Distributed Estimation in Network Systems Using Event-Driven Receding Horizon Control

Shirantha Welikala and Christos G. Cassandras

***Abstract*—We consider the problem of estimating the states of a distributed network of nodes (targets) through a team of cooperating agents (sensors) persistently visiting the nodes so that an overall measure of estimation error covariance evaluated over a finite period is minimized. We formulate this as a multi-agent persistent monitoring problem where the goal is to control each agent's trajectory defined as a sequence of target visits and the corresponding dwell times spent making observations at each visited target. A distributed on-line agent controller is developed where each agent solves a sequence of receding horizon control problems (RHCPs) in an event-driven manner. A novel objective function is proposed for these RHCPs so as to optimize the effectiveness of this distributed estimation process and its unimodality property is established under some assumptions. Moreover, a machine learning solution is proposed to improve the computational efficiency of this distributed estimation process by exploiting the history of each agent's trajectory. Finally, extensive numerical results are provided indicating significant improvements compared to other state-of-the-art agent controllers.**

I. Introduction

This paper considers the problem of controlling a team of mobile agents (*sensors*) deployed to monitor a finite set of points of interest (*targets*) in a mission space. Each target state follows independent (from other targets) stochastic dynamics and the goal of the agent team is to estimate the target states so that an overall measure of estimation error covariance evaluated over a finite period is minimized.

As introduced in [1], this problem is different from conventional *distributed estimation* problems [2] because: (i) we use mobile, rather than stationary, sensors, (ii) we aim to estimate a distributed set of target states rather than a common global state and (iii) we focus on developing an optimal distributed control strategy for the mobile sensor trajectories rather than an optimal fusion framework for the distributed sensor measurements. However, the proposed approach can still be seen as one of optimal data fusion, but one that estimates a potentially large number of target states using a typically small number of sensors. In fact, this shortage of sensors is what motivates the exploitation of each sensor's mobility. We also highlight that this is the same motivation that expanded the study of conventional optimal *coverage* problems [3] (where stationary agents are used to monitor a given mission space) to optimal *persistent monitoring* problems [4] (where mobile agents are used). Considering this analogy and many other similarities, we cast this optimal estimation problem as an optimal *persistent monitoring* problem.

In the literature, persistent monitoring problems have been widely studied and find many applications such as in sensing [5], data collection [6], surveillance [7] and energy management [8]. Many variants of persistent monitoring problems have been considered in the literature under different forms of (i) target state dynamics [4], [9], (ii) global objectives [10]–[13], (iii) agent motion dynamics [14], [15] and (iv) mission spaces [4], [16]–[18].

A closely related persistent monitoring problem is studied in [4] where the target state dynamics are assumed to be deterministic (i.e., each target state itself is a measure of uncertainty with no explicit stochasticity) and the agent team is tasked with minimizing the target state (uncertainty) values via sensing targets. To find the optimal agent trajectories in this problem setting, [4] proposes a network (graph) abstraction for the target-agent system along with a gradient-based distributed on-line parametric control solution. The subsequent work in [12] appends to this solution a centralized off-line stage to find an effective set of periodic agent trajectories as an initial condition. For the same persistent monitoring problem, our recent work in [14] takes an alternative approach and develops a distributed on-line solution based on Event-Driven Receding Horizon Control (RHC) [19]. This RHC solution has many attractive features, such as being gradient-free, parameter-free, initialization-free, computationally cheap and adaptive to various forms of state and system perturbations.

In contrast to [4], [12], [14], the persistent monitoring problems considered in this paper and [10], [20], [21] are more challenging as they assume that each target state follows independent stochastic dynamics and task the agent team to persistently estimate the set of target states so that an overall measure of error covariance associated with target state estimates is minimized. However, despite such differences in the problem setup, this class of persistent monitoring problems can be addressed by adopting the key concepts used in [12], [14]. For example, [10] formulates a minimax problem over an infinite horizon and proposes a centralized off-line periodic solution inspired by [12]. Similarly, this paper considers a mean overall estimation error covariance objective evaluated over a finite horizon and develops a distributed on-line RHC solution (not constrained to be periodic) inspired by [14].

While [10] and [21] adopt the persistent monitoring setting to address the underlying estimation task, they only consider single-agent scenarios. Therefore, they require additional clustering and assignment stages to handle multi-agent scenarios (analogous to [12]). Moreover, both [10] and [21] consider infinite horizon objective functions and develop periodic solutions in a centralized off-line stage. In contrast, [20] and this work (both of which also adopt the persistent monitoring setting to address the underlying estimation task) use finite horizon objective functions and develop distributed solutions

*Supported in part by NSF under grants ECCS-1931600, DMS-1664644, CNS-1645681, by AFOSR under grant FA9550-19-1-0158, by ARPA-E under grant DE-AR0001282 and by the NEXTCAR program under grant DE-AR0000796 and by the MathWorks.

The authors are with the Division of Systems Engineering and Center for Information and Systems Engineering, Boston University, Brookline, MA 02446, {shiran27,cgc}@bu.edu.

well-suited for multi-agent scenarios. However, the solution proposed in [20] is computationally expensive, off-line and time-driven. To address these limitations, this work restricts the target state dynamics to a one-dimensional space (as in [4], [12], [14]) and develops a computationally efficient, on-line and event-driven persistent monitoring solution.

The contributions of this paper are as follows. First, it is shown that each agent's trajectory is fully defined by the sequence of *control decisions* it makes at specific discrete event times. Second, a *receding horizon control problem* (RHCP) is formulated for an agent to solve at any one of these event times so as to determine the immediate set of optimal control decisions to execute within a *planning horizon*. Hence, the event-driven nature of this control approach significantly reduces the computational complexity due to its flexibility in the frequency of control updates. A novel element in this RHCP is that, unlike conventional RHC where the planning horizon is an exogenously selected global parameter [19], [22]–[24], it can simultaneously determine the *optimal* planning horizon length along with the optimal control decisions locally and asynchronously by each agent. Note that, similar to all RHC solutions [19], [22], [23], the determined optimal control decisions are subsequently executed only over a shorter *action horizon* defined by the next event that the agent observes, thus defining an event-driven process. Third, a novel RHCP objective function form (as opposed to the one used in [14]) is proposed to maximize the utilization of each agent's sensing capabilities over its planning horizon. Properties of this RHCP objective function form are studied, leading to establish its unimodality under certain conditions. This unimodality property is crucial as it ensures that each agent can independently solve their RHCPs globally and computationally efficiently using a simple gradient descent algorithm. In addition, a machine learning solution is developed to improve the computational efficiency of the proposed RHC-based agent controllers (i.e., of the overall distributed estimation process) by exploiting the history of each agent's optimal controls. Finally, the performance of the proposed RHC-based agent controllers is investigated in terms of providing accurate target state estimates and enabling target state controls compared to other state-of-the-art agent controllers.

Compared to related work in [4], [12], [14], [25] where agents are tasked with regulating deterministic piece-wise linear target state trajectories, this paper and [1], [10], [26] are entirely different as we task the agents with estimating stochastic linear target state trajectories. Further, compared to [14], this paper and [1] use a novel RHCP objective function form. This paper also provides a justification and extensive analysis of this novel RHCP objective function form. In addition, compared to [14], this paper proposes a machine learning solution to improve the computational efficiency of the RHC-based agent controllers. Moreover, while [10], [26] propose centralized off-line parametric control solutions, [1] and this paper propose distributed on-line RHC solutions. Compared to [1], this paper provides: 1) the details of all the RHCPs, 2) a milder assumption, 3) new theoretical results, 4) a machine learning based computational efficiency improvement scheme, and 5) new numerical results.

The paper is organized as follows. The problem formulation is presented in Section II and a few preliminary theoretical results are discussed in Section III. Section IV and V present the RHCP formulation and its solution, respectively. The subsequent Section VI describes how machine learning can be used to assist in solving RHCPs efficiently. The performance of the proposed RHC method is demonstrated using simulation results in Section VII. Finally, concluding remarks and future work are provided in Section VIII.

II. Problem Formulation

We consider M stationary nodes (targets) in the set $\mathscr{V} = \{1,2,\dots,M\}$ and N mobile agents (sensors) in the set $\mathscr{A} = \{1,2,\dots,N\}$ located in an l-dimensional mission space. The location of a target $i \in \mathscr{V}$ is fixed at $Y_i \in \mathbb{R}^l$ and that of an agent $a \in \mathscr{A}$ at time t is denoted by $s_a(t) \in \mathbb{R}^l$.

*a) **Graph Topology**:* A directed graph topology $\mathscr{G} = (\mathscr{V}, \mathscr{E})$ is embedded into the mission space so that the *targets* are represented by the graph *vertices* $\mathscr{V}$ and the inter-target *trajectory segments* available for agents to travel between targets are represented by the graph *edges* $\mathscr{E} \subseteq \{(i,j) : i,j \in \mathscr{V}\}$. These trajectory segments are allowed to take arbitrary shapes in $\mathbb{R}^l$ to account for possible constraints in the mission space and agent dynamics. We use ρ_{ij} to represent the *travel time* that an agent spends on a trajectory segment $(i,j) \in \mathscr{E}$ to reach target j from target i.

*b) **Target Dynamics**:* Each target $i \in \mathscr{V}$ has an associated state $\phi_i(t) \in \mathbb{R}^n$ that follows the dynamics

$$\dot{\phi}_i(t) = A_i\phi_i(t) + B_i\upsilon_i(t) + w_i(t), \tag{1}$$

where $\{w_i(t)\}_{i\in\mathscr{V}}$ are mutually independent, zero mean, white, Gaussian distributed processes with $E[w_i(t)w_i'(t)] = Q_i$. Similar to [10], [20], [21], in this work, the main focus is on persistently maintaining an accurate set of estimates of these target states using the fleet of mobile agents as target state sensors. Hence, in this setting, agents are not responsible for controlling target states and thus we assume that each target i independently selects (or is aware of) its control input $\upsilon_i(t)$.

When an agent visits a target $i \in \mathscr{V}$, it takes the measurements $z_i(t) \in \mathbb{R}^m$ which follow a linear observation model

$$z_i(t) = H_i\phi_i(t) + v_i(t), \tag{2}$$

where $\{v_i(t)\}_{i\in\mathscr{V}}$ are mutually independent, zero mean, white, Gaussian distributed processes with $E[v_i(t)v_i'(t)] = R_i$.

The matrices in (1) and (2): A_i, B_i, Q_i, H_i and R_i are time invariant and known at target $i \in \mathscr{V}$. Moreover, the matrices Q_i and R_i are positive definite.

*c) **Kalman-Bucy Filter**:* Considering the models (1) and (2), the maximum likelihood estimator $\hat{\phi}_i(t)$ of the target state $\phi_i(t)$ is a Kalman-Bucy filter [27] evaluated at target i:

$$\dot{\hat{\phi}}_i(t) = A_i\hat{\phi}_i(t) + B_i\upsilon_i(t) + \eta_i(t)\Omega_i(t)H_i'R_i^{-1}\left(z_i(t) - H_i\hat{\phi}_i(t)\right),$$

where $\Omega_i(t)$ is the estimation *error covariance* (i.e., $\Omega_i(t) = E(e_i(t)e_i'(t))$ with $e_i(t) = \phi_i(t) - \hat{\phi}_i(t)$) given by the matrix Riccati equation [28]:

$$\dot{\Omega}_i(t) = A_i\Omega_i(t) + \Omega_i(t)A_i' + Q_i - \eta_i(t)\Omega_i(t)G_i\Omega_i(t), \tag{3}$$

with $G_i = H_i'R_i^{-1}H_i$ and (using $\mathbf{1}\{\cdot\}$ as the indicator function)

$$\eta_i(t) \triangleq \mathbf{1}\{\text{Target } i \text{ is observed by an agent at time } t\}. \tag{4}$$

Based on (3), when a target $i \in \mathscr{V}$ is being sensed by an agent (i.e., when $\eta_i(t) = 1$), the covariance $\Omega_i(t)$ decreases and, as a result, the state estimate $\hat{\phi}_i$ becomes more accurate. The opposite occurs when a target is not being observed by an agent. It is also worth pointing out that the dynamics of the covariance $\Omega_i(t)$ (3) are independent of the target state control $\upsilon_i(t)$ in (1), due to the principle of separation [29].

d) ***Agent Model:*** According to the adopted graph topology, we assume that once an agent $a \in \mathscr{A}$ spends the required travel time to reach a target $i \in \mathscr{V}$, its location $s_a(t)$ falls within a certain range from the target location Y_i which enables establishing a constant sensing ability (i.e., H_i, R_i are fixed) of the target state $\phi_i(t)$. Without loss of generality, we denote the number of agents present at target $i \in \mathscr{V}$ at time t as

$$N_i(t) \triangleq \sum_{a \in \mathscr{A}} \mathbf{1}\{s_a(t) = Y_i\}. \tag{5}$$

To prevent resource (agent sensing) wastage and simplify the analysis, similar to [12], [14], we next introduce a control constraint that prevents *simultaneous target sharing* by multiple agents at each target $i \in \mathscr{V}$ as

$$N_i(t) \in \{0, 1\}, \ \forall t \geq 0. \tag{6}$$

Evidently, this constraint only applies if $N \geq 2$. Moreover, under (6), according to the definitions (4) and (5), $\eta_i(t) = N_i(t)$. Also, note that due to the use of a fixed set of travel times $\{\rho_{ij} : (i,j) \in \mathscr{E}\}$, the analysis in this paper is independent of the agent motion dynamic model (similar to [4], [12], [14]). However, analogous to the work in [25], with some modifications, the proposed solution in this paper can be adapted to accommodate specific agent dynamic models.

e) ***Global Objective:*** The goal is to design controllers for the agents to minimize the finite horizon global objective

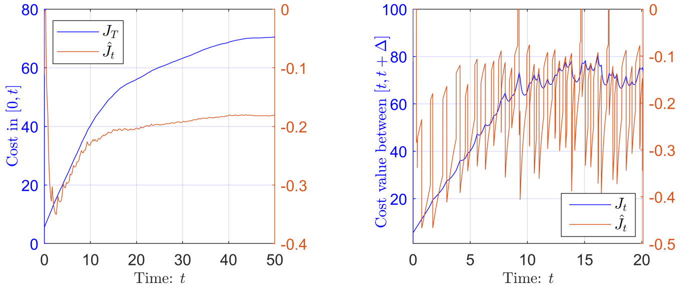
$$J_T \triangleq \frac{1}{T} \int_0^T \sum_{i \in \mathscr{V}} \mathrm{tr}(\Omega_i(t)) dt. \tag{7}$$

This choice of global objective (7) was inherited from related prior work [26]. Based on its form, clearly it motivates finding agent trajectories that minimize target state estimation error covariance values across the target network over the specified finite horizon. Therefore, (7) is a natural and intuitive choice for the global objective. Moreover, as we will see in the sequel, (7) is simple enough to be conveniently decomposed, and thus, it also motivates the development of distributed on-line optimal agent control strategies.

Nevertheless, we also propose an alternative global objective

$$\hat{J}_T \triangleq -\frac{\int_0^T \sum_{i \in \mathscr{V}} \eta_i(t) \mathrm{tr}(\Omega_i(t)) dt}{\int_0^T \sum_{i \in \mathscr{V}} \mathrm{tr}(\Omega_i(t)) dt}. \tag{8}$$

First, note that the denominator of (8) is proportional to (7). Hence, optimizing (minimizing) (8) will lead to agent trajectories that minimize (7) (notice the negative sign in (8)). Second, note that the numerator of (8) represents a quantity that reflects the overall agent *sensing effort* as $\eta_i(t) = 1$ only when an agent senses the target i. Therefore, this alternative objective (8), omitting the negative sign, can be seen as the efficiency of overall agent sensing effort, i.e., the fraction of resources (agents) used for sensing (as opposed to both sensing and traveling). Hence, it is clear that optimizing (8) requires agents to optimally allocate their sensing resources across the target network over the finite horizon $[0, T]$. Moreover, it is worth pointing out that compared to (7), (8) is conveniently normalized, thus its value has a direct intuitive interpretation.



(a) Accumulated cost

(b) Instantaneous cost

Fig. 1: Comparison between J_t in (7) and $\hat{J}_t$ in (8).

A final observation is that numerical experiments have shown that both J_t (7) and $\hat{J}_t$ (8) profiles (under the same agent controls over $t \in [0, T]$) behave in the same manner after a brief transient phase (e.g., see Fig. 1(a)). Therefore, we can see that both (7) and (8) are equally appropriate global objective function choices for this optimal estimation problem.

As mentioned earlier, we view this optimal estimation problem as a *persistent monitoring on networks* (PMN) problem. Moreover, we assume that the initial condition of this PMN problem setup, defined by $\phi_i(0)$, $\hat{\phi}_i(0)$ and $\Omega_i(0)$, $\forall i \in \mathscr{V}$, is known at respective targets.

f) ***Agent Control:*** Based on the graph topology $\mathscr{G} = (\mathscr{V}, \mathscr{E})$, we define the *neighbor set* and the *neighborhood* of a target $i \in \mathscr{V}$ as $\mathscr{N}_i \triangleq \{j : (i,j) \in \mathscr{E}\}$ and $\bar{\mathscr{N}_i} \triangleq \mathscr{N}_i \cup \{i\}$, respectively. The basic agent controls are as follows. Whenever an agent $a \in \mathscr{A}$ is ready to leave a target $i \in \mathscr{V}$, it selects a *next-visit* target $j \in \mathscr{N}_i$. Thereafter, the agent travels over $(i,j) \in \mathscr{E}$ to arrive at target j upon spending ρ_{ij} amount of time. Subsequently, it selects a *dwell-time* $u_j \in \mathbb{R}_{\geq 0}$ to spend at target j (which contributes to decreasing $\Omega_j(t)$, making $\hat{\phi}_j(t) \to \phi_j(t)$), and then makes another next-visit decision.

Therefore, the overall control exerted by an agent consists of a sequence of *dwell-times* $u_i \in \mathbb{R}_{\geq 0}$ and *next-visit* targets $j \in \mathscr{N}_i$. Our goal is to determine $(u_i(t), j(t))$ for any agent residing at a target $i \in \mathscr{V}$ at any time $t \in [0, T]$ which are optimal in the sense of minimizing the global objective (7).

This PMN problem is much more complicated than the well-known NP-hard traveling salesman problem (TSP) based on the metrics: (i) the number of decision variables, and (ii) the nature of feasible decision spaces. In particular, in PMN: (i) we need to determine dwell-times at each visited target, and account for (ii) the involved target dynamics, (iii) the presence of multiple agents, and (iv) the freedom to revisit targets.

The same reasons make it computationally intractable to apply dynamic programming techniques so as to obtain the optimal controls, even for a relatively simple PMN problem.

g) ***Receding Horizon Control:*** In order to address this hard dynamic optimization problem, this paper proposes an *Event-Driven Receding Horizon Controller* (RHC) at each agent $a \in \mathscr{A}$. Even though the key idea behind RHC derives from Model Predictive Control (MPC), it exploits the event-driven nature of the considered problem to significantly reduce the complexity by effectively decreasing the frequency of control updates. As introduced and extended later on in [19] and [14], [22], [23], respectively, the RHC method involves solving an optimization problem of the form (7) limited to a finite *planning horizon*, whenever an event of interest to the (agent) controller is observed. The determined optimal controls are then executed over an *action horizon* defined by the occurrence of the next such event. This event-driven process is continued iteratively.

Pertaining to the PMN problem considered in this paper, the RHC, when invoked at time t for an agent residing at target $i \in \mathscr{V}$, aims to determine (i) the immediate dwell-time u_i at target i, (ii) the next-visit target $j \in \mathscr{N}_i$ and (iii) the next dwell-time u_j at target j. These *control decisions* are jointly denoted by $U_i(t) \triangleq [u_i(t),\, j(t),\, u_j(t)]$ and its optimal value is obtained by solving an optimization problem of the form

$$U_i^*(t) = \underset{U_i(t)\in\mathbb{U}(t)}{\arg\min} \ \left[J_H(X_i(t),U_i(t);H) + \hat{J}_H(X_i(t+H))\right], \quad (9)$$

where $X_i(t)$ is the current local state and $\mathbb{U}(t)$ is the feasible control set at t (exact definition is provided later). The term $J_H(X_i(t),U_i(t);H)$ represents the immediate cost over the planning horizon $[t,t+H]$ and $\hat{J}_H(X_i(t+H))$ stands for an estimate of the future cost based on the state at $t+H$.

In contrast to standard methods where the planning horizon length H is exogenously selected, here we adopt the *variable horizon* approach proposed in [14] where the planning horizon length is treated as an upper-bounded function of control decisions $w(U_i(t)) \leq H$. In other words, we constrain the planning horizon to be $[t,t+w(U_i(t))] \subseteq [t,t+H]$ where $[t,t+H]$ is now treated as a predefined *fixed planning horizon*. Hence, this approach incorporates the selection of planning horizon length $w(U_i(t))$ into the optimization problem (9), which now can be re-stated as

$$\begin{aligned} U_i^*(t) = \ &\underset{U_i(t)\,\in\,\mathbb{U}(t)}{\arg\min} \quad J_H(X_i(t),U_i(t);w(U_i(t))) \\ &\text{subject to} \quad w(U_i(t)) \leq H. \end{aligned} \quad (10)$$

Note that the term $\hat{J}_H(X_i(t+H))$ in (9) has been omitted in (10). This compromise is made to enforce the proposed RHC method to be *distributed* so that each agent can separately solve (10) on-line using only its local state information. Apart from achieving such desirable qualities, this compromise also provides additional compensation by optimizing the planning horizon length $w(U_i(t))$ in (10) (as opposed to using a fixed planning horizon length H in (9)). In particular, this optimization of $w(U_i(t))$ compensates for the intrinsic inaccuracies resulting from the omission of $\hat{J}_H(X_i(t+H))$. This claim is justified in the following remark.

Remark 1: Note that in (9), $\hat{J}_H(X_i(t+H))$ motivates an agent to optimize its state at the end of the planning horizon. However, under (10), in lieu of $\hat{J}_H(X_i(t+H))$, the $w(U_i(t))$ term motivates an agent to optimize instead (through $U_i(t)$) its end of the planning horizon $t+w(U_i(t))$, thus compensating for the omission of the future cost.

III. Preliminary Results

Based on (3), the error covariance $\Omega_i(t)$ of any target $i \in \mathscr{V}$ is continuous and piece-wise differentiable. Specifically, $\dot{\Omega}_i(t)$ jumps only when one of the following two (strictly local) *events* occurs: (i) an agent arrival at target i, or (ii) an agent departure from target i. These two events occur alternatively and respectively trigger two different *modes* of subsequent $\Omega_i(t)$ behaviors, named *active* and *inactive* modes, described by using $\eta_i(t) = 1$ and $\eta_i(t) = 0$ in (3). In the following discussion, we use $\eta_i(t) \in \{0,1\}$ to represent the mode of target $i \in \mathscr{V}$ and I_n as the identity matrix in $\mathbb{R}^n$.

Lemma 1: If a target $i \in \mathscr{V}$ is in the mode $\eta_i(t) \in \{0,1\}$ during a time period $[t_0,t_1]$, its error covariance $\Omega_i(t)$ for any time $t \in [t_0,t_1]$ is given by

$$\Omega_i(t) = C_i(t)D_i^{-1}(t), \quad (11)$$

where $\begin{bmatrix} C_i(t) \\ D_i(t) \end{bmatrix} = e^{\Psi_i(t-t_0)} \begin{bmatrix} \Omega_i(t_0) \\ I_n \end{bmatrix}$ with $\Psi_i = \begin{bmatrix} A_i & Q_i \\ \eta_i(t)G_i & -A_i^T \end{bmatrix}$.

Proof: Omitting the argument t for notational convenience and using the substitution $\Omega_i = C_iD_i^{-1}$ in (3) gives

$$\dot{C}_i - C_iD_i^{-1}\dot{D}_i = (A_iC_i + Q_iD_i) - C_iD_i^{-1}(\eta_iG_iC_i - A_i^TD_i).$$

Note that there is an extra degree of freedom left for selecting C_i and D_i as the substitution (11) has replaced Ω_i with two new variables C_i and D_i. We exploit this extra degree of freedom to derive two equations from the above equation as follows. Notice that both sides of the above equation takes an affine linear form with respect to $-C_iD_i^{-1}$. Therefore, equating the coefficients of 1 and $-C_iD_i^{-1}$ terms above gives two equations:

$$\begin{bmatrix} \dot{C}_i(t) \\ \dot{D}_i(t) \end{bmatrix} = \begin{bmatrix} A_i & Q_i \\ \eta_i(t)G_i & -A_i^T \end{bmatrix} \begin{bmatrix} C_i(t) \\ D_i(t) \end{bmatrix}.$$

Recall that $\eta_i(t) = 1$ if the target i is active and $\eta_i(t) = 0$ otherwise. Finally, setting the initial conditions: $C_i(t_0) = \Omega_i(t_0)$ and $D_i(t_0) = I_n$, the above linear differential equation can be solved to obtain the result in (11). ■

Using the above lemma, a simpler expression for $\Omega_i(t)$ than (11) can be derived if target i is in the inactive mode: $\eta_i(t) = 0$.

Corollary 1: If a target $i \in \mathscr{V}$ is inactive during $t \in [t_0,t_1]$, the corresponding $\Omega_i(t)$ for any time $t \in [t_0,t_1]$ is given by

$$\Omega_i(t) = \Phi_i(t)\left[\Omega_i(t_0) + (t-t_0)\int_0^1 \psi_i(t,x)dx\right]\Phi_i^T(t), \quad (12)$$

where $\Phi_i(t) = e^{A_i(t-t_0)}$ and $\psi_i(t,x) = (\Phi_i(t))^{-x}Q_i(\Phi_i^T(t))^{-x}$.

Proof: According to Lemma 1, when $\eta_i(t) = 0$, $\Omega_i(t)$ is given by (11) where Ψ_i is a block triangular matrix. Therefore, $e^{\Psi_i(t-t_0)}$ can be written (using [30, p. 1]) as

$$e^{\Psi_i(t-t_0)} = \begin{bmatrix} \Phi_i(t) & (t-t_0)\int_0^1 (\Phi_i^T(t))^{1-x}Q_i(\Phi_i^T(t))^{-x}dx \\ 0 & (\Phi_i^T(t))^{-1} \end{bmatrix}.$$

Applying this in (11) gives $D_i(t) = \Phi_i^{-1}(t)$ and $C_i(t) = \Phi_i(t)\Omega_i(t_0) + (t-t_0)\int_0^1(\Phi_i^T(t))^{1-x}Q_i(\Phi_i^T(t))^{-x}dx$. Finally, $\Omega_i(t) = C_i(t)D_i^{-1}(t)$ gives the expression in (12). ■

From Lemma 1 and Corollary 1, it is clear that the exact form of the "tr$(\Omega_i(t))$" expression required for the global objective (7) cannot be written more compactly - unless the matrices A_i and Ψ_i have some additional properties.

*a) **One-Dimensional PMN Problem**:* In the remainder of this paper, similar to [4], [12], [14], we constrain ourselves to one-dimensional target state dynamics and agent observation models by setting

$$n = m = 1, \quad (13)$$

in (1) and (2). This is a reasonable assumption given that the goal of this work is to derive necessary theoretical results to apply the RHC method and then to explore its feasibility for the considered particular PMN problem setup. In future work, we expect to generalize these theoretical results and the RHC solution to higher-dimensional target state models. Therefore, we henceforth consider the fixed target parameters A_i, Q_i, H_i, R_i, G_i and the time-varying quantities $\phi_i(t), z_i(t), \hat{\phi}_i(t), \Omega_i(t), \forall i \in \mathcal{V}$ as scalars.

*b) **Local Contribution**:* The contribution to the global objective J_T in (7) by a target $i \in \mathcal{V}$ during a time period $[t_0,t_1]$ is defined as $\frac{1}{T}J_i(t_0,t_1)$ where

$$J_i(t_0,t_1) \triangleq \int_{t_0}^{t_1} \mathrm{tr}(\Omega_i(t))dt = \int_{t_0}^{t_1} \Omega_i(t)dt. \tag{14}$$

We further define the corresponding active and inactive portions of the above *local contribution* term $J_i(t_0,t_1)$ in (14) respectively as $J_i^A(t_0,t_1)$ and $J_i^I(t_0,t_1)$ where

$$\begin{aligned} J_i^A(t_0,t_1) &\triangleq \int_{t_0}^{t_1} \eta_i(t)\Omega_i(t)dt \quad \text{and} \\ J_i^I(t_0,t_1) &\triangleq \int_{t_0}^{t_1} (1-\eta_i(t))\Omega_i(t)dt. \end{aligned} \tag{15}$$

Notice that, by definition, $J_i(t_0,t_1) = J_i^A(t_0,t_1) + J_i^I(t_0,t_1)$.

Lemma 2: If a target $i \in \mathcal{V}$ is *active* during $t \in [t_0,t_1]$, the corresponding $\Omega_i(t)$ for any time $t \in [t_0,t_1]$ is given by

$$\Omega_i(t) = \frac{c_{i1} + c_{i2}e^{-\lambda_i(t-t_0)}}{v_{i1}c_{i1} + v_{i2}c_{i2}e^{-\lambda_i(t-t_0)}}, \tag{16}$$

where $\lambda_i = 2\sqrt{A_i^2 + Q_iG_i}$, $v_{i1} = \frac{1}{Q_i}(-A_i + \sqrt{A_i^2+Q_iG_i})$, $v_{i2} = \frac{1}{Q_i}(-A_i - \sqrt{A_i^2+Q_iG_i})$, $c_{i1} = v_{i2}\Omega_i(t_0) - 1$ and $c_{i2} = -v_{i1}\Omega_i(t_0) + 1$. The corresponding local contribution $J_i(t_0,t_0+w)$ in (14) (where $w = (t-t_0)$) is given by $J_i(t_0,t_0+w) = J_i^A(t_0,t_0+w)$ where

$$J_i^A(t_0,t_0+w) = \frac{1}{G_i}\log\left(\frac{v_{i1}c_{i1}+v_{i2}c_{i2}e^{-\lambda_i w}}{v_{i2}-v_{i1}}\right) + \frac{1}{v_{i1}}w. \tag{17}$$

Proof: We first use Lemma 1 to derive (16). Note that under (13), Ψ_i in (11) is such that $\Psi_i \in \mathbb{R}^{2\times 2}$ and it can be simplified using $\eta_i(t) = 1$, since, by assumption, the target i is active. The eigenvalues of Ψ_i are $\pm\lambda_i/2$ and the corresponding generalized eigenvector matrix is $\begin{bmatrix} 1 & 1 \\ v_{i1} & v_{i2}\end{bmatrix}$. Therefore, the matrix exponent $e^{\Psi_i w}$ required in (11) can be evaluated as $e^{\Psi_i w} = \begin{bmatrix} 1 & 1 \\ v_{i1} & v_{i2}\end{bmatrix}\mathrm{diag}(e^{\frac{\lambda_i w}{2}}, e^{-\frac{\lambda_i w}{2}})\begin{bmatrix} 1 & 1 \\ v_{i1} & v_{i2}\end{bmatrix}^{-1}$. Applying this result in (11) gives

$$\begin{bmatrix} C_i(t) \\ D_i(t) \end{bmatrix} = e^{\frac{\lambda_i w}{2}} \begin{bmatrix} v_{i2} - v_{i1}e^{-\lambda_i w} & -1+e^{-\lambda_i w} \\ v_{i1}v_{i2}(1-e^{-\lambda_i w}) & -v_{i1}+v_{i2}e^{\lambda_i w} \end{bmatrix} \begin{bmatrix} \Omega_i(t_0) \\ 1 \end{bmatrix}.$$

Since $\Omega_i(t) = C_i(t)/D_i(t)$ (from (11)), we now can use the above result to obtain (16).

Finally, using (15), (17) can be obtained by analytically evaluating the integral of $\Omega_i(t)$ over the period $[t_0, t_0+w]$:

$$J_i^A(t_0,t_0+w) = \int_{t_0}^{t_0+w} \frac{c_{i1}+c_{i2}e^{-\lambda_i(\tau-t_0)}}{v_{i1}c_{i1}+v_{i2}c_{i2}e^{-\lambda_i(\tau-t_0)}}d\tau. \qquad \blacksquare$$

Lemma 3: If a target $i \in \mathcal{V}$ is *inactive* during $t \in [t_0,t_1]$, the corresponding $\Omega_i(t)$ for any time $t \in [t_0,t_1]$ is given by

$$\Omega_i(t) = \left(\Omega_i(t_0) + \frac{Q_i}{2A_i}\right)e^{2A_i(t-t_0)} - \frac{Q_i}{2A_i}, \tag{18}$$

and the corresponding local contribution $J_i(t_0,t_0+w)$ (where $w = (t-t_0)$) is given by $J_i(t_0,t_0+w) = J_i^I(t_0,t_0+w)$ where

$$J_i^I(t_0,t_0+w) = \frac{1}{2A_i}\left(\Omega_i(t_0) + \frac{Q_i}{2A_i}\right)(e^{2A_iw}-1) - \frac{Q_i}{2A_i}w. \tag{19}$$

Proof: These two results (18) and (19) can be proved by using Ψ_i in (11) as $\Psi_i = [A_i\ Q_i; 0\ -A_i^T]$ and following the same steps used in the proof of Lemma 2. $\blacksquare$

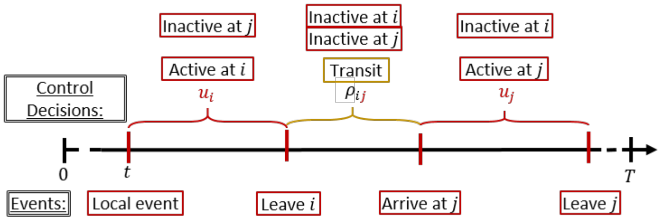


Fig. 2: Event timeline and control decisions under RHC.

IV. RHC PROBLEM (RHCP) FORMULATION

Consider an agent $a \in \mathcal{A}$ residing on a target $i \in \mathcal{V}$ at some time $t \in [0,T]$. Recall that control $U_i(t)$ in (10) consists of the dwell-time u_i at the current target i, the next-visit target $j \in \mathcal{N}_i$, and the dwell-time u_j at the next-visit target j (see Fig. 2). Therefore, agent a has to optimally select the three control decisions (control vector) $U_i(t) = [u_i(t), j(t), u_j(t)]$.

*a) **The RHCP**:* Let us denote the real-valued component of the control vector $U_i(t)$ in (10) as $U_{ij} \triangleq [u_i, u_j]$ (omitting time arguments for notational simplicity). The discrete component of $U_i(t)$ is simply the next-visit target $j \in \mathcal{N}_i$. Let us define the planning horizon length $w(U_i(t))$ in (10) as

$$w = w(U_{ij}) \triangleq |U_{ij}| + \rho_{ij} = u_i + \rho_{ij} + u_j \tag{20}$$

($|\cdot|$ denotes the cardinality operator or the 1-norm depending on the argument) so that it covers the control decisions and corresponding controllable events that can occur in the immediate future pertaining to the current neighborhood $\bar{\mathcal{N}_i}$ as shown in Fig. 2. The current local state $X_i(t)$ in (10) is taken as $X_i(t) = \{\Omega_j(t) : j \in \bar{\mathcal{N}_i}\}$. Then, the optimal controls are obtained by solving (10), which can be re-stated as the following set of optimization problems, henceforth called the RHC Problem (RHCP):

$$\begin{aligned} U_{ij}^* = \quad & \underset{U_{ij}\,\in\,\mathbb{U}}{\arg\min} \quad J_H(X_i(t), U_{ij}; w(U_{ij}));\ \forall j \in \mathcal{N}_i, \\ & \text{subject to} \quad w(U_{ij}) \leq H \end{aligned} \tag{21}$$

$$j^* = \underset{j\,\in\,\mathcal{N}_i}{\arg\min} \quad J_H(X_i(t), U_{ij}^*; w(U_{ij}^*)). \tag{22}$$

Before getting into details, note that (21) involves solving $|\mathcal{N}_i|$ optimization problems, one for each neighbor $j \in \mathcal{N}_i$. Then, (22) determines j^* through a simple numerical comparison. Therefore, the optimal control vector $U_i^*(t)$ of (10) is the composition: $U_i^*(t) = \{U_{ij^*}^*, j^*\}$.

The RHCP objective function $J_H(\cdot)$ is chosen in terms of the *local objective function* of target i, which is denoted by $\bar{J}_i(t_0,t_1)$ over any interval $[t_0,t_1] \subseteq [0,T]$ (the exact definition of $\bar{J}_i(t_0,t_1)$ is provided later on in (24)). In particular, we define the RHCP objective function as the local objective function of target i evaluated over the planning horizon $[t, t+w]$:

$$J_H(X_i(t), U_{ij}; H) = \bar{J}_i(t, t+w), \tag{23}$$

and the RHCP feasible control space as $\mathbb{U} = \{U : U \in \mathbb{R}^2,\ U \geq 0,\ |U| + \rho_{ij} \leq H\}$ (including the constraint $w(U_{ij}) \leq H$)

*b) **Planning Horizon**:* In conventional RHC methods, the RHCP objective function is evaluated over a fixed planning horizon, e.g., $[t, t+H] \subseteq [0,T]$, where H is selected exogenously. This leads to control solutions that are dependent on the choice of the used fixed planning horizon length H. When developing on-line control methods, having such a dependence on a predefined parameter is undesirable, as it prevents the controller from having the opportunity to fine-tune H and re-evaluate its controls.

However, through (20) and (23) above, we have made the RHCP solution (i.e., (21) and (22)) free of the parameter H (i.e., fixed planning horizon length) by only using H as an upper-bound to the actual planning horizon length $w(U_{ij})$ in (20) and selecting H to be sufficiently large (e.g., $H = T - t$). Moreover, since the planning horizon length $w(U_{ij})$ is control-dependent, this RHCP formulation simultaneously determines the optimal planning horizon length $w^* = |U^*_{ij^*}| + \rho_{ij^*}$ in terms of the optimal control vector $U^*_i(t) = \{U^*_{ij^*}, j^*\}$.

In all, we use two planning related horizon concepts in this paper: 1) the fixed planning horizon $[t, t+H]$, and 2) the planning horizon $[t, t+w(U_{ij})]$. They are related such that $[t, t+w(U_{ij})] \subseteq [t, t+H]$ where H is a predefined sufficiently large constant such as $H = T - t$ and $w(U_{ij})$ is optimized online through (21)-(22) to be $w(U^*_{ij^*})$.

*c) **Local Objective**:* As mentioned earlier, the local objective function of a target $i \in \mathscr{V}$ over a period $[t_0, t_1) \subseteq [0, T]$ is denoted by $\bar{J}_i(t_0, t_1)$. The purpose of $\bar{J}_i$ is to be used in (23) as the RHCP objective by each agent that visits target i for the selection of its controls $U_i = [u_i, j, u_j]$.

The local versions of the global objective (7) (based on "local contribution" functions (14)): J_i and $\sum_{j \in \bar{\mathscr{N}}_i} J_j$ are two conventional candidates for the local objective function $\bar{J}_i$ [14]. However, note that: (i) such candidate forms can be written as summations of the contribution terms in (17) and (19), (ii) both (17) and (19) increase monotonically with its argument w and (iii) in this case, $w = u_i + \rho_{ij} + u_j$ (20). Therefore, when minimizing $\bar{J}_i$, both of its candidate forms (J_i and $\sum_{j \in \bar{\mathscr{N}}_i} J_j$) yield the controls $u^*_i = 0, u^*_j = 0$ in an attempt to minimize w (making $w^* = \rho_{ij}$). This would imply that no agent ever dwells at any target.

Hence, instead of using a local version of the global objective (7), we propose to use a local version of the alternative global objective (8) as the local objective function:

$$\bar{J}_i(t_0, t_1) \triangleq -\frac{\sum_{j \in \bar{\mathscr{N}}_i} J^A_j(t_0, t_1)}{\sum_{j \in \bar{\mathscr{N}}_i} J_j(t_0, t_1)}. \tag{24}$$

This choice of local objective represents the *normalized active contribution* (i.e., the contribution during agent visits in (15)) of the targets in the neighborhood $\bar{\mathscr{N}}_i$ over the interval $[t_0, t_1)$. Due to this particular form (24), when it is used as the RHCP objective function (23), the agent (residing at target i) will have to optimally allocate its sensing capabilities (resources) over the target i and the next-visit target $j \in \mathscr{N}_i$ (i.e., have to optimally select the controls u_i and u_j). Moreover, in the sequel, we will show that this local objective function is *unimodal* in most cases of interest.

Since we have already shown that (8) and (7) perform in an equivalent manner (see Fig. 1(a) and [31, Tab. II, Fig.10]), we can also conclude that agents minimizing a local version of (8) (i.e., (24)) can in fact lead to minimizing (7). Moreover, when the instantaneous values of J_t (7) and $\hat{J}_t$ (8) (i.e., evaluated over a very small period $[t, t+\Delta]$) are compared, we have seen that $\hat{J}_t$ is more sensitive to the variations of the system (while remaining within a small interval) compared to J_t (e.g., see Fig 1(b)). These qualities imply the feasibility of (24) as a local objective function for the use of agents to decide their controls (in a distributed manner) so as to optimize the global objective (7).

Finally, note that the local objective function $\bar{J}_i(t_0, t_1)$ in (24) can be obtained locally at target i by evaluating the required $J^A_j(t_0, t_1)$ and $J^I_j(t_0, t_1)$ terms for all $j \in \bar{\mathscr{N}}_i$ using Lemmas 2 and 3, respectively (recall $J_j(t_0, t_1) = J^A_j(t_0, t_1) + J^I_j(t_0, t_1)$).

*d) **Event-Driven Action Horizon**:* Similar to all receding horizon controllers, an optimal receding horizon control solution computed over a planning horizon $[t, t+w^*] \subseteq [t, t+H]$ is generally executed only over a shorter *action horizon* $[t, t+h] \subseteq [t, t+w^*]$. In this event-driven persistent monitoring setting, the value of h is determined by the first event that takes place after the time instant when the RHCP was last solved. Therefore, in the proposed RHC approach, the control is updated whenever asynchronous events occur. This prevents unnecessary steps to re-solve the RHCP (i.e., (21)-(22)) unlike time-driven receding horizon control.

In general, the determination of the action horizon h may be *controllable* or *uncontrolled*. The latter case occurs as a result of random or external events in the system (if such events are allowed), while the former corresponds to the occurrence of any one event resulting from an agent finishing the execution of a RHCP solution determined at an earlier time. We next define two *controllable* events associated with an agent when it resides at target i. Both of these events define the action horizon h based on the RHCP solution $U^*_i(t) = [u^*_i(t), j^*(t), u^*_j(t)]$ obtained by the agent at time $t \in [0, T]$:

1. Event $[h \rightarrow u^*_i]$: This event occurs at time $t + u^*_i(t)$ and indicates the end of the active time at target i. By definition, this coincides with an agent departure event from i.

2. Event $[h \rightarrow \rho_{ij^*}]$: This event occurs at time $t + \rho_{ij^*(t)}$ and is only feasible after an event $[h \rightarrow u^*_i]$ has occurred (including the possibility that $u^*_i(t) = 0$). Clearly, this coincides with an agent arrival event at target $j^*(t)$.

Among these two types of events, only one is feasible at any one time. However, it is also possible for a different event to occur after t, before one of these two events occurs. Such an event is either external, random (if our model allows for such events) or is controllable but associated with a different target than i. In particular, let us define two additional events that may occur at any neighboring target $j \in \mathscr{N}_i$ and affect the agent residing at target i. These events aim to ensure the control constraint (6) (to prevent simultaneous target sharing) and apply only to *multi-agent* persistent monitoring problems.

At time t, if a target $j \in \mathscr{V}$ already has a residing agent or if an agent is en route to visit it from a neighboring target in $\mathscr{N}_j$, it is said to be *covered*. Now, an agent $a \in \mathscr{A}$ residing in target i can prevent simultaneous target sharing at $j \in \mathscr{N}_i$ by simply modifying the neighbor set $\mathscr{N}_i$ used in its RHCP solved at time t to exclude all such covered targets. Let us use $\mathscr{N}_i(t)$ to indicate a *time-varying* neighbor set of i. Then, if target j becomes covered at t, we set $\mathscr{N}_i(t) = \mathscr{N}_i(t^-) \backslash \{j\}$. Most importantly, note that as soon as an agent a is en route to j^*, j^* becomes covered - preventing any other agent from visiting j^* prior to agent a's subsequent departure from j^*.

Based on this discussion, we define the following two additional *neighbor-induced local events* at $j \in \mathscr{N}_i$ affecting an agent a residing at target i:

3. Covering Event C_j, $j \in \mathscr{N}_i$: This event causes $\mathscr{N}_i(t)$ to be modified to $\mathscr{N}_i(t)\backslash\{j\}$.

4. Uncovering Event $\bar{C}_j$, $j \in \mathscr{N}_i$: This event causes $\mathscr{N}_i(t)$ to be modified to $\mathscr{N}_i(t) \cup \{j\}$.

If one of these two events takes place while an agent remains active at target i (i.e., prior to the occurrence of event $[h \rightarrow u_i^*]$), then the RHCP is re-solved to account for the updated $\mathscr{N}_i(t)$. This may affect the optimal solution's values U_i^* compared to the previous solution. Note, however, that the new solution will still give rise to a subsequent event $[h \rightarrow u_i^*]$.

*e) **Two Forms of RHCPs**:* The exact form of the RHCP ((21)-(22)) that needs to be solved at a certain event time depends on the event that triggered the end of the previous action horizon. In particular, corresponding to the two controllable event types, there are two possible RHCP forms:

1. RHCP1: This problem is solved by an agent when an event $[h \rightarrow \rho_{ki}]$ occurs at time t at target i for any $k \in \mathscr{N}_i(t)$, i.e., at the arrival of the agent at target i. The solution $U_i^*(t)$ includes $u_i^*(t) \geq 0$, representing the active time to be spent at i. This problem may also be solved while the agent is active at i if a C_j or $\bar{C}_j$ event occurs at any neighbor $j \in \mathscr{N}_i(t)$.

2. RHCP2: This problem is solved by an agent residing at target i when an event $[h \rightarrow u_i^*]$ occurs at time t. The solution $U_i^*(t)$ is now constrained to include $u_i^*(t) = 0$ by default, implying that the agent must immediately depart from i.

The interconnection between different events and RHCPs involved in the proposed RHC method is illustrated in Fig. 3.

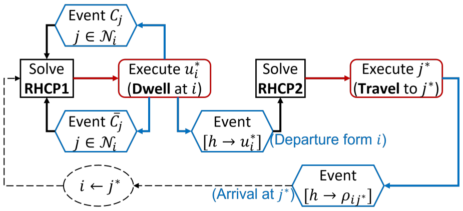


Fig. 3: The proposed event-driven receding horizon control (RHC) strategy (focusing on an agent at a target $i \in \mathscr{V}$).

V. Solving Event-Driven RHCPs

A. Solution of RHCP2

We begin with **RHCP2** as it is the simplest RHCP given that in this case $u_i^*(t) = 0$ by default. Hence, U_{ij} in (21) is limited to $U_{ij} = u_j$ and the planning horizon length $w(U_{ij})$ in (20) becomes $w(U_{ij}) = \rho_{ij} + u_j$. Based on the control constraints: $w(U_{ij}) = \rho_{ij} + u_j \leq H$ and $u_j \geq 0$, any target $j \in \mathscr{N}_i(t)$ such that $\rho_{ij} > H$ will not result in a feasible dwell-time value u_j. Thus, such targets are directly omitted from (21).

*a) **Constraints**:* According to the above discussion, note that in this **RHCP2**, u_j is constrained by $0 \leq u_j \leq H - \rho_{ij}$.

*b) **Objective**:* According to (23), the objective function of **RHCP2** is $\bar{J}_i(t, t+w)$. To obtain an exact expression for $\bar{J}_i$, it is decomposed using (24) as

$$\bar{J}_i = -\frac{J_j^A + \sum_{k \in \bar{\mathscr{N}}_i(t)\backslash\{j\}} J_k^A}{(J_j^A + J_j^I) + \sum_{k \in \bar{\mathscr{N}}_i(t)\backslash\{j\}} (J_k^A + J_k^I)} = -\frac{J_j^A}{J_j^A + J_j^I + \sum_k J_k^I}, \tag{25}$$

where the last equality follows from the fact that $J_k^A = J_k^A(t, t+w) = 0$ as any target $k \in \bar{\mathscr{N}}_i\backslash\{j\}$ will not be visited during the planning horizon $[t, t+w]$ (e.g., see Fig. 4). Similarly, we can write $J_j^A = J_j(t+\rho_{ij}, t+\rho_{ij}+u_j)$, $J_j^I = J_j(t, t+\rho_{ij})$ and $J_k^I = J_k(t, t+\rho_{ij}+u_j)$. Each of these terms can be evaluated using Lemmas 2 and 3.

These results together with (25) give the objective function $\bar{J}_i(t, t+w)$ required for **RHCP2** in the form $\bar{J}_i(u_j)$ (with a slight abuse of notation) as

$$\bar{J}_i(u_j) = -\frac{A(u_j)}{A(u_j) + B(u_j)}, \tag{26}$$

$$A(u_j) \triangleq J_j^A = c_1 + c_2 \log(1 + c_3 e^{-\lambda_j u_j}) + c_4 u_j, \tag{27}$$

$$B(u_j) \triangleq J_j^I + \sum_k J_k^I = c_5 + c_6 u_j + \sum_k c_{7k} e^{2A_k u_j}, \tag{28}$$

with $\lambda_j = 2\sqrt{A_j^2 + Q_j G_j}$ and

$$\begin{aligned}
&c_1 = -c_2 \log(1 + c_3), \quad c_2 = \frac{1}{G_j}, \quad c_3 = -\frac{G_j \Omega_j' + Q_j v_{j2}}{G_j \Omega_j' + Q_j v_{j1}}, \\
&\Omega_j' = \Omega_j(t + \rho_{ij}), \quad v_{j1}, v_{j2} = \frac{1}{Q_j}(-A_j \pm \sqrt{A_j^2 + Q_j G_j}), \\
&c_4 = \frac{1}{v_{j1}}, \quad c_5 = \frac{1}{2A_j}\left(\Omega_j + \frac{Q_j}{2A_j}\right) \times (e^{2A_j \rho_{ij}} - 1) - \frac{Q_j \rho_{ij}}{2A_j} \\
&\quad -\sum_k \frac{1}{2A_k}\left(\Omega_k + \frac{Q_k}{2A_k} + Q_k \rho_{ij}\right), \quad \Omega_j = \Omega_j(t), \\
&\Omega_k = \Omega_k(t), \quad c_6 = -\sum_k \frac{Q_k}{2A_k}, \quad c_{7k} = \frac{1}{2A_k}\left(\Omega_k + \frac{Q_k}{2A_k}\right) e^{2A_k \rho_{ij}}.
\end{aligned} \tag{29}$$

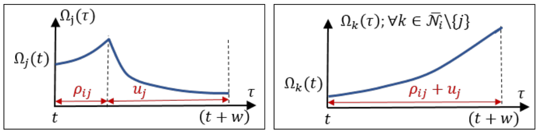


Fig. 4: Example state trajectories during $[t, t+w)$ for **RHCP2**.

*c) **Unimodality of $\bar{J}_i(u_j)$**:* We first prove the following lemma to eventually establish the unimodality of $\bar{J}_i(u_j)$.

Lemma 4: The **RHCP2** objective function $\bar{J}_i(u_j)$ (26) satisfies the following properties:

$$\lim_{u_j \rightarrow 0} \bar{J}_i(u_j) = 0 \quad \text{and} \tag{30}$$

$$\lim_{u_j \rightarrow \infty} \bar{J}_i(u_j) = \begin{cases} L_i & \text{if } A_k < 0, \forall k \in \bar{\mathscr{N}}_i\backslash\{j\}, \\ 0 & \text{otherwise,} \end{cases} \tag{31}$$

where $L_i = -1/(1 + \frac{c_6}{c_4})$ (with c_4, c_6 are as defined in (29)), and all the limits are approached from below.

Proof: To establish these two results, we exploit the $A(u_j), B(u_j)$ notation introduced in (26). The result in (30) is proved using the relationships: $\lim_{u_j \rightarrow 0} A(u_j) = 0$ (note that, from (29): $c_1 + c_2 \log(1 + c_3) = 0$) and $\lim_{u_j \rightarrow 0} B(u_j) = c_5 + \sum_k c_{7k} \geq 0$. Similarly, (31) is proved using the limit (given by L'Hospital's rule):

$$\lim_{u_j \rightarrow \infty} \frac{B(u_j)}{A(u_j)} = \begin{cases} \frac{c_6}{c_4} & \text{if } A_k < 0, \forall k \in \bar{\mathscr{N}}_i\backslash\{j\}, \\ \infty & \text{otherwise.} \end{cases}$$

The directions in which each of these limits are approached can be established using the inequalities: $A(u_j) \geq 0$, $B(u_j) > 0$ (since $A(u_j)$ and $B(u_j)$ defined respectively in (27) and (28) are contributions of the targets and the travel time $\rho_{ij} > 0$) and $\bar{J}_i(u_j) \leq 0$ (based on the definition in (26)), for all $u_j \geq 0$. ■

Second, in the following lemma, we establish two possible steady-state target error covariance values.

Lemma 5: If a target $i \in \mathscr{V}$ is sensed by an agent for an infinite duration of time, its error covariance $\Omega_i(t)$ is such that

$$\lim_{t\to\infty} \Omega_i(t) = \Omega_{i,ss} \triangleq \frac{Q_i}{-A_i + \sqrt{A_i^2 + Q_i G_i}}. \tag{32}$$

On the other hand, if a target $i \in \mathscr{V}$ is not sensed by an agent for an infinite duration of time,

$$\lim_{t\to\infty} \Omega_i(t) = \begin{cases} \bar{\Omega}_{i,ss} \triangleq -\frac{Q_i}{2A_i} & \text{if } A_i < 0, \\ \infty & \text{if } A_i > 0. \end{cases} \tag{33}$$

Proof: The relationships in (32) and (33) can be obtained by simply evaluating the limit $t \to \infty$ of the $\Omega_i(t)$ expressions proved in Lemma 2 and Lemma 3, respectively. ■

Note that $\Omega_{i,ss}$ and $\bar{\Omega}_{i,ss}$ respectively defined in (32) and (33) are two fixed characteristic values of a target $i \in \mathscr{V}$ where: 1) $0 < \Omega_{i,ss} < \bar{\Omega}_{i,ss}$ if $A_i < 0$, and 2) $\bar{\Omega}_{i,ss} < 0 < \Omega_{i,ss}$ if $A_i > 0$.

Assumption 1: The initial error covariance value $\Omega_i(0)$ of a target $i \in \mathscr{V}$ is such that: 1) $\Omega_i(0) \in (\Omega_{i,ss},\ \bar{\Omega}_{i,ss})$ if $A_i < 0$, and 2) $\Omega_i(0) \in (\Omega_{i,ss},\ \infty)$ if $A_i > 0$.

The mildness of the above assumption can be justified using the steady state target error covariance values established in Lemma 5. In particular, note that this assumption is violated by a target $i \in \mathscr{V}$ only if 1) $\Omega_i(0) < \Omega_{i,ss}$, or 2) $\Omega_i(0) > \bar{\Omega}_{i,ss}$ with $A_i < 0$. In the first case, based on (32), there should exist a finite time $\bar{t}_{i,0} > 0$ where $\Omega_i(\bar{t}_{i,0}) > \Omega_{i,ss}$ occurs - if the target i was not sensed by an agent during a finite interval $[t_{i,0},\ \bar{t}_{i,0}] \subseteq [0,\ \bar{t}_{i,0}]$. In the second case, based on (33) (with $A_i < 0$), there should exist a finite time $\bar{t}_{i,0} > 0$ where $\Omega_i(\bar{t}_{i,0}) < \bar{\Omega}_{i,ss}$ occurs - if the target i was sensed by an agent during a finite interval $[t_{i,0},\ \bar{t}_{i,0}] \subseteq [0, \bar{t}_{i,0}]$. This implies that even if Assumption 1 is violated at some target $i \in \mathscr{V}$ (at the initial time $t = 0$), we can enforce it at an alternative initial time $t = \bar{t}_{i,0}$ simply by temporarily regulating agent visits to the target i. Finally, we point out that $0 < \bar{t}_{i,0} \ll T$ due to the exponentially fast error covariance dynamics proved in (16) and (18).

Lemma 6: Under Assumption 1, the error covariance value $\Omega_i(t)$ of a target $i \in \mathscr{V}$ at any time $t \geq 0$ satisfies: 1) $\Omega_i(t) \in (\Omega_{i,ss},\ \bar{\Omega}_{i,ss})$ if $A_i < 0$, and 2) $\Omega_i(t) \in (\Omega_{i,ss},\ \infty)$ if $A_i > 0$.

Proof: The proof follows directly from the results established in Lemma 5. This is because, under Assumption 1, the proved limiting values in (32) and (33) can respectively be considered as minimum and maximum achievable $\Omega_i(t)$ values. Note that this result holds irrespective of how target i is sensed by the agents, i.e., irrespective of the form of $\eta_i(t)$ signal in (3). ■

Remark 2: Irrespective of Assumption 1, using the Lyapunov stability analysis [32], it can be shown that, under arbitrarily switched $\eta_i(t)$ signals (agent visits), the *closed* intervals $[\Omega_{i,ss},\ \bar{\Omega}_{i,ss}]$ and $[\Omega_{i,ss},\ \infty]$ are globally attractive positively invariant sets for the error covariance dynamics (3) under $A_i < 0$ and $A_i > 0$, respectively. Moreover, the said attractiveness can be proved to be a *finite-time* attractiveness to the corresponding *open* intervals if we *omit* the switching signals (agent visits) of the form: 1) $\eta_i(t) = 0, \forall t \geq 0$ if $A_i < 0$ and $\Omega_i(0) > \bar{\Omega}_{i,ss}$, and 2) $\eta_i(t) = 1, \forall t \geq 0$ if $A_i > 0$ and $\Omega_i(0) < \Omega_{i,ss}$, respectively. In other words, by temporarily regulating agent visits to a target $i \in \mathscr{V}$ where Assumption 1 is violated, we still can ensure the statement in Lemma 6 for that target i, but for any time $t \geq \bar{t}_{i,0}$ where $0 < \bar{t}_{i,0} \ll T$.

Lemma 7: Under Assumption 1, for any target $i \in \mathscr{V}$ and time $t \geq 0$, $2\Omega_i(t)A_i + Q_i > 0$.

Proof: First, note that in general $\Omega_i(t) > 0$ and $Q_i > 0$. Therefore, $A_i > 0 \implies 2\Omega_i(t)A_i + Q_i > 0$. On the other hand, if $A_i < 0$, according to Lemma 6, $\Omega_i(t) < \bar{\Omega}_{i,ss}$, i.e., $\Omega_i(t) < -\frac{Q_i}{2A_i}$ as $\bar{\Omega}_{i,ss} = -\frac{Q_i}{2A_i}$ (from (33)). Therefore, $A_i < 0 \implies \Omega_i(t) < -\frac{Q_i}{2A_i} \iff 2\Omega_i(t)A_i + Q_i > 0$. This completes the proof. ■

We are now ready to prove the unimodality of $\bar{J}_i(u_j)$.

Theorem 1: Under Assumption 1, the objective function of **RHCP2** (i.e., $\bar{J}_i(u_j)$ in (26)) is unimodal.

Proof: Again we exploit the $A(u_j), B(u_j)$ notation introduced in (26). As argued in the proof of Lemma 4, $A(u_j) \geq 0$, $B(u_j) > 0$ and $\bar{J}_i(u_j) \leq 0$ for all $u_j \geq 0$. Now, according to the limits of $\bar{J}_i(u_j)$ established in Lemma 4, it is clear that $\bar{J}_i(u_j)$ has at least one or more local minimizers.

Through differentiating (26), we can obtain an equation for the stationary points of $\bar{J}_i(u_j)$ as: $\frac{d\bar{J}_i(u_j)}{du_j} = 0 \iff A(u_j)\frac{dB(u_j)}{du_j} - B(u_j)\frac{dA(u_j)}{du_j} = 0$. For notational convenience, let us re-state this equation as $AB' - BA' = 0$. Using the same notation, the second derivative of $\bar{J}_i(u_j)$ can be written as $\frac{d^2\bar{J}_i(u_j)}{du_j^2} = \frac{AB'' - BA''}{(A+B)^2} - \frac{(AB'-BA')(A'+B')}{(A+B)^3}$. Therefore, the nature of a stationary point of $\bar{J}_i(u_j)$ is determined by the sign of the term $AB'' - BA''$. Since we already know $A \geq 0$ and $B > 0$, let us focus on the A'' and B'' terms.

Using the $A(u_j)$ expression in (27), we can write

$$A'' = \frac{d^2A(u_j)}{du_j^2} = \frac{c_3 c_2 \lambda_j^2 e^{\lambda_j u_j}}{(c_3 + e^{\lambda_j u_j})^2}. \tag{34}$$

From (29), $c_2 > 0$ and $c_3 < 0 \iff \Omega_j(t + \rho_{ij}) > -\frac{Q_j v_{j2}}{G_j} = \frac{1}{v_{j1}} = \Omega_{j,ss}$ (the last two steps respectively used the relationships $v_{j1}v_{j2} = -\frac{G_j}{Q_j}$ and (32)). Since $\Omega_j(t + \rho_{ij}) > \Omega_{j,ss}$ (from Lemma 6), $c_3 < 0$ and thus (34) implies $A'' < 0$ for all $u_j \geq 0$.

Using the $B(u_j)$ expression in (28), we can write

$$B'' = \frac{d^2B(u_j)}{du_j^2} = \sum_{k \in \bar{\mathscr{N}}_i \backslash \{j\}} (2\Omega_k(t)A_k + Q_k)e^{2A_k(\rho_{ij}+u_j)}. \tag{35}$$

Notice that $(2\Omega_k(t)A_k + Q_k) > 0, \forall t \geq 0, k \in \bar{\mathscr{N}}_i \backslash \{j\}$ (from Lemma 7) and $e^{2A_k(\rho_{ij}+u_j)} > 0$ for all $u_j \geq 0$. Therefore, (35) implies $B'' > 0$ for all $u_j \geq 0$.

So far, we have shown that $A \geq 0$, $B > 0$, $B'' > 0$ while $A'' < 0$, $\forall u_j \geq 0$. Therefore, $AB'' - BA'' > 0$ for all $u_j \geq 0$. Hence, all the stationary points of $\bar{J}_i(u_j)$ should be local minimizers. Since $\bar{J}_i(u_j)$ and all its derivatives are continuous, it cannot have two (or more) local minimizers without having a local maximizer(s). Therefore, $\bar{J}_i(u_j)$ has only one stationary point which is the global minimizer and thus $\bar{J}_i(u_j)$ is unimodal. ■

d) ***Solving RHCP2 for optimal control u_j^*:*** The solution $U_{ij}^* = [u_j^*]$ of (21) is given by u_j^* where

$$u_j^* = \underset{0 \leq u_j \leq H - \rho_{ij}}{\arg\min}\ \bar{J}_i(u_j). \tag{36}$$

Since the above objective function $\bar{J}_i(u_j)$ is unimodal (from Theorem 1) and its feasible space is convex, we use the *projected gradient descent* [33] algorithm to efficiently obtain the *globally* optimal control decision u_j^*.

*e) **Solving for Optimal Next-Visit Target** j^*:* Using the obtained u_j^* values in (36) for all $j \in \mathscr{N}_i(t)$, we now know the optimal trajectory costs $\bar{J}_i(u_j^*), \forall j \in \mathscr{N}_i(t)$. Based on (22), the optimal target to visit next is $j^* = \arg\min_{j\in\mathscr{N}_i(t)} \bar{J}_i(u_j^*)$.

Thus, upon solving **RHCP2**, agent a departs from target i at time t and follows the path $(i, j^*) \in \mathscr{E}$ to visit target j^*. In the spirit of RHC, recall that the optimal control will be updated upon the occurrence of the next event, which, in this case, will be the arrival of the agent at j^*, triggering the solution of an instance of **RHCP1** at j^*.

*B. Solution of **RHCP1***

We next consider the **RHCP1**, which is the most general version among the two RHCP forms. In **RHCP1**, U_{ij} in (21) is directly $U_{ij} = [u_i, u_j]$ and the planning horizon w is the same as in (20), where $w(U_{ij}) = u_i + \rho_{ij} + u_j$.

*a) **Constraints**:* Based on the control constraints in (21), note that in this **RHCP1**, u_i and u_j are constrained by $0 \leq u_i,\ \ 0 \leq u_j$ and $u_i + u_j \leq H - \rho_{ij}$, where the last constraint follows form $w(U_{ij}) \leq H$.

*b) **Objective**:* According to (23), the objective function of **RHCP1** is $\bar{J}_i(t, t+w)$. To obtain an exact expression for $\bar{J}_i$, it is decomposed using (24) as

$$\begin{aligned}\bar{J}_i &= -\frac{J_i^A + J_j^A + \sum_{k\in\mathscr{N}_i(t)\backslash\{j\}} J_k^A}{(J_i^A + J_i^I) + (J_j^A + J_j^I) + \sum_{k\in\mathscr{N}_i(t)\backslash\{j\}}(J_k^A + J_k^I)}\\ &= -\frac{J_i^A + J_j^A}{(J_i^A + J_j^A) + (J_i^I + J_j^I + \sum_{k\in\mathscr{N}_i(t)\backslash\{j\}} J_k^I)}. \qquad (37)\end{aligned}$$

Similar to (25), note that each term in (37) is also evaluated over the planning horizon $[t, t+w]$. Therefore $J_k^A = J_k^A(t, t+w) = 0$ as any target $k \in \mathscr{N}_i \backslash \{j\}$ will not be visited during the planning horizon (e.g., see Fig. 5). Moreover, we can write $J_i^A = J_i(t, t+u_i)$, $J_j^A = J_j(t+u_i+\rho_{ij}, t+u_i+\rho_{ij}+u_j)$, $J_i^I = J_i(t+u_i, t+u_i+\rho_{ij}+u_j)$, $J_j^I = J_j(t, t+u_i+\rho_{ij})$ and $J_k^I = J_k(t, t+u_i+\rho_{ij}+u_j)$. Each of these terms can be evaluated using Lemmas 2 and 3. These results together with (37) give the objective function $\bar{J}_i(t, t+w)$ required for **RHCP2** in the form $\bar{J}_i(u_i, u_j)$ (again with a slight abuse of notation) as

$$\bar{J}_i(u_i, u_j) = -\frac{A(u_i, u_j)}{A(u_i, u_j) + B(u_i, u_j)}, \qquad (38)$$

where $A(u_i, u_j) \triangleq J_i^A + J_j^A$, $B(u_i, u_j) \triangleq J_i^I + J_j^I + \sum_{k\in\mathscr{N}_i\backslash\{j\}} J_k^I$. Specifically, $A(u_i, u_j)$ and $B(u_i, u_j)$ take the following forms:

$$\begin{aligned}A(u_i, u_j) =& a_1 + a_2\log(1 + a_3 e^{-\lambda_i u_i})\\ &+ a_4\log(1 + a_5 e^{2A_j u_i} + a_6 e^{-\lambda_j u_j} + a_7 e^{2A_j u_i - \lambda_j u_j})\\ &+ a_8 u_i + a_9 u_j, \qquad (39)\\ B(u_i, u_j) =& b_1 + b_2 u_i + b_3 u_j + b_4 e^{2A_j u_i} + b_5 e^{2A_i u_j}\\ &+ \sum_{k\in\mathscr{N}_i\backslash\{j\}} b_{6k} e^{2A_k(u_i+u_j)} + C(u_i, u_j), \qquad (40)\\ C(u_i, u_j) =& c_1\left[\frac{1 + c_2 e^{-\lambda_i u_i} + c_3 e^{2A_i u_j} + c_4 e^{-\lambda_i u_i + 2A_i u_j}}{1 + c_5 e^{-\lambda_i u_i}}\right],\end{aligned}$$

where the coefficients $a_l, b_l, c_l, \forall l$ present in (39) and (40) are given in appendix A.

*c) **Unimodality of** $\bar{J}_i(u_i, u_j)$:* Proving the unimodality of $\bar{J}_i(u_i, u_j)$ is a challenging task due to the complexity of the $A(u_i, u_j)$ and $B(u_i, u_j)$ expressions in (38). However, we establish that $\bar{J}_i(u_i, u_j)$ is unimodal along the lines $u_i = 0$ and $u_j = 0$. Further, we show that $\bar{J}_i(u_i, u_j) \to 0$ whenever $(u_i, u_j) \to (0, 0)$, $u_i \to \infty$ or $u_j \to \infty$. Based on these theoretical observations and the experimental results (see Fig. 6), we conjecture that $\bar{J}_i(u_i, u_j)$ is unimodal. However, to date, we have not provided a formal proof of this.

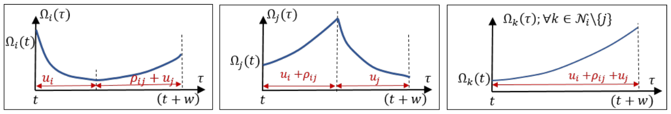


Fig. 5: Example state trajectories during $[t, t+w)$ for **RHCP1**.

Lemma 8: The **RHCP1** objective function $\bar{J}_i(u_i, u_j)$ satisfies the following properties:

$$\begin{gathered}\lim_{(u_i,u_j)\to(0,0)} \bar{J}_i(u_i, u_j) = 0, \qquad \lim_{(u_i,u_j)\to(\infty,\infty)} \bar{J}_i(u_i, u_j) = 0\\ \lim_{u_i\to\infty} \bar{J}_i(u_i, 0) = \begin{cases} L_i & \text{if } A_k < 0, \forall k \in \mathscr{N}_i \backslash \{j\},\\ 0 & \text{otherwise,}\end{cases} \qquad (41)\\ \lim_{u_j\to\infty} \bar{J}_i(0, u_j) = \begin{cases} L_j & \text{if } A_k < 0, \forall k \in \mathscr{N}_i \backslash \{j\},\\ 0 & \text{otherwise,}\end{cases}\end{gathered}$$

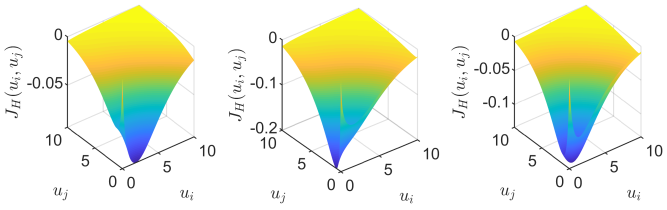
where $L_i = -1/(1 + \frac{b_2}{a_8})$ and $L_j = -1/(1 + \frac{b_3}{a_9})$ (with b_2, a_8, b_3, a_9 are as defined in Appendix A).

Proof: The result in (41) can be obtained by following the same steps used in the proof of Lemma 4. ■

Theorem 2: Under Assumption 1, the functions $\bar{J}_i(u_i, 0)$ and $\bar{J}_i(0, u_j)$ are unimodal.

Proof: This proof basically follows the same steps as the proof of Theorem 1. As an example, let us consider proving the unimodality of $\bar{J}_i(u_i, 0)$. First, $\bar{J}_i(u_i, 0)$ is written as $\bar{J}_i(u_i, 0) = -\bar{A}(u_i)/(\bar{A}(u_i) + \bar{B}(u_i))$ where $\bar{A}(u_i) = A(u_i, 0)$ and $\bar{B}(u_i) = B(u_i, 0)$. Therefore, similar to before, the nature of the stationary points of $\bar{J}_i(u_i, 0)$ is dependent on the sign of $\bar{A}\bar{B}'' - \bar{B}\bar{A}''$. Next, using $A(u_i, u_j)$ and $B(u_i, u_j)$ expressions in (39) and (40), we can derive $\bar{B}'' = \frac{d^2B(u_i,0)}{du_i^2}$ and $\bar{A}'' = \frac{d^2A(u_i,0)}{du_i^2}$ in terms of the the coefficients shown in Appendix A. Finally, using Lemma 6 and Lemma 7 (under Assumption 1), it can be proven that $\bar{B}'' > 0$ and $\bar{A}'' < 0, \forall u_i \geq 0$. Since $\bar{A} \geq 0$, $\bar{B} > 0, \forall u_i \geq 0$, we now can conclude that $\bar{A}\bar{B}'' - \bar{B}\bar{A}'' > 0, \forall u_i \geq 0$.

This result, together with the limits established in Lemma 8 implies that there exists only one stationary point in $\bar{J}_i(u_i, 0)$, which is the global minimizer. Further, since $\bar{J}_i(u_i, 0)$ and all of its derivatives are continuous, it can also be concluded that $\bar{J}_i(u_i, 0)$ is a unimodal function. Following the same steps, the unimodality of $\bar{J}_i(0, u_j)$ can also be established. ■



(a) $u_j^* = 0$ (b) $u_i^* = 0$ (c) $u_i^*, u_j^* > 0$

Fig. 6: Three example cases of **RHCP1** objective function $\bar{J}_i(u_i, u_j)$ plots (location of the minimizer: (u_i^*, u_j^*)).

*d) **Solving RHCP1 for Optimal Controls** u_i^*, u_j^*:* The solution $U_{ij}^* = [u_i^*, u_j^*]$ of (21) is given by (u_i^*, u_j^*) where

$$\begin{aligned}(u_i^*, u_j^*) = &\underset{u_i,u_j}{\arg\min}\ \bar{J}_i(u_i,u_j),\\ &0 \le u_i,\ 0 \le u_j,\\ &u_i + u_j \le H - \rho_{ij}\end{aligned} \tag{42}$$

In (42), the feasible space is convex and we already conjectured that the objective function is unimodal. Therefore, again, we use the projected gradient descent algorithm [33] to efficiently obtain the optimal control decisions (u_i^*, u_j^*).

*e) **Solving for Optimal Next-Visit Target** j^*:* Using the obtained $U_{ij}^* = [u_i^*, u_j^*]$ values in (42) for all $j \in \mathcal{N}_i(t)$, we now have at our disposal the optimal trajectory costs $\bar{J}_i(u_i^*, u_j^*)$ for all $j \in \mathcal{N}_i(t)$. Based on (22), the optimal neighbor to plan as the next-visit target is given by $j^* = \arg\min_{j\in\mathcal{N}_i(t)} \bar{J}_i(u_i^*, u_j^*)$.

Upon solving **RHCP1**, the agent remains stationary (active) on target i for a duration of u_i^* or until any other event occurs. If the agent completes the determined active time u_i^* (i.e., if the corresponding event $[h \rightarrow u_i^*]$ occurs), the agent will have to subsequently solve an instance of **RHCP2** to determine the next-visit target and depart from target i. However, if a different event occurred before the anticipated event $[h \rightarrow u_i^*]$, the agent will have to re-solve **RHCP1** to re-compute the remaining active time at target i.

Remark 3: In the proposed RHC solution, there are only three tunable parameters: 1) the upper bound H to the planning horizon, 2) the gradient descent step size, and 3) the gradient descent initial condition. For H, as stated before, selecting a sufficiently large value (e.g., $H = T - t$) ensures that it does not affect the RHC solutions. For the gradient descent step size, there are established standard choices [33]. Finally, the established unimodality properties imply that RHC solutions are not affected by the gradient descent initial condition.

VI. Improving Computational Efficiency Using Machine Learning

Recall that when solving a RHCP, an agent $a \in \mathscr{A}$ (residing on a target $i \in \mathscr{V}$ at some time $t \in [0,T]$) has to solve the optimization problems (21)-(22). The problem in (21) involves solving $|\mathcal{N}_i|$ different optimization problems (one for each neighbor $j \in \mathcal{N}_i$) to get the optimal *continuous* (real-valued) controls: $\{U_{ij}^* : j \in \mathcal{N}_i\}$. The problem in (22) is only a simple numerical comparison that determines the optimal *discrete* control: the next-visit $j^* \in \mathcal{N}_i$. Upon solving this RHCP, the agent will only use $U_{ij^*}^*$ and j^* to make its immediate decisions. Hence, the continuous controls: $\{U_{ij}^* : j \in \mathcal{N}_i \backslash \{j^*\}\}$ found when solving (21) are wastefully discarded. This motivates the use of derived information $\{U_{ij}^* : j \in \mathcal{N}_i\}$ (part of which will surely be discarded) to introduce a learning component as explained next.

Note that if j^* can be determined ahead of solving (21), we can prevent this waste of computational resources by limiting the evaluation of (21) only for the pre-determined neighbor j^* to directly get $U_{ij^*}^*$. Roughly speaking, this approach should save a fraction $\frac{|\mathcal{N}_i|-1}{|\mathcal{N}_i|}$ of the processing (CPU) time required to solve the RHCP (i.e., (21)-(22)).

*a) **Ideal Classification Function:*** The goal is to approximate an ideal classification function $F_i : \mathbb{R}_{\geq 0}^{|\bar{\mathcal{N}_i}|} \rightarrow \mathcal{N}_i$ where

$$\begin{aligned} j^* = \ &F_i(X_i(t)) \triangleq \underset{j\in\mathcal{N}_i}{\arg\min}\ J_H\big(X_i(t), U_{ij}^*; w(U_{ij}^*)\big) \\ &= \underset{j\in\mathcal{N}_i}{\arg\min}\ J_H\big(X_i(t), \underset{U_{ij}\in\mathbb{U}}{\arg\min}\ J_H(X_i(t), U_{ij}; w(U_{ij})); w(U_{ij}^*)\big)\end{aligned} \tag{43}$$

where $X_i(t) = \{\Omega_j(t) : j \in \bar{\mathcal{N}_i}\}$ is the local state at target i at time t and $F_i(X_i(t))$ (explicitly expressed in the second line) is the result of combining equations (21) and (22). In the machine learning literature, this kind of an ideal classification function $F_i(X_i)$ is commonly known as an *underlying function* (or a *target function*) and X_i is considered as a *feature vector* [34].

We emphasize that F_i is strictly dependent on: (i) the current target $i \in \mathscr{V}$, (ii) the agent $a \in \mathscr{A}$ and (iii) the RHCP type. Therefore, F_i in actuality should be written as $F_i^{a,l}$ (where $l \in \{1,2\}$ represents the RHCP type) even though we omit doing so for notational simplicity.

*b) **Classifier Function:*** Due to the complexity of this ideal classification function $F_i(X_i)$ in (43), we cannot analytically simplify it to obtain a closed form solution for a generic input (feature) X_i. Therefore, we propose to use machine learning techniques to model $F_i(X_i)$ by an estimate of it - which we denote as $f_i(X_i; \mathscr{D}_i)$. Here, $\mathscr{D}_i$ represents a collected *data set* of size L and the notation $f_i(X_i; \mathscr{D}_i)$ implies that this *classifier function* has been constructed (trained) based on $\mathscr{D}_i$. Note that similar to F_i, both f_i and $\mathscr{D}_i$ depend not only on the target i but also on the agent a and the RHCP type l.

Since our aim is to develop a distributed on-line persistent monitoring solution, the agent a itself has to collect this data set $\mathscr{D}_i$ based on its very first L instants where a RHCP of type l was fully solved at target i. Specifically, $\mathscr{D}_i$ can be thought of as a set of input-output pairs: $\mathscr{D}_i = \{(X_i(\tau), j^*(\tau)) : \tau \in \Gamma_i^{a,l}\}$ where $\Gamma_i^{a,l}$ is the set of the first L event times where agent a fully solved a RHCP of type l while residing at target i.

*c) **Application of Neural Networks**:* In order to construct the classifier function $f_i(X_i; \mathscr{D}_i)$, among many commonly used classification techniques such as linear classifiers, support vector machines, kernel estimation techniques, etc., we chose an *Artificial Neural Networks* (ANN) based approach. This choice was made because of the key advantages that an ANN-based classification approach holds [34]: (i) generality, (ii) data-driven nature, (iii) non-linear modeling capability and (iv) the ability to provide posterior probabilities.

Let us denote a shallow feed forward ANN model as $y = h_i(x; \Theta)$ where x is the $|\bar{\mathcal{N}_i}|$-dimensional input feature vector and y (or $h_i(x; \Theta)$) is the $|\mathcal{N}_i|$-dimensional output vector under the ANN weight parameters Θ. For simplicity, we propose to use only one hidden layer with ten neurons with hyperbolic tangent sigmoid (*tansig*) activation functions. At the output layer, we propose to use the *softmax* activation function so that each component of the output (denoted as $h_{i,k}(x; \Theta)$, $k \in \mathcal{N}_i$) will be in the interval $(0,1)$.

Based on this ANN model, the classifier function is

$$\hat{j}^* = f_i(X_i; \mathscr{D}_i) = \underset{k\in\mathcal{N}_i}{\arg\max}\ h_{i,k}(X_i; \Theta^*), \tag{44}$$

where Θ^* represents the optimal set of ANN weights obtained by training the ANN model $y = h_i(x; \Theta)$ using the data set $\mathscr{D}_i$. Specifically, these optimal weights Θ^* are determined through back-propagation (and gradient descent) [34] such that the (standard) cross-entropy based cost function $H(\Theta)$ evaluated over the data set $\mathscr{D}_i = \{(X_i(\tau), j^*(\tau)) : \tau \in \Gamma_i^{a,l}\}$ given by

$$H(\Theta) = \frac{1}{L}\sum_{\tau\in\Gamma_i^{a,l}}\sum_{k\in\mathscr{N}_i}\left[\mathbf{1}\{j^*(\tau)=k\}\log\left(h_{i,k}(X_i(\tau);\Theta)\right)\right. \tag{45}$$
$$\left.+\mathbf{1}\{j^*(\tau)\neq k\}\log\left(1-h_{i,k}(X_i(\tau);\Theta)\right)\right] + \frac{\lambda}{2L}\|\Theta\|^2,$$

is minimized (λ represents the regularization constant).

*d) **RHC with Learning (RHC-L)**:* Needless to say, the optimal weights Θ^* (and hence the classifier function $f_i(X_i;\mathscr{D}_i)$ in (44)) are determined only when the agent a has accumulated a data set $\mathscr{D}_i$ of length L. In other words, the agent a has to be familiar enough with solving the RHCP type l at target i in order to learn $f_i(X_i;\mathscr{D}_i)$.

Upon learning $f_i(X_i;\mathscr{D}_i)$, the RHCP given in (21) and (22) can be solved very efficiently by simply evaluating:

$$\hat{j}^* = f_i(X_i(t);\mathscr{D}_i), \tag{46}$$
$$U_{i\hat{j}^*}^* = \underset{U_{i\hat{j}^*}\in\mathbb{U}}{\arg\min}\ J_H(X_i(t),U_{i\hat{j}^*};H), \tag{47}$$

to obtain the optimal controls $U_i^*(t) = \{U_{i\hat{j}^*}^*, \hat{j}^*\}$ (i.e., without having to solve (21) associated with targets $j \in \mathscr{N}_i\backslash\{\hat{j}^*\}$). For convenience, we call this approach the RHC-L method.

Notice that (47) (when compared to (21)) only involves a single continuous optimization problem - which may even be a redundant one to solve if the underlying RHCP is of type 2 (i.e., a **RHCP2**) where knowing the next-visit target (i.e., j^* now approximated by $\hat{j}^*$) is sufficient to take the immediate action. Therefore, the proposed RHC-L method can be expected to have significantly lower processing times for evaluating the RHCPs faced by the agents compared to the RHC method - upon the completion of the learning phase.

The only drawback in this RHC-L approach when compared to the original RHC method is the performance degradation that can be expected due to learning-related errors, i.e., due to the mismatch between $F_i(X_i)(=j^*)$ in (43) and its estimate $f_i(X_i;\mathscr{D}_i)(=\hat{j}^*)$ learned in (44).

*e) **RHC with Active Learning (RHC-AL)**:* We next propose a technique to suppress the aforementioned performance degradation that stems from the learning-related errors. For this purpose, we exploit the fact that ANN outputs are actually estimates of the *posterior probabilities* [34], i.e.,

$$h_{i,k}(X_i;\Theta^*) \simeq P(j^*=k|X_i), \tag{48}$$

where, $h_{i,k}(X_i;\Theta^*)$ is the output of the ANN corresponding to the neighbor $k \in \mathscr{N}_i$ and $P(j^*=k|X_i)$ is the probability of the ideal classification function $j^* = F_i(X_i)$ in (43) resulting $F_i(X_i) = k \in \mathscr{N}_i$, given the feature vector X_i. Note that $F_i(\cdot)$ here is an unknown function that we try to estimate and hence $j^*(=F_i(X_i))$ is a random variable.

Based on (48) and (44), the *mismatch error* between $F_i(X_i)$ and $f_i(X_i;\mathscr{D}_i)$ given the feature vector X_i can be estimated as $e_i(X_i) \triangleq P(F_i(X_i)\neq f_i(X_i;\mathscr{D}_i)|X_i) = 1-\max_{k\in\mathscr{N}_i}\ h_{i,k}(X_i;\Theta^*)$. Clearly, prior to solving the RHC-L problems (46) and (47), the agent can evaluate this mismatch error metric $e_i(X_i)$ and if it falls above a certain threshold, it can resort to follow the original RHC approach and solve (21) and (22), instead. Moreover, in such a case, the obtained RHC solutions can be incorporated into the data set $\mathscr{D}_i$ and re-train the ANN (to update the weight parameters Θ^* in $h_i(X_i;\Theta^*)$ (44)).

We call this "active learning" approach the RHC-AL method. It is important to highlight that this RHC-AL approach helps agents to make correct decisions in the face of unfamiliar scenarios. Therefore, we can expect the RHC-AL method to perform well compared to the RHC-L method - only at the expense of trading off the advantage that the RHC-L method had in terms of the processing times compared to the RHC method.

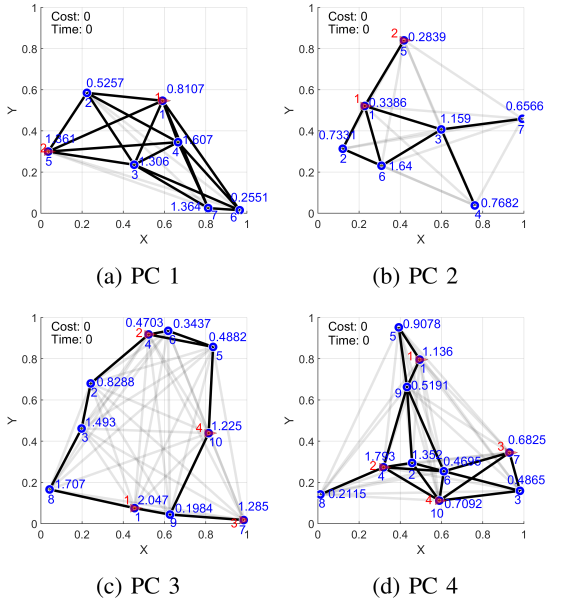
Remark 4: The proposed on-line learning process can alternatively be carried out off-line (if the system allows it) as each agent (for each target $i \in \mathscr{V}$ and each RHCP type) can synthetically generate data sets $\mathscr{D}_i$ exploiting the relationship (43) with a set of randomly generated features X_i. Moreover, if the agents are homogeneous, the proposed distributed learning process can be made centralized by allowing agents to share their data sets (pertaining to the same targets and RHCP types).

Remark 5: The proposed RHC-L and RHC-AL solutions in this section involve several tunable parameters related to the used: 1) ANN architecture, 2) ANN learning process, and 3) the data set (size). As stated before, in experiments, these were chosen to promote simplicity.

VII. Simulation Results

This section contains the details of three different simulation studies. In the first, we explore how the RHC-based agent control method performs compared to four other agent control techniques, in terms of the performance metric J_T in (7) evaluated over a relatively short period: $T = 50s$. Second, we study how well the persistent target state estimates provided by different agent control methods can facilitate local target state tracking control tasks. Finally, we explore the long-term performance of agent controllers by selecting $T = 750s$. In particular, we compare the agent control methods: RHC, RHC-L and RHC-AL in terms of the performance metric J_T and the average processing (CPU) time taken to solve each RHCP.

***Persistent Monitoring Problem Configurations**:* In this section, we consider the four randomly generated persistent monitoring problem configurations (PCs) shown in Fig. 7. In there, blue circles represent the targets and dark black lines indicate the trajectory segments that are available for the agents to travel between targets. Agents and target error covariance values at $t=0$ are represented by red triangles and yellow vertical bars/blue texts, respectively (see also Fig. 9 for PCs at $t=T$). The PCs 1,2 have seven targets and two agents each and the PCs 3,4 have ten targets and four agents each. In each PC, the target parameters were selected using the uniform distribution $U[\cdot,\cdot]$ as follows: $Y_i \sim U[0,1]$, $A_i \sim U[0.01,0.41]$, $B_i \sim U[0.01,0.41]$, $Q_i \sim U[0.1,2.1]$, $R_i \sim U[2,10]$ and we set $H_i = 1$ for all $i \in \mathscr{V}$. If the distance between any two targets is less than a certain threshold σ, a linear shaped trajectory segment was deployed between those targets. For PC 1, $\sigma = 0.7$ (dense) was used and for the reset, $\sigma = 0.45$ (sparse) was used. Each agent is assumed to travel with a unit speed on these trajectory segments and $H = 10$ is used.



(a) PC 1

(b) PC 2

(c) PC 3

(d) PC 4

Fig. 7: Four randomly generated persistent monitoring problem configurations (PCs) and their initial conditions.

A. Simulation Study 1: The effect of agent controls on target state estimation over a relatively short period.

In this section, we compare the performance metric J_T (defined in (7) with $T = 50$) observed for the four PCs shown in Fig. 7 when using five different agent control methods: (i) the centralized off-line periodic control (MTSP) method proposed in [10] (ii) a basic distributed on-line control (BDC) method (which is an ad-hoc control method), (iii) the proposed RHC method, (iv) a periodic version of the BDC (BDC-P) method and (v) a periodic version of the RHC (RHC-P) method. In essence, this simulation study is aimed to observe the effectiveness of the overall *target state estimation* process (as it is directly reflected by the metric J_T) rendered by the aforementioned different agent controllers. According to (3), the choice of target controls $\upsilon_i(t)$ in (1) do not affect J_T. Hence in this study, we set: $\upsilon_i(t) = 0,\ \forall i \in \mathcal{V},\ \forall t \in [0,T]$.

a) ***The Basic Distributed Control (BDC) Method****:* The BDC method uses the same event-driven control architecture as the RHC method. However, instead of u_i^* and j^* given in (21) and (22), it uses: $u_i^* = \arg\min_{\tau \geq 0} \mathbf{1}\{\Omega_i(t+\tau) \leq (1+\varepsilon)\Omega_{i,ss}\}$, and $j^* = \arg\max_{j \in \mathcal{N}_i(t)} \Omega_j(t)$ with $\varepsilon = 0.075$ and $\Omega_{i,ss}$ defined in (32). In all, the BDC method forces an agent to dwell at each visited target i until its error covariance $\Omega_i(t)$ drops to an ε fraction closer to the corresponding $\Omega_{i,ss}$ value, and then, the next-visit target is determined as the neighbor $j \in \mathcal{N}_i(t)$ with the maximum $\Omega_j(t)$ value.

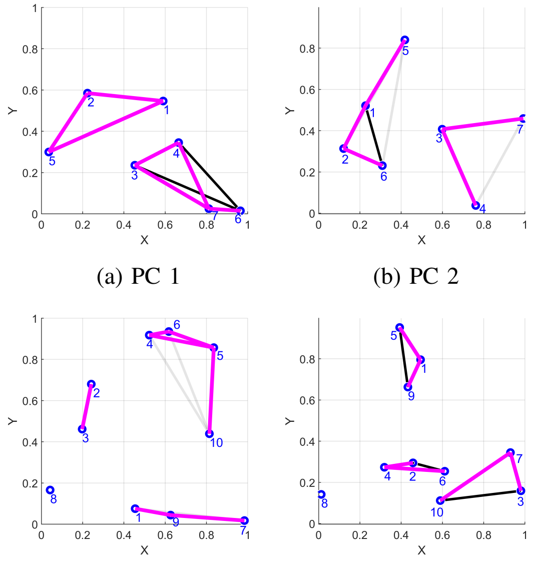
b) ***The Centralized Off-line Control (MTSP) Method [10]****:* Unlike the distributed on-line agent control methods: RHC and BDC, the MTSP method proposed in [10] fully computes the agent trajectories in a centralized off-line stage, focusing on minimizing an infinite horizon objective function:

$$\max_{i \in \mathcal{V}} \limsup_{t \to \infty} \operatorname{tr}(\Omega_i(t)) \tag{49}$$

via selecting periodic agent trajectories. However, this objective is in the same spirit of J_T (7) as it also aims to maintain the target error covariances as low as possible.

The MTSP method first uses the spectral clustering algorithm [35] to decompose the target topology $\mathcal{G}$ into sub-graphs among the agents. Then, on each sub-graph, starting from the *traveling salesman problem* (TSP) solution, a greedy target visitation cycle is constructed. Essentially, this set of target visitation cycles is a candidate solution for the famous multi-TSP [36] (hence the acronym: MTSP). Finally, the dwell-time spent at each target (on the constructed target visitation cycle) is found using a golden ratio search algorithm exploiting many interesting mathematical properties.

c) ***Hybrid Methods: BDC-P and RHC-P****:* In some applications, having a periodicity in visiting targets can be a crucial constraint (e.g., bus routes). Even in such cases, the proposed RHC method (or the BDC method) can still be used to make the dwell-time decisions at each visited target instead of using a fixed set of predetermined dwell-times like in the MTSP method. However, the optimal next-visit target, i.e., j^* in (22) (or in BDC method) would now be given by the off-line computed target visitation cycles (similar to the MTSP method). We use the label RHC-P (or BDC-P) to represent such a hybrid periodic agent control method. Note that in this RHC-P method, when solving for the dwell-times, (i.e., (21)), the RHCP objective (24) should only consider neighboring targets in the agent's target visitation cycle. Pertaining to the PCs shown in Fig. 7, target clusters and corresponding periodic target visitation cycles used by the periodic agent control methods (MTSP, BDC-P and RHC-P) are shown in Fig. 8.



(a) PC 1

(b) PC 2

(c) PC 3

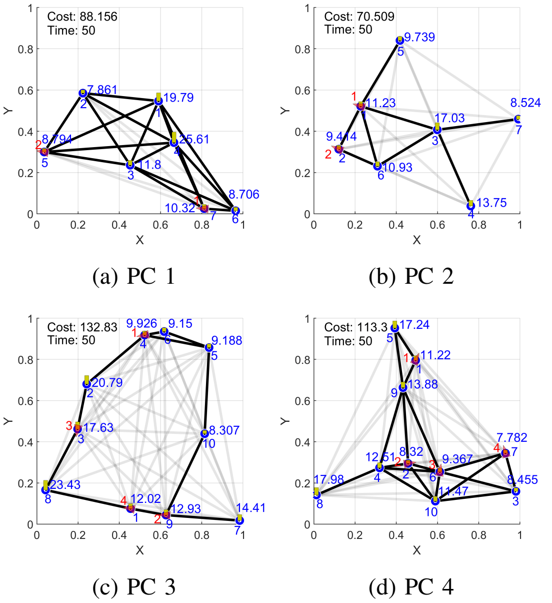
(d) PC 4

Fig. 8: Target clusters and periodic agent trajectories (magenta contours) used by the methods: MTSP, BDC-P and RHC-P.

Results and Discussion*:* Obtained results from the comparison are summarized in Tab. I. According to these results, on average, the RHC method has outperformed all the other agent control methods. It can be seen that the RHC-P method has the second-best performance level, and it has even performed slightly better than the RHC method in two cases. This observation is justifiable because the RHC-P method has a significant centralized and off-line component compared to the RHC method, which is completely distributed and on-line. Corresponding final states of the PCs given by the RHC method are shown in Fig. 9.

TABLE I: Performance comparison of target state estimation (i.e., J_T in (7)) under five agent control methods in four PCs.

Target State Estimator Performance (J_T)		Agent Control Mechanism				
		Off-line	Off-line/On-line		On-line	
		MTSP	BDC-P	RHC-P	BDC	RHC
PC No.	1	99.88	119.23	**84.41**	88.68	88.16
	2	90.08	155.25	77.80	101.75	**70.51**
	3	133.50	268.85	**128.90**	162.48	132.83
	4	187.88	231.28	123.70	174.32	**113.30**
Average:		127.83	193.65	103.70	131.81	**101.20**



(a) PC 1 (b) PC 2

(c) PC 3 (d) PC 4

Fig. 9: Final state of the PCs after using the RHC method.

The Worst-Case Performance: Inspired by the objective function (49) used in [10] (and also to make the performance comparison with [10] fair), we define the *worst-case performance* of an agent controller over the period $[0,T]$ as J_W where

$$J_W = \max_{i\in\mathscr{V},\ t\in[0,T]} \operatorname{tr}(\Omega_i(t)). \tag{50}$$

In our case, J_W is simply the maximum recorded target error covariance value in the network over the period $[0,T]$. For the same experiments that gave the results shown in Tab. I, we have evaluated the corresponding J_W (50) value and the obtained results are summarized in Tab. II. According to these results, it is evident that the periodic agent control methods have an advantage compared to the fully distributed and on-line methods like RHC and BDC in terms of worst-case performance. Nevertheless, the fact that the RHC-P method has obtained the best average J_W value (and the second-best average J_T value as shown in Tab. I) implies that the proposed RHC method can successfully be adopted to address different problem settings (with different constraints, objectives, etc.).

B. Simulation Study 2: The effect of agent controls on local target state tracking control

In this simulation study, we explore a byproduct of achieving reasonable target state estimates: the ability to control the target states effectively. Here, we assume each target has its own tracking control task that needs to be achieved through a simple local state feedback control mechanism. Clearly, for this purpose, each target has to rely on its own state estimate - of which the accuracy deteriorates when the target is not visited by an agent regularly. We define a new metric J_C to represent the performance of the overall *target state tracking control* process and compare the obtained J_C values by different agent controllers under different PCs.

TABLE II: The worst-case performance comparison (i.e., J_W in (50)) under five agent control methods in four PCs.

The Worst-Case Target State Estimator Performance (J_W)		Agent Control Mechanism				
		Off-line	Off-line/On-line		On-line	
		MTSP	BDC-P	RHC-P	BDC	RHC
PC No.	1	**40.23**	288.34	50.19	96.33	71.57
	2	38.03	586.19	65.62	270.42	**32.29**
	3	47.54	427.53	**46.52**	386.66	306.84
	4	115.74	403.38	**49.26**	487.65	77.41
Average:		60.38	426.36	**52.90**	310.27	122.03

a) ***Target Control Mechanism:*** In this simulation study, we assume that each target $i \in \mathscr{V}$ has to control its state $\phi_i(t)$ such that a signal $y_i(t) = C_i\phi_i(t) + D_i$, tracks a given reference signal $r_i(t)$ (C_i, D_i are also given).

Let us define the tracking error as $e_i(t) = y_i(t) - r_i(t)$. In order to make $e_i(t)$ follow the asymptotically stable dynamics: $\dot{e}_i = -K_i e_i(t)$ (with $K_i > 0$), the target i needs to select its control input $\upsilon_i(t)$ in (1) as (also recall (13)) $\upsilon_i(t) = -\frac{1}{B_iC_i}\left(C_i(A_i+K_i)\phi_i(t) + K_iD_i - (\dot{r}_i(t) + K_ir_i(t))\right)$. However, since target i is unaware of its state ϕ_i, naturally, the target state tracking controller can use the state estimate $\hat{\phi}_i$ in the above state feedback control law as $\upsilon_i(t) = -\frac{1}{B_iC_i}\left(C_i(A_i+K_i)\hat{\phi}_i(t) + K_iD_i - (\dot{r}_i(t) + K_ir_i(t))\right)$. To measure the performance of this target state tracking control task, we propose to use the performance metric J_C where

$$J_C \triangleq \frac{1}{T}\int_0^T |e_i(t)|dt. \tag{51}$$

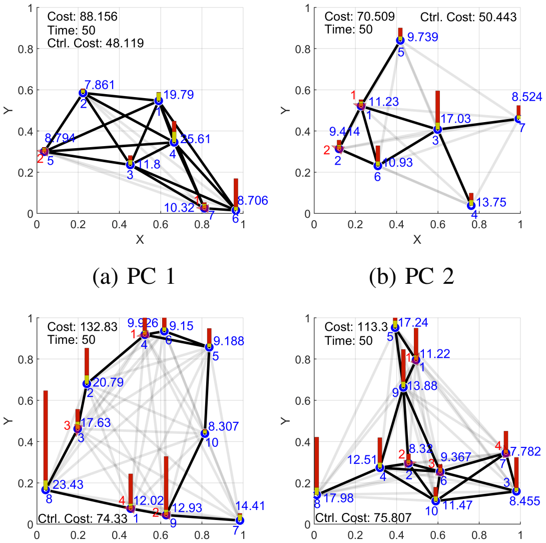
b) ***Results and Discussion:*** In this study, we set $C_i = 1$, $D_i = 0$, $K_i = 2$ and select the reference signal that needs to be tracked as: $r_i(t) = 10\sin(2t+i), \forall i \in \mathscr{V}, t \in [0,T]$ with $T = 50s$. The performance metric J_C observed for different PCs with different agent controllers are summarized in Tab III. Similar to before, the obtained results show that the RHC method, on average, has outperformed all the other agent controllers. Corresponding final states of the PCs observed under the RHC method are shown in Fig. 10. The red vertical bars (drawn on top of yellow vertical bars) represent the absolute tracking error $|e_i(t)|$ of each target i at $t = T$. These results imply that having an agent control mechanism that provides superior target state estimation capabilities (i.e., lower J_T) indirectly enables the targets to have better control over their states (i.e., lower J_C).

C. Simulation Study 3: Long term performance with learning

In both previous simulation studies, we focused on a relatively short period ($T = 50s$) that essentially encompassed the transient phase of the PMN system (which, in general, is the most challenging part to control/regulate). However, in this final simulation study, we aim to explore the performance of agent controllers over a lengthy period ($T = 750s$) that

TABLE III: Performance comparison of target state tracking (i.e., J_C in (51)) under five agent control methods in four PCs.

Target State Controller Performance ($J_C \times T$)		Agent Control Mechanism				
		Off-line	Off-line/on-line		On-line	
		MTSP	BDC-P	RHC-P	BDC	RHC
PC No.	1	57.97	51.99	50.99	48.40	**48.12**
	2	51.66	56.80	52.35	55.54	**50.14**
	3	**73.81**	81.40	74.08	80.89	74.33
	4	85.76	86.34	77.35	86.61	**75.81**
Average:		67.30	69.13	63.69	67.86	**62.10**



(a) PC 1

(b) PC 2

(c) PC 3

(d) PC 4

Fig. 10: Final state of the PCs after using the RHC method with target state tracking control.

includes both the transient and the steady-state phases of the PMN system. Note that this kind of a problem setup is ideal for deploying the machine learning influenced RHC solutions: RHC-L and RHC-AL proposed in Section VI. Therefore, in this study, we specifically compare the three controllers: RHC, RHC-L and RHC-AL for PCs 1 and 2, in terms of the evolution of: (i) the performance metric J_t (7) and (ii) the average processing time (commonly known as the "CPU time") taken to solve a RHCP, throughout the period $t \in [0,T]$. Note that these CPU times were recorded on an Intel Core i7-8700 CPU 3.20 GHz Processor with a 32 GB RAM.

As shown in Figs. 11(a) and 12(a), the RHC method takes the highest amount of CPU time to solve a RHCP. Its upward trend in the initial stages of the simulations indicates a transient phase of the processor (due to system cache utilization). We point out that this particular transient phase is independent of that of J_t curves shown in respective Figs. 11(b) and 12(b). Table IV shows that based on the steady-state averages for PC 1 (in Fig. 11), the RHC-L method spends 86.5% less CPU time compared to the RHC method but at a loss of 4.7% in performance. For the same PC, the RHC-AL method shows a 66.7% reduction in CPU time while having only a 0.1% loss in performance.

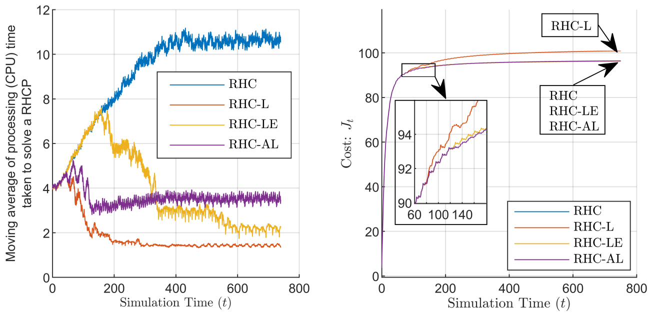
In these simulations of RHC-L and RHC-AL methods, for the on-line training of classifiers $f_i(X_i;\mathscr{D}_i)$ (required in (46)), we have selected the data set size $L=25$ (i.e., $|\mathscr{D}_i|=25$). As implied by Figs. 11(a) and 12(a), agents have been able to collect that amount of data points well within their transient phase (of the J_t curve). Even though learning based on transient data has a few advantages, it is mostly regarded as ineffective - especially if the learned controller would mostly operate in a steady-state condition. Therefore, we next extend the data set size to be $L=|\mathscr{D}_i|=75$ and execute the same RHC-L method, which henceforth is called the RHC-LE method. According to the summarized steady-state averaged data given in Tab IV, for PC 1 and 2, the RHC-LE method respectively shows 77.5% and 68.9% reductions in CPU time compared to the RHC method - while having almost no loss in performance ($< 0.001\%$) in both cases. Based on this observation, it is clear that the proposed RHC-LE method significantly improves computational efficiency with almost no loss in performance.

Remark 6: We attribute the observed superior performance of the proposed control solution to its specifically designed agent controllers (21)-(22). Apart from achieving such performance, the proposed agent controllers have many desirable practical qualities (e.g., computationally efficient, distributed, on-line, asynchronous and event-driven); moreover, they are the theoretically optimal controllers in their respective spatial and temporal neighborhoods (as explained using the established theoretical results in Theorems 1 and 2). However, while these practical qualities are coveted, they also make it difficult to establish global performance guarantees beyond the local performance guarantees [10], [12].

Remark 7: The reported experimental results in Tables I and III support the goal of this paper to develop a computationally efficient, distributed, on-line, and event-driven persistent monitoring solution. In particular, they show that the proposed RHC method performs better than other computationally expensive methods such as MTSP and RHC-P (both involve significant off-line computation components). Moreover, according to Table IV, the introduced learning-based computational efficiency improvement (e.g., RHC-LE method) does not compromise performance to achieve computational efficiency improvements. Therefore, it is clear that the improvement in computational efficiency (and achieving any other practical quality) has not come at the cost of a performance loss [14].

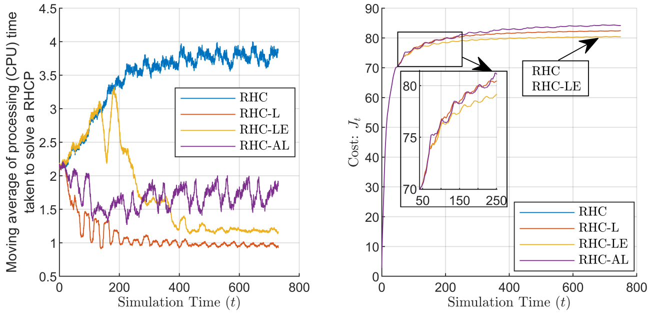
VIII. Conclusion

The goal of the estimation problem considered in this paper is to observe a distributed set of target states in a network using a mobile fleet of agents so as to minimize an overall measure of estimation error covariance evaluated over a finite period. Compared to existing centralized off-line agent control solutions, a novel computationally efficient distributed on-line solution is proposed based on event-driven receding horizon control. In particular, each agent determines their optimal planning horizon and the immediate sequence of optimal decisions at each event of interest faced in its trajectory. Numerical results show higher performance levels in multiple aspects than existing other centralized and distributed agent control methods. Future work aim to generalize the proposed solution for multidimensional target state dynamics.



(a) (b)

Fig. 11: Evolution of the average processing time taken to solve a RHCP and the objective function value for PC 1.



(a) (b)

Fig. 12: Evolution of the average processing time taken to solve a RHCP and the objective function value for PC 2.

TABLE IV: Average over the steady-state period $t \in [500, 750]$ of the curves in Figs. 11 and 12.

Average over steady-state: (Interval: [500, 750])		RHC	RHC-L ($L = 25$)	RHC-LE ($L = 75$)	RHC-AL ($L = 25$)
PC 1	CPU Time	10.589	1.429	2.379	3.526
	J_t	96.132	100.611	96.132	96.269
PC 2	CPU Time	3.791	0.967	1.181	1.733
	J_t	80.289	82.190	80.289	83.968

Appendix

A. Coefficients of the **RHCP1** objective function (38)

$$a_1 = \frac{1}{G_j}\log\left[\frac{G_j - 2A_j v_{j1}}{2A_j(v_{j2} - v_{j1})}\right] + \frac{1}{G_i}\log\left[\frac{-G_i\Omega_i - Q_i}{Q_i(v_{i2} - v_{i1})}\right],\ a_2 = \frac{1}{G_i},$$

$$a_3 = -\frac{G_i\Omega_i + Q_i v_{i2}}{G_i\Omega_i + Q_i v_{i1}},\ a_4 = \frac{1}{G_j},\ a_5 = -\frac{v_{j2}(2A_j\Omega_j + Q_j)e^{2A_j\rho_{ij}}}{Q_j v_{j2} + 2A_j},$$

$$a_6 = -\frac{G_j - 2A_j v_{j2}}{G_j - 2A_j v_{j1}}, a_7 = -a_5,\ a_8 = \frac{1}{v_{i1}},\ a_9 = \frac{1}{v_{i2}},$$

$$b_1 = -\frac{Q_i}{4A_i^2}(1 + 2A_i\rho_{ij}) - \frac{Q_j}{4A_j^2}(1 + 2A_j\rho_{ij}) - \frac{\Omega_j}{2A_j}$$

$$-\sum_{k\in\mathscr{N}_i\backslash\{j\}}\left[\frac{Q_k}{4A_k^2}(1 + 2A_k\rho_{ij}) + \frac{\Omega_k}{2A_k}\right],\ b_2 = -\sum_{k\in\mathscr{N}_i}\frac{Q_k}{2A_k},$$

$$b_3 = -\sum_{k\in\mathscr{N}_i\backslash\{j\}}\frac{Q_k}{2A_k},\ b_4 = \frac{1}{4A_j^2}(Q_j + 2A_j\Omega_j)e^{2A_j\rho_{ij}},$$

$$b_5 = \frac{Q_i}{4A_i^2}e^{2A_i\rho_{ij}},\ b_{6k} = \frac{1}{4A_k^2}(Q_k + 2A_k\Omega_k)e^{2A_k\rho_{ij}},$$

$$c_1 = -\frac{1}{2A_i v_{i1}},\ c_2 = -\frac{v_{i1}\Omega_i - 1}{v_{i2}\Omega_i - 1},\ c_3 = -e^{2A_i\rho_{ij}},$$

$$c_4 = -c_2,\ c_5 = -\frac{G_i\Omega_i + Q_i v_{i2}}{G_i\Omega_i + Q_i v_{i1}}.$$

References

[1] S. Welikala and C. G. Cassandras, “Event-Driven Receding Horizon Control for Distributed Estimation in Network Systems,” in *Proc. of American Control Conf.*, 2021, pp. 1559–1564.

[2] S. He, H. S. Shin, S. Xu, and A. Tsourdos, “Distributed Estimation Over a Low-Cost Sensor Network: A Review of State-Of-The-Art,” *Information Fusion*, vol. 54, pp. 21–43, 2020.

[3] M. Zhong and C. G. Cassandras, “Distributed Coverage Control and Data Collection with Mobile Sensor Networks,” *IEEE Trans. on Automatic Control*, vol. 56, no. 10, pp. 2445–2455, 2011.

[4] N. Zhou, C. G. Cassandras, X. Yu, and S. B. Andersson, “Optimal Threshold-Based Distributed Control Policies for Persistent Monitoring on Graphs,” in *Proc. of American Control Conf.*, 2019, pp. 2030–2035.

[5] J. Trevathan and R. Johnstone, “Smart Environmental Monitoring and Assessment Technologies (SEMAT)—A New Paradigm for Low-Cost, Remote Aquatic Environmental Monitoring,” *Sensors (Switzerland)*, vol. 18, no. 7, 2018.

[6] S. L. Smith, M. Schwager, and D. Rus, “Persistent Monitoring of Changing Environments Using a Robot with Limited Range Sensing,” in *Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2011, pp. 5448–5455.

[7] K. Leahy, D. Zhou, C. I. Vasile, K. Oikonomopoulos, M. Schwager, and C. Belta, “Persistent Surveillance for Unmanned Aerial Vehicles Subject to Charging and Temporal Logic Constraints,” *Autonomous Robots*, vol. 40, no. 8, pp. 1363–1378, 2016.

[8] N. Mathew, S. L. Smith, and S. L. Waslander, “Multirobot Rendezvous Planning for Recharging in Persistent Tasks,” *IEEE Trans. on Robotics*, vol. 31, no. 1, pp. 128–142, 2015.

[9] N. Rezazadeh and S. S. Kia, “A Sub-Modular Receding Horizon Approach to Persistent Monitoring for A Group of Mobile Agents Over an Urban Area,” in *IFAC-PapersOnLine*, vol. 52, no. 20, 2019, pp. 217–222.

[10] S. C. Pinto, S. B. Andersson, J. M. Hendrickx, and C. G. Cassandras, “Optimal Minimax Mobile Sensor Scheduling Over a Network,” in *Proc. of American Control Conf. (to appear)*, 2021.

[11] J. Yu, S. Karaman, and D. Rus, “Persistent Monitoring of Events With Stochastic Arrivals at Multiple Stations,” *IEEE Trans. on Robotics*, vol. 31, no. 3, pp. 521–535, 2015.

[12] S. Welikala and C. G. Cassandras, “Greedy Initialization for Distributed Persistent Monitoring in Network Systems,” *Automatica*, vol. 134, p. 109943, 2021.

[13] S. K. Hari, S. Rathinam, S. Darbha, K. Kalyanam, S. G. Manyam, and D. Casbeer, “The Generalized Persistent Monitoring Problem,” in *Proc. of American Control Conf.*, 2019, pp. 2783–2788.

[14] S. Welikala and C. G. Cassandras, “Event-Driven Receding Horizon Control for Distributed Persistent Monitoring in Network Systems,” *Automatica*, vol. 127, p. 109519, 2021.

[15] Y.-W. Wang, Y.-W. Wei, X.-K. Liu, N. Zhou, and C. G. Cassandras, “Optimal Persistent Monitoring Using Second-Order Agents with Physical Constraints,” *IEEE Trans. on Automatic Control*, vol. 64, no. 8, pp. 3239–3252, 2017.

[16] P. Maini, K. Yu, P. B. Sujit, and P. Tokekar, “Persistent Monitoring with Refueling on a Terrain Using a Team of Aerial and Ground Robots,” in *Proc. of IEEE Intl. Conf. on Intelligent Robots and Systems*, 2018, pp. 8493–8498.

[17] N. Zhou, X. Yu, S. B. Andersson, and C. G. Cassandras, “Optimal Event-Driven Multi-Agent Persistent Monitoring of a Finite Set of Data Sources,” *IEEE Trans. on Automatic Control*, vol. 63, no. 12, pp. 4204–4217, 2018.

[18] C. Song, L. Liu, G. Feng, and S. Xu, “Optimal Control for Multi-Agent Persistent Monitoring,” *Automatica*, vol. 50, no. 6, pp. 1663–1668, 2014.

[19] W. Li and C. G. Cassandras, “A Cooperative Receding Horizon Controller for Multi-Vehicle Uncertain Environments,” *IEEE Trans. on Automatic Control*, vol. 51, no. 2, pp. 242–257, 2006.

[20] S. S. Park, Y. Min, J. S. Ha, D. H. Cho, and H. L. Choi, “A Distributed ADMM Approach to Non-Myopic Path Planning for Multi-Target Tracking,” *IEEE Access*, vol. 7, pp. 163 589–163 603, 2019.

[21] X. Lan and M. Schwager, “Planning Periodic Persistent Monitoring Trajectories for Sensing Robots in Gaussian Random Fields,” in *In Proc. of IEEE Intl. Conf. on Robotics and Automation*, 2013, pp. 2415–2420.

[22] Y. Khazaeni and C. G. Cassandras, “Event-Driven Cooperative Receding Horizon Control for Multi-Agent Systems in Uncertain Environments,” *IEEE Trans. on Control of Network Systems*, vol. 5, no. 1, pp. 409–422, 2018.

[23] R. Chen and C. G. Cassandras, "Optimal Assignments in Mobility-on-Demand Systems Using Event-Driven Receding Horizon Control," *IEEE Trans. on Intelligent Transportation Systems*, pp. 1–15, 2020. [Online]. Available: https://doi.org/10.1109/TITS.2020.3030218

[24] A. Ma, K. Liu, Q. Zhang, T. Liu, and Y. Xia, "Event-Triggered Distributed MPC with Variable Prediction Horizon," *IEEE Trans. on Automatic Control*, vol. 66, no. 10, pp. 4873–4880, 2020.

[25] S. Welikala and C. G. Cassandras, "Event-Driven Receding Horizon Control of Energy-Aware Dynamic Agents for Distributed Persistent Monitoring," *arXiv e-prints*, p. 2102.12963, 2021. [Online]. Available: http://arxiv.org/abs/2102.12963

[26] S. C. Pinto, S. B. Andersson, J. M. Hendrickx, and C. G. Cassandras, "Multi-Agent Infinite Horizon Persistent Monitoring of Targets with Uncertain States in Multi-Dimensional Environments," in *Proc. of 21st IFAC World Congress*, 2020.

[27] M. Athans and E. Tse, "A Direct Derivation of the Optimal Linear Filter Using the Maximum Principle," *IEEE Trans. on Automatic Control*, vol. 12, no. 6, pp. 690–698, 1967.

[28] J. Nazarzadeh, M. Razzaghi, and K. Y. Nikravesh, "Solution of the Matrix Riccati Equation for the Linear Quadratic Control Problems," *Mathematical and Computer Modelling*, vol. 27, no. 7, pp. 51–55, 1998.

[29] B. Friedland, *Control System Design: An Introduction to State-Space Methods*. Dover Publications, 2012.

[30] L. Dieci and A. Papini, "Conditioning of the Exponential of a Block Triangular Matrix," *Numerical Algorithms*, vol. 28, no. 1-4, pp. 137–150, 2001.

[31] S. Welikala and C. G. Cassandras, "Event-Driven Receding Horizon Control for Distributed Estimation in Network Systems," *arXiv e-prints*, p. 2009.11958, 2020. [Online]. Available: https://arxiv.org/abs/2009.11958

[32] H. Lin and P. J. Antsaklis, *Hybrid Dynamical Systems: An Introduction to Control and Verification*. now, 2014. [Online]. Available: https://ieeexplore.ieee.org/document/8187294

[33] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 2016.

[34] G. P. Zhang, "Neural Networks for Classification: A Survey," *IEEE Trans. on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 30, no. 4, pp. 451–462, 2000.

[35] U. von Luxburg, "A Tutorial on Spectral Clustering," *arXiv e-prints*, p. 0711.0189, 2007. [Online]. Available: http://arxiv.org/abs/0711.0189

[36] T. Bektas, "The Multiple Traveling Salesman Problem: An Overview of Formulations and Solution Procedures," *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

Shirantha Welikala received the B.Sc. degree in electrical and electronic engineering from University of Peradeniya, Peradeniya, Sri Lanka, in 2015 and the M.Sc. and the Ph.D. degrees in systems engineering from Boston University, Brookline, MA, USA, in 2019 and 2021, respectively. From 2015 to 2017, he was with the Department of Electrical and Electronic Engineering, University of Peradeniya, Sri Lanka, where he worked first as a Temporary Instructor and subsequently as a Research Assistant. He is currently a Postdoctoral Research Fellow in the Department of Electrical Engineering, University of Notre Dame, South Bend, IN, USA. His main research interests include control and optimization of cooperative multi-agent systems (focusing coverage and monitoring applications), control of networked systems, passivity, symbolic control, machine-learning, robotics, and smart grid. s, including the 2015 Ceylon Electricity Board Gold Medal, the 2019 President's Award for Scientific Research in Sri Lanka and the 2021 Outstanding Ph.D. Dissertation Award in Systems Engineering.

Christos G. Cassandras is Distinguished Professor of Engineering at Boston University. He is Head of the Division of Systems Engineering, Professor of Electrical and Computer Engineering, and co-founder of Boston University's Center for Information and Systems Engineering (CISE). He received degrees from Yale University, Stanford University, and Harvard University. In 1982-84 he was with ITP Boston, Inc. where he worked on the design of automated manufacturing systems. In 1984-1996 he was a faculty member at the Department of Electrical and Computer Engineering, University of Massachusetts/Amherst. He specializes in the areas of discrete event and hybrid systems, cooperative control, stochastic optimization, and computer simulation, with applications to computer and sensor networks, manufacturing systems, and transportation systems. He has published about 450 refereed papers in these areas, and six books. He has guest-edited several technical journal issues and currently serves on several journal Editorial Boards, including Editor of Automatica. In addition to his academic activities, he has worked extensively with industrial organizations on various systems integration projects and the development of decision support software. He has most recently collaborated with The MathWorks, Inc. in the development of the discrete event and hybrid system simulator SimEvents.

Dr. Cassandras was Editor-in-Chief of the IEEE Transactions on Automatic Control from 1998 through 2009 and has also served as Editor for Technical Notes and Correspondence and Associate Editor. He was the 2012 President of the IEEE Control Systems Society (CSS). He has also served as Vice President for Publications and on the Board of Governors of the CSS, as well as on several IEEE committees, and has chaired several conferences. He has been a plenary/keynote speaker at numerous international conferences, including the 2017 IFAC World Congress, the American Control Conference in 2001 and the IEEE Conference on Decision and Control in 2002 and 2016, and has also been an IEEE Distinguished Lecturer.

He is the recipient of several awards, including the 2011 IEEE Control Systems Technology Award, the Distinguished Member Award of the IEEE Control Systems Society (2006), the 1999 Harold Chestnut Prize (IFAC Best Control Engineering Textbook) for Discrete Event Systems: Modeling and Performance Analysis, a 2011 prize and a 2014 prize for the IBM/IEEE Smarter Planet Challenge competition, the 2014 Engineering Distinguished Scholar Award at Boston University, several honorary professorships, a 1991 Lilly Fellowship and a 2012 Kern Fellowship. He is a member of Phi Beta Kappa and Tau Beta Pi. He is also a Fellow of the IEEE and a Fellow of the IFAC.