23919/ACC.2017.7963817.

Yu, X., Andersson, S. B., Zhou, N., and Cassandras, C. G. (2018). Optimal Visiting Schedule Search for Persistent Monitoring of a Finite Set of Targets. In *2018 American Control Conference*, pages 4032–4037. DOI: 10.23919/ACC.2018.8431454.

Zhang, F. (2011). *Matrix Theory: Basic Results and Techniques*. Springer Science & Business Media.

Zhao, L., Zhang, W., Hu, J., Abate, A., and Tomlin, C. J. (2014). On the Optimal Solutions of the Infinite-horizon Linear Sensor Scheduling Problem. *IEEE Transactions on Automatic Control*, 59(10):2825–2830.

Zhou, N., Cassandras, C. G., Yu, X., and Andersson, S. B. (2019). Optimal Threshold-Based Distributed Control Policies for Persistent Monitoring on Graphs. In *2019 of American Control Conference*, pages 2030–2035. DOI: 10.23919/ACC.2019.8814440.

Zhou, N., Cassandras, C. G., Yu, X., and Andersson, S. B. (2020). The Price of Decentralization: Event-Driven Optimization for Multi-Agent Persistent Monitoring Tasks. *IEEE Transactions on Control of Network Systems*. DOI: 10.1109/TCNS.2020.3047314.

Zhou, N., Yu, X., Andersson, S. B., and Cassandras, C. G. (2018). Optimal Event-Driven Multiagent Persistent Monitoring of a Finite Set of Data Sources. *IEEE Transactions on Automatic Control*, 63(12):4204–4217.

CURRICULUM VITAE