2024 American Control Conference (ACC)
July 8-12, 2024. Toronto, Canada

# Distributed Differentially Private Control Synthesis for Multi-Agent Systems with Metric Temporal Logic Specifications

Nasim Baharisangari, Narendhiran Saravanane, and Zhe Xu

*Abstract*— **We propose a distributed differentially private receding horizon control (RHC) approach for multi-agent systems (MAS) with metric temporal logic (MTL) specifications. In the MAS considered in this paper, each agent privatizes its sensitive information from other agents using a differential privacy mechanism. In other words, each agent adds privacy noise (e.g., Gaussian noise) to its output to maintain its privacy. We define two types of MTL specifications for the MAS: agent-level specifications and system-level specifications. Agents should collaborate to satisfy the system-level MTL specifications while each agent must satisfy its own agent-level MTL specifications at the same time. In the proposed distributed RHC approach, each agent communicates with its neighboring agents to acquire their estimate of the system-level trajectory and updates its estimate of the system-level trajectory. Then, each agent synthesizes its own control inputs such that the system-level specifications are satisfied with a probabilistic guarantee while the agent-level specifications are also satisfied with a deterministic guarantee. In the proposed optimization formulation of RHC, we directly incorporate Kalman filter equations to calculate the system-level trajectory estimates. We use mixed-integer linear programming (MILP) to encode the MTL specifications as optimization constraints. Finally, we implement the proposed distributed RHC approach in two scenarios.**

## I. Introduction

In multi-agent systems (MAS), agents often work together to accomplish various system-level tasks through communication with their neighboring agents [1]. Distributed control is typically used in MAS, as it offers benefits such as scalability and fast computing compared to centralized control [2], [3]. Centralized control can be computationally expensive and if the central control unit fails, the entire system may fail. In contrast, distributed control has better potential for fault tolerance [4].

In a MAS, agents may collaborate to achieve system-level objectives through communication, while simultaneously ensuring the protection of their sensitive information (e.g., actual position state) from neighboring agents [5]. To address this issue, *differential privacy* can be employed to safeguard agent privacy in a MAS. Differential privacy allows for system-level decision making while preventing an adversary from deducing an agent's sensitive information [6]. For dynamical systems, such as multi-agent systems, differential privacy preserves the privacy of each agent by adding *differential privacy noise* (e.g., Gaussian noise) to the trajectories that contain sensitive information, making it challenging for an adversary to deduce the privatized trajectories [5].

*Metric temporal logic* (MTL) is a powerful tool for defining complex tasks in MAS due to its expressiveness and interpretability [7]. MTL is a type of temporal logic that operates over real-valued data in the discrete-time domain [8]. Moreover, MTL is amenable to formal analysis, making it an excellent choice for defining challenging tasks, such as collision avoidance [9].

In this paper, we consider a MAS in which agents cooperate to fulfill system-level tasks while protecting their privacy and satisfying agent-level tasks. The tasks are defined using MTL formulas, and each agent employs differential privacy to privatize its trajectory that contains sensitive information. To satisfy the system-level tasks, each agent computes an estimate of the system-level trajectory. Specifically, each agent communicates with its neighboring agents to acquire their estimate of the system-level trajectory. Next, each agent applies receding horizon control (RHC) to synthesize control inputs for satisfying both the system-level tasks and agent-level tasks while using Kalman filter.

**Contributions:** We summarize our contributions as follows. (a) We propose a distributed differentially private receding horizon control (RHC) formulation for multi-agent systems (MAS) (unlike [5] which uses a centralized RHC), where each agent collaborates with its neighbors to accomplish a task with a probabilistic guarantee while preserving its privacy. (b) In the proposed approach, we incorporate Kalman filter equations directly into the RHC optimization formulation to account for uncertainties resulting from differential privacy. In our optimization formulation, we use a *one-step ahead prediction* of the noisy outputs to be used in the Kalman filter equations. (c) By assigning individual tasks in addition to system-level tasks, we exploit the capability of each agent to accomplish different tasks with a deterministic guarantee in our proposed distributed RHC. (d) We demonstrate the flexibility of our proposed framework in two different scenarios, where individual tasks can be adjusted without affecting the system-level tasks.

## II. Preliminaries

In this section, we explain the notations, definitions, and concepts that we use in this paper.

### A. System Dynamics and Features of Multi-Agent Systems

In this paper, a MAS consisting of $|\mathcal{Z}|$ agents moves in a bounded environment $\mathcal{S} \subseteq \mathbb{R}^{|\mathcal{D}|}$, where $\mathcal{Z}$ denotes the set

Nasim Baharisangari, Narendhiran Saravanane, and Zhe Xu are with the School for Engineering of Matter, Transport and Energy, Arizona State University, Tempe, AZ 85287. {nbaharis,nsarava4, xzhe1}@asu.edu (Corresponding author: Zhe Xu)

This work is partially supported by NSF CNS 2304863 and CNS 2339774 and ONR N00014-23-1-2505.

of the agents in the MAS, $|\mathcal{Z}|$ denotes the cardinality of $\mathcal{Z}$, and $\mathcal{D} = \{1, 2, ..., |\mathcal{D}|\}$ with $|\mathcal{D}|$ being the cardinality of the set $\mathcal{D}$. We represent the system dynamics of this MAS in the finite discrete time domain $\mathbb{T} = \{1, 2, ..., \tau\}$ (where $\tau \in \mathbb{N} = \{1, 2, ...\}$) with Eq. (1).

$$\boldsymbol{s}[t] = \boldsymbol{A}\boldsymbol{s}[t-1] + \boldsymbol{B}\boldsymbol{u}[t-1], \tag{1}$$

where $\boldsymbol{s}[t] = [(s_1[t])^T, (s_2[t])^T, ..., (s_{|\mathcal{Z}|}[t])^T]^T$ is the vector of the states of the agents in the MAS at time step $t$ and $s_i[t] = [s_{i,1}[t], s_{i,2}[t], ..., s_{i,|\mathcal{D}|}[t]]^T$ (where $i \in \mathcal{Z}$) denotes the state of agent $i$ at time step $t$; $\boldsymbol{u}[t-1] = [(u_1[t-1])^T, (u_2[t-1])^T, ..., (u_{|\mathcal{Z}|}[t-1])^T]^T$ is the vector of the control inputs at time step $t-1$ and $u_i[t-1] = [u_{i,1}[t-1], u_{i,2}[t-1], ..., u_{i,|\mathcal{D}|}[t-1]]^T$ is the control input vector of agent $i$ at time step $t-1$, and $\boldsymbol{A}$ and $\boldsymbol{B}$ are $|\mathcal{D}||\mathcal{Z}| \times |\mathcal{D}||\mathcal{Z}|$ diagonal time-invariant matrices. The dynamics equation of agent $i$ can be expressed as $s_i[t] = \boldsymbol{A}_{i*}s_i[t-1] + \boldsymbol{B}_{i*}u_i[t-1]$, where $s_i[t] \in \mathcal{S}$ and $u_i[t-1] \in \mathcal{U} = \{u | \|u\|_\infty \leq u_{\max}\}$ for all $i \in \mathcal{Z}$ and for all $t \in \mathbb{T}$. For agent $i$, $\boldsymbol{A}_{i*}$ and $\boldsymbol{B}_{i*}$ refer to diagonal $|\mathcal{D}| \times |\mathcal{D}|$ matrices with the same diagonal elements with the row and column indices $|\mathcal{D}|i - |\mathcal{D}| + 1, |\mathcal{D}|i - |\mathcal{D}| + 2, ..., |\mathcal{D}|i$ as in $\boldsymbol{A}$ and $\boldsymbol{B}$, respectively.

**Definition 1.** *We define the **system-level** trajectory $\eta$ as a function $\eta : \mathbb{T} \to \mathcal{S}$ to denote the evolution of the average of the states of all the agents in the MAS within a finite time horizon defined in the discrete time domain $\mathbb{T}$ and we define $\eta[t] := \frac{1}{|\mathcal{Z}|}\sum_{i=1}^{|\mathcal{Z}|} s_i[t]$. We also define the **agent-level** trajectory $s_i$ as a function $s_i : \mathbb{T} \to \mathcal{S}$ to denote the evolution of the state of each agent $i$ within a finite time horizon defined in the discrete time domain $\mathbb{T}$.*

In this paper, we represent the topology of the MAS with an *undirected graph* $G$ that is time-invariant.

**Definition 2.** *We denote an undirected graph by $G = (\mathcal{C}, \mathcal{E})$, where $\mathcal{C} = \{c_1, c_2, ..., c_{n_\mathcal{C}}\}$ is a finite set of nodes, $\mathcal{E} \subseteq \mathcal{E}' = \{e_{1,2}, e_{1,3}, ..., e_{1,n_\mathcal{E}}, e_{2,3}, ..., e_{n_\mathcal{E}-1,n_\mathcal{E}}\}$ is a finite set of edges where $e_{i,l} \in \mathcal{E}$ if nodes $c_i$ and $c_l$ are connected by an edge in the graph $G$, and $n_\mathcal{C}, n_\mathcal{E} \in \mathbb{N} = \{1, 2, ...\}$.*

Each node $c_i$ of the undirected graph $G$ represents an agent in the MAS. Each edge $e_{i,l}$ connecting the nodes $i$ and $l$ represents the fact that agents $i$ and $l$ are neighbors, i.e., agent $i$ and $l$ communicate with each other. Hereafter, we denote the set of the neighboring agents of agent $i$ with $\mathcal{Z}_i$. Also, we denote the adjacency matrix of the graph $G$ with $\boldsymbol{D}$.

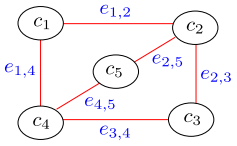


Fig. 1: An undirected graph $G = (\mathcal{C}, \mathcal{E})$ with $\mathcal{C} = \{c_1, c_2, ..., c_5\}$ and $\mathcal{E} = \{e_{1,2}, e_{1,4}, ..., e_{4,5}\}$.

Fig. 1 depicts an undirected graph $G = (\mathcal{C}, \mathcal{E})$ with $\mathcal{C} = \{c_1, c_2, ..., c_5\}$ and $\mathcal{E} = \{e_{1,2}, e_{1,4}, e_{2,3}, e_{2,5}, ..., e_{4,5}\}$. For example, $c_1$ corresponds to agent 1, $e_{1,2}$ represents that agents 1 and 2 are neighbors (i.e., agents 1 and 2 can communicate with each other), and $e_{1,4}$ represents that agents 1 and 4 are neighbors (i.e., agents 1 and 4 can communicate with each other).

### B. Differential Privacy

We use differential privacy to protect the sensitive information of each agent, e.g., it states [5], [10]. In the literature of differential privacy, $(\epsilon, \delta)$-differential privacy is a privacy framework used in data analysis to ensure that the inclusion or exclusion of any individual's data doesn't significantly affect the outcome of the analysis. Here, $\epsilon$ represents the privacy budget, controlling the level of privacy protection, while $\delta$ is an additional parameter used to provide a very small, but non-zero, probability that the privacy guarantee might be breached. A mechanism adhering to $(\epsilon, \delta)$-differential privacy adds noise to the data or modifies the results of computations in a controlled manner, balancing the need for accurate analysis with the protection of individual privacy. In differential privacy for MAS, each agent adds noise to its state and then shares its noisy output with its neighboring agents. In [10], it is proven that the Gaussian mechanism adds i.i.d Gaussian noise point-wise in time to the output of agent $i$ to keep its state private with the differential parameters $\epsilon > 0$ and $\delta \in (0, \frac{1}{2})$. This paper assumes that the Gaussian noise $v_i$ is time-invariant. Also, we denote the vector of the noisy outputs of all the $|\mathcal{Z}|$ agents at time step $t$ with $\tilde{\boldsymbol{y}}[t]$. In addition in this paper, we apply the differential privacy mechanism to the finite trajectory $s_i$ for each agent $i$. For further reading, refer to [5], [10].

### C. Metric Temporal Logic

In this subsection, we briefly review the metric temporal logic (MTL) [11]. We start with the Boolean semantics of MTL. The domain $\mathbb{B} = \{True, False\}$ is the Boolean domain. Moreover, we introduce a set $\Pi$ which is a set of *atomic predicates* each of which maps $\mathcal{S}$ to $\mathbb{B}$. Each of these predicates can hold values *True* or *False*. The syntax of MTL is defined recursively as follows.

$$\phi := \top \mid \pi \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \phi_1 \, \mathbf{U}_I \, \phi_2$$

where $\top$ stands for the Boolean constant *True*, $\pi$ is an atomic predicate such that $\pi \in \Pi$. $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction) are standard Boolean connectives, and "$\mathbf{U}$" is the temporal operator "until". We add syntactic sugar, and introduce the temporal operators "$\mathbf{F}$" and "$\mathbf{G}$" representing "eventually" and "always", respectively. $I$ is a time interval of the form $I = [a, b]$, where $a < b$, and they are non-negative integers. We define the set of the states that satisfy $\pi$ as $\mathcal{O}(\pi) \subset \mathcal{S}$. We denote the distance from $s$ to a set $\mathcal{J} \subseteq \mathcal{S}$ as $\mathbf{dist}_f(s, \mathcal{J}) := \inf\{f(s, s') | s' \in cl(\mathcal{J})\}$, where $f$ is a metric on $\mathcal{S}$, and $cl(\mathcal{J})$ denotes the closure of the set $\mathcal{J}$. In this paper, we use the metric $f(s, s') = \|s - s'\|_2$, where $\|.\|_2$ denotes the 2-norm. We denote the depth of $s$ in $\mathcal{J}$ by $\mathbf{depth}_f(s, \mathcal{J}) := \mathbf{dist}_f(s, \mathcal{S} \setminus \mathcal{J})$. We define the signed distance from $s$ to $\mathcal{J}$ as $\mathbf{Dist}_f(s, \mathcal{J}) := -\mathbf{dist}_f(s, \mathcal{J})$, if $s \notin \mathcal{J}$; and $\mathbf{Dist}_f(s, \mathcal{J}) := \mathbf{depth}_f(s, \mathcal{J})$ if $s \in \mathcal{J}$ [5].

**Definition 3.** *The **minimum necessary length** of an MTL*

*formula $\phi$, denoted by $H(\phi)$, is the minimum time steps required to evaluate the truth value of $\phi$. For the Boolean connectives and temporal operators, we have the following.*

$$\begin{aligned} H(\pi) &= 0, \\ H(\neg\phi) &= H(\phi), \\ H(\phi_1 \wedge \phi_2) &= \max\{H(\phi_1), H(\phi_2)\}, \\ H(\phi_1 \vee \phi_2) &= \min\{H(\phi_1), H(\phi_2)\}, \\ H(\phi_1 \mathbf{U}_{\mathcal{I}} \phi_2) &= H(\phi_1), \\ H(\mathbf{F}_{\mathcal{I}} \phi) &= H(\phi) + a, \\ H(\mathbf{G}_{\mathcal{I}} \phi) &= H(\phi) + b. \end{aligned}$$

**Definition 4.** *The Boolean semantics of an MTL formula $\phi$ with the necessary length of $H(\phi)$, for a trajectory $s$ at time step $t$ is defined recursively as follows.*

$$\begin{aligned} (s,t) \vDash \pi \ \textit{iff} \ & \ t \le H(\phi) \ \textit{and} \ s[t] \in \mathcal{O}(\pi) \\ (s,t) \vDash \neg\phi \ \textit{iff} \ & \ (s,t) \nvDash \phi, \\ (s,t) \vDash \phi_1 \wedge \phi_2 \ \textit{iff} \ & \ (s,t) \vDash \phi_1 \ \textit{and} \\ & \ (s,t) \vDash \phi_2, \\ (s,t) \vDash \phi_1 \mathbf{U}_{[a,b]} \phi \ \textit{iff} \ & \ \exists t' \in [t+a, t+b], \\ (s,t') \vDash \phi_2 \ \textit{and} \ & \ \forall t'' \in [t+a, t'), \ (s,t''] \vDash \phi_1. \end{aligned}$$

Robust semantics quantifies the degree at which a certain trajectory satisfies or violates an MTL formula $\phi$ at time step $t$. The robustness degree of an MTL formula $\phi$ with respect to a trajectory $s$ at time step $t$ is given by $r(s,\phi,t)$, where $r(s,\phi,t)$ can be calculated recursively via the robust semantics as follows.

$$r(s,\pi,t) = \mathbf{Dist}_f(s[t], \mathcal{O}(\pi)), \tag{2}$$

$$r(s,\neg\phi,t) = -r(s,\phi,t), \tag{3}$$

$$r(s,\phi_1 \wedge \phi_2, t) = \min(r(s,\phi_1,t), r(s,\phi_2,t)), \tag{4}$$

$$\begin{aligned} r(s,\phi_1 \mathbf{U}_{[a,b)} \phi_2, t) = & \max_{t+a \le t' < t+b} (\min(r(s,\phi_2,t'), \\ & \min_{t+a \le t'' < t'} r(s,\phi_1,t''))). \end{aligned} \tag{5}$$

### D. Estimation of the States Using Kalman Filter

In this subsection, we review the Kalman filter equations that are used to calculate the optimal estimates of the states using given noisy outputs. The Kalman filter equations are as follow [12].

$$\hat{\boldsymbol{s}}[t] = \boldsymbol{A}\hat{\boldsymbol{s}}[t-1] + \boldsymbol{B}\boldsymbol{u}[t-1] \tag{6}$$

$$+ \boldsymbol{K}[t](\tilde{\boldsymbol{y}}[t] - \boldsymbol{A}\hat{\boldsymbol{s}}[t-1] - \boldsymbol{B}\boldsymbol{u}[t-1]), \tag{7}$$

$$\boldsymbol{K}[t] = \boldsymbol{\Sigma}[t-1](\boldsymbol{\Sigma}[t-1] + \mathcal{K})^{-1}, \tag{8}$$

$$\boldsymbol{\Sigma}[t] = (I_{|\mathcal{Z}|} - \boldsymbol{K}[t])\boldsymbol{\Sigma}[t-1], \tag{9}$$

where $\hat{\boldsymbol{s}}[t] = [(\hat{s}_1[t])^T, (\hat{s}_2[t])^T, ..., (\hat{s}_{|\mathcal{Z}|}[t])^T]^T$ is the vector of the estimated states of $|\mathcal{Z}|$ agents at time step $t$ and $\hat{s}_i[t] = [\hat{s}_{i,1}[t], \hat{s}_{i,2}[t], ..., \hat{s}_{i,|\mathcal{D}|}[t]]^T$ (where $i \in \mathcal{Z}$) denotes the $|\mathcal{D}|$-dimensional estimated state of agent $i$ at time step $t$, $\boldsymbol{K}[t]$ is the Kalman gain matrix at time step $t$ and is a $|\mathcal{D}||\mathcal{Z}| \times |\mathcal{D}||\mathcal{Z}|$ matrix, $\boldsymbol{\Sigma}[t]$ is the covariance matrix of state estimation error at time step $t$ and is a $|\mathcal{D}||\mathcal{Z}| \times |\mathcal{D}||\mathcal{Z}|$ matrix, $\mathcal{K} = \mathbb{E}(\boldsymbol{v}\boldsymbol{v}^T)$ is the covariance matrix of the noise vector $\boldsymbol{v}$. Also, we assume that each agent $i$ knows that $\boldsymbol{v}$ conforms to a Gaussian distribution with 0 mean and covariance matrix $\mathcal{K}$. Also, each agent $i$ knows that $\mathbb{E}(\|\boldsymbol{v}\|^2) \le |\mathcal{Z}| v_{\max}$ and $\mathbb{E}(\|\hat{s}_i[0] - s_i[0]\|^2) \le s_{\max}$ with $v_{\max}$ and $s_{\max}$ being arbitrary values [13].

## III. Problem Formulation

In this section, we formalize the problem of synthesizing controller inputs for a MAS consisting of $|\mathcal{Z}|$ agents in a differentially private manner, where agents are required to collaborate to satisfy a system-level task specified using an MTL specification $\phi_s$ with a probabilistic guarantee, and at the same time, each agent $i$ should satisfy an agent-level MTL specification $\phi_i$. Intuitively, in order to collaborate in satisfying a system-level task $\phi_s$, each agent $i$ needs to have access to the system-level trajectory $\eta$. However, in the situation where each agent $i$ can only communicate with its neighboring agent while preserving its privacy regarding its state, each agent $i$ needs to have an estimate of $\eta$ while taking into consideration that the probability of the satisfaction of the MTL specification $\phi_s$ is higher than a given minimum value $\gamma_{\min}$.

In this MAS, each agent $i$ must keep its actual state $s_i[t]$ private from its neighboring agents while agent $i$ is aware of its own actual state $s_i[t]$. Here, the communication is asynchronous i.e., only two agents can communicate at each time step $t$ and the probability of each agent $i$ being *active* at time step $t$ is $\frac{1}{|\mathcal{Z}|}$ (here active means agent $i$ can initiate communication with another agent). In other words, if agent $i$ is not active at time step $t$, then it can not initiate communication with its neighboring agent.

We want to synthesize the controller inputs in a distributed manner, i.e., each agent $i$ synthesizes its own controller input $u_i[t]$ at time step $t$ while taking into consideration that the probability of the satisfaction of the MTL specification $\phi_s$ by the actual system-level trajectory $\eta[t]$ has a probabilistic guarantee.

Now, we formalize the problem of synthesizing control inputs for the control horizon of $2H$ for a MAS with $|\mathcal{Z}|$ agents in a distributed and differentially private manner.

**Problem 1.** *Given a MAS consisting of $|\mathcal{Z}|$ agents, the privacy parameters $\epsilon_i \in [\epsilon_{min}, \epsilon_{max}]$ (where $0 < \epsilon_{min} < \epsilon_{max}$) and $\delta_i \in [\delta_{min}, \delta_{max}]$ (where $0 < \delta_{min} < \delta_{max} < \frac{1}{2}$), and the objective function $J = \sum_{k=1}^{2H} \|\boldsymbol{u}_i[k]\|^2$, synthesize the controller inputs $\boldsymbol{u}_i[t]$ in the control horizon $2H$ in a distributed and differentially private manner such that the satisfaction of the system-level MTL specification $\phi_s$ with the probability higher than $\gamma_{min}$, i.e., $\mathbb{P}((\eta[0:H-1], 0) \vDash \phi_s) > \gamma_{min}$ and the satisfaction of the agent-level MTL specification $\phi_i$ by agent $i$ is guaranteed at time step $t$ while the objective function $J$ is minimized.*

## IV. Estimation of System-Level Trajectory in a MAS with MTL Specifications

In this section, we review a method to estimate the system-level trajectory $\eta$. In this MAS, each agent $i$ has asynchronous communication with only its neighboring agents. In

what follows, we review a method by which each agent can estimate the system-level trajectory $\eta$ in a distributed manner [13]. The main idea of this method is that each agent is able to compute an estimate of the actual system-level trajectory $\eta$ such that the estimation error converges to zero when the time $t$ goes to infinity. We explain this framework for one dimension and the same framework can be applied to other dimensions as well. Hence, we assume that $|\mathcal{D}| = 1$. We assume that at time step $t-1$, agents $i$ and $l$ have computed the estimated system-level trajectory $\zeta_i[t-1]$ and $\zeta_l[t-1]$. At time step $t$, agents $i$ and $l$ communicate with each other and update their estimates of the system-level trajectory $(\eta[t])_d$ using Eqs. (10) and (11), respectively. Let $\boldsymbol{W}$ be a $|\mathcal{Z}|\times|\mathcal{Z}|$ matrix associated with the MAS where each entry $\boldsymbol{W}_{il}$ represents the probability that agent $i$ communicates with agent $l$ according to the adjacency matrix $\boldsymbol{D}$.

$$\zeta_i[t] = \frac{1}{2}(\zeta_i[t-1] + \zeta_l[t-1]) + \hat{s}_i[t]) - \hat{s}_i[t-1], \quad (10)$$

$$\zeta_l[t] = \frac{1}{2}(\zeta_i[t-1]) + \zeta_l[t-1]) + \hat{s}_l[t] - \hat{s}_l[t-1], \quad (11)$$

and other agents update their estimates of the system-level trajectory using Eq. (12)

$$\zeta_k[t] = \zeta_k[t-1] + \hat{s}_k[t] - \hat{s}_k[t-1], \ k \in \mathcal{Z} \text{ and } k \neq i, l. \quad (12)$$

We can reformulate the equations of the update of the estimated system-level trajectory by the agents in the vector form using Eq. (13)

$$\boldsymbol{\zeta}[t] = \boldsymbol{V}[t-1]\boldsymbol{\zeta}[t-1] + \hat{\boldsymbol{s}}[t] - \hat{\boldsymbol{s}}[t-1], \quad (13)$$

where $\boldsymbol{\zeta}[t] = [(\zeta_1[t])^T, (\zeta_2[t])^T, ..., (\zeta_{|\mathcal{Z}|}[t])^T]^T$ is the vector containing the estimates of the actual system-level trajectory $\eta[t]$ made by the $|\mathcal{Z}|$ agents at time step $t$, and $\zeta_i[t] = [\zeta_{i,1}[t], ..., \zeta_{i,|\mathcal{D}|}[t]]^T$. As we mentioned earlier, the probability of agent $i$ to be active is $\frac{1}{|\mathcal{Z}|}$; thus matrix $\boldsymbol{V}[t]$ has a probability of $\frac{1}{|\mathcal{Z}|}\boldsymbol{W}_{il}$ to be equal to $(\tilde{\boldsymbol{V}})_{il} = I_{|\mathcal{Z}|} - \frac{(e_i - e_l)(e_i - e_l)^T}{2}$, where $I_{|\mathcal{Z}|}$ is a $|\mathcal{Z}|\times|\mathcal{Z}|$ identity matrix and $e_i = [0, ..., 1, ..., 0]^T$ is a $|\mathcal{Z}|\times 1$ vector with the $i$-th entry to be 1 and zero in all other entries [14] [13].

It can be shown that the expected value of the matrix $\boldsymbol{V}[t]$, denoted by $\mathbb{E}(\boldsymbol{V}[t]) = \boldsymbol{V}$, is constant at different time steps $t$ [14] and can be calculated using optimization (33) in [13] and the second largest eigenvalue of $\boldsymbol{V}$ denoted by $\lambda$ governs the convergence rate estimation of $\eta$.

In order to use $\boldsymbol{\zeta}[t]$ for synthesizing controller inputs for the MAS, we need to address two important matters: (1) it is crucial to guarantee that the estimation error of the vector of the estimated system-level trajectories $\boldsymbol{\zeta}[t]$ (in comparison with the actual system-level trajectory $\eta[t]$) converges to zero when $t \to \infty$, and (2) we need to guarantee that the actual system-level trajectory $\eta$ satisfies $\phi_s$ with a probability higher than a minimum value $\gamma_{\min}$ given that agents do not have access to the actual system-level trajectory. Therefore, we need to provide the guarantee of $\eta[t]$ satisfying $\phi_s$ using the vector of the estimated system-level trajectories $\boldsymbol{\zeta}[t]$. In what follows, we provide Theorem 1 and Lemma 1, which address the two mentioned issues, respectively [13].

**Theorem 1.** *The estimation error $\mathbb{E}(\|\boldsymbol{\zeta}[t] - \eta[t]\mathbf{1}\|_\infty)$ converges to zero when $t \to \infty$ for a MAS consisting of $|\mathcal{Z}|$ agents, where $\boldsymbol{\zeta}[t]$ is the vector of estimates of the actual system-level trajectory $\eta[t]$ at time step $t$ calculated by $|\mathcal{Z}|$ agents.*

We can provide an upper bound for the estimation error $\mathbb{E}(\|\boldsymbol{\zeta}[t] - \eta[t]\mathbf{1}\|_\infty)$ using Corollary 1. Here we assume that each agent knows $\mathbb{E}(\|\boldsymbol{\zeta}[0] - \overline{\zeta}[0]\mathbf{1}\|_\infty) \leq \zeta_{\max}$, where $\overline{\zeta}[0] = \frac{1}{|\mathcal{Z}|}\sum_{i=1}^{|\mathcal{Z}|}\zeta_i[0]$ and $\zeta_{\max}$ is an arbitrary value.

**Corollary 1.** *For the estimation error $\mathbb{E}(\|\boldsymbol{\zeta}[t] - \eta[t]\mathbf{1}\|_\infty)$ at time step $t$, we have the upper bound defined as*

$$\rho[t] = \lambda^t\sqrt{|\mathcal{Z}|}\zeta_{max} + \mathcal{L}_1\sum_{k=1}^{t}\lambda^{t-k}\sqrt{\delta_{max}(k) + \delta_{max}(k-1) + 2|\mathcal{Z}|(u_{max})^2} + \mathcal{L}_1\sqrt{\delta_{max}(t)}, \quad (14)$$

*where $\lambda$ is the second largest eigenvalue of matrix $\boldsymbol{V}$, $\mathcal{L}_1$ and $\mathcal{L}_2$ are two Lipschitz constants and $\delta_{max}(t) = \frac{|\mathcal{Z}|^2 s_{max} v_{max}}{v_{max} + t s_{max}}$.*

For further details regarding the derivation of the upper bound $\rho[t]$, we kindly refer the reader to Lemma 4 in [13].

We use the upper bound $\rho[t]$ to provide the guarantee that the probability of $\eta[t]$ satisfying the system-level MTL specification $\phi_s$, referred to as *confidence level*, is higher than a minimum value $\gamma_{\min}$; therefore, we provide a set of constraints that each agent $i$ must satisfy recursively [13].

**Lemma 1.** *Let $r_{\min} \geq 0$ denote the minimum required robustness degree of $\boldsymbol{\zeta}$ satisfying a given system-level MTL specification $\phi$ at time step $t$, the confidence level of agent $i$ of $\eta[t]$ satisfying $\phi$ at time step $t$, denoted by $\mathbb{P}_i((\eta, t) \vDash \phi)$, must satisfy the following constraints.*

$$\mathbb{P}_i((\eta,t) \vDash \pi) \geq \begin{cases} 1 - \frac{\rho[t]}{r_{\min}}, & \text{if } \rho[t] < r_{min} \\ 0, & \text{otherwise} \end{cases}$$

$$\mathbb{P}_i((\eta,t) \vDash \phi_1 \wedge \phi_2) \geq \mathbb{P}_i((\eta,t) \vDash \phi_1) + \mathbb{P}_i((\eta,t) \vDash \phi_2) - 1,$$

$$\mathbb{P}_i((\eta,t) \vDash \phi_1 \vee \phi_2) \geq 1 - \min\{1 - \mathbb{P}_i((\eta,t) \vDash \phi_1), 1 - \mathbb{P}_i((\eta,t) \vDash \phi_2)\},$$

$$\mathbb{P}_i((\eta,t) \vDash \phi_1 \, \mathbf{U}_{[a,b]} \, \phi_2) \geq 1 - \min_{t' \in [t+a, t+b]}\{1 - \mathbb{P}_i((\eta,t') \vDash \phi_2) + (\sum_{t''=t+a}^{t'} 1 - \mathbb{P}_i((\eta,t'') \vDash \phi_1)\}.$$

## V. Distributed Differentially Private Receding Horizon Control for Multi-Agent Systems with MTL Specifications

In this section, we introduce an approach for synthesizing control inputs for a MAS with MTL specifications in a distributed and differentially private manner. Based on the settings of Problem 1, at each time step $t$, each agent $i$ should synthesize its own control inputs in the time horizon $[t, t+H-1]$ for satisfying the system-level and agent-level specifications $\phi_s$ and $\phi_i$. For solving Problem 1, we adopt a receding horizon control (RHC) for synthesizing control inputs for satisfying MTL specifications $\phi_s$ and $\phi_i$ while

minimizing a given objective function $J$.

In RHC for satisfying MTL specifications, we incorporate mixed-integer linear programming (MILP) to encode given MTL specifications as constraints in the optimization problem that is solved at each time step $t$. In [15], Raman *et. al.* introduce a framework for encoding MTL specifications as MILP constraints, and we incorporate this framework in this paper. Implementing MTL formulas using the technique in [15] guarantees the satisfaction of a given MTL formula with a minimum robustness degree.

We want to synthesize the controller inputs in a distributed manner, i.e., each agent $i$ synthesizes its own controller input $u_i[t]$ at time step $t$. Hence, agent $i$ must calculate $\zeta_i[t]$ for computing the controller input $u_i[t]$ while taking into consideration that the probability of the satisfaction of the MTL specification $\phi_{\text{s}}$ by the actual system-level trajectory $\eta[t]$ is higher than a minimum value.

**Remark 1.** *In the MAS with $|\mathcal{Z}|$ agents, each agent $i$ (1) has access to the time-invariant graph-structure of the MAS given by the undirected graph $G$, (2) knows the fact that the asynchronous communication of agent $i$ is only with its neighboring agents $l \in \mathcal{Z}_i$ where $\mathcal{Z}_i$ is deduced from the undirected graph $G$, and (3) calculates the error upper bound* (14) *independently.*

The optimization formulation proposed for synthesizing control inputs in a differentially private manner at time step $t$ for a MAS referred to as Diff-MILP is presented in the following.

$$\arg \min_{u_i[t:t+H-1]} \sum_{k=t}^{t+H-1} \|u_i[k]\|^2 \tag{15}$$

$$\text{subject to: } s_i[k+1] = \boldsymbol{A}_{i*} s_i[k] + \boldsymbol{B}_{i*} u_i[k], \quad \forall k \in \{t, t+1, ..., t+H-1\}, \tag{16}$$

$$\hat{s}_i[k+1] = \boldsymbol{A}_{i*} \hat{s}_i[k] + \boldsymbol{B}_{i*} u_i[k] + \boldsymbol{K}_{i*}[k] \big(\tilde{y}_i[k+1] - \boldsymbol{A}_{i*} \hat{s}_i[k] + \boldsymbol{B}_{i*} u_i[k]\big), \quad \forall k \in \{t, t+1, ..., t+H-1\}, \tag{17}$$

$$\zeta_i[k+1] = \zeta_i[k] + \hat{s}_i[k+1] - \hat{s}_i[k], \quad \forall k \in \{t, t+1, ..., t+H-1\}, \tag{18}$$

$$\tilde{y}_i[k+1] = \big(\hat{s}_i[k] + u_i[k]\big) + \frac{\boldsymbol{C}_{i*} \boldsymbol{B}_{i*} \boldsymbol{A}_{i*}}{2} (u_i[k] - u_i[k-1]), \quad \forall k \in \{t, t+1, ..., t+H-1\}, \tag{19}$$

$$r_i(s_i[0:H-1], \phi_i, j) > P[j], \ \ \forall j \in \{0, 1, ..., H-1\}, \tag{20}$$

$$\mathbb{P}_i((\eta[0:H-1], j) \vDash \phi_{\text{s}}) > Co[j], \ \ \forall j \in \{0, 1, ..., H-1\}, \tag{21}$$

$$u_{\min} \le u_i[k] \le u_{\max} \ \ \forall k \in \{0, 1, ..., H-1\}. \tag{22}$$

A challenge in incorporating Eq. (6), in the optimization formulation in Diff-MILP, is to calculate the vector of the noisy output $\tilde{y}_i[t]$ in the time horizon $[t, t+H]$. To overcome this challenge, we exploit the technique of *one-step ahead prediction* of the vector of the noisy output introduced in [16]. Based on the idea introduced in [16], at time step $t$, we can calculate the one-step ahead prediction of vector of the noisy output $\tilde{y}[t+1]$ using Eq. (23).

$$\tilde{y}[t+1] = (\hat{s}[t] + u[t]) + \frac{\boldsymbol{C}_{i*} \boldsymbol{B}_{i*} \boldsymbol{A}_{i*}}{2} (u[t] - u[t-1]) \tag{23}$$

where $\boldsymbol{C}$ is the output matrix with the dimension $|\mathcal{D}||\mathcal{Z}| \times |\mathcal{D}||\mathcal{Z}|$. In what follows, we explain the details of the proposed approach. As was mentioned earlier, we want to synthesize the controller inputs in a distributed manner, i.e., each agent $i$ solves Diff-MILP in the control horizon of $H$, at each time step $t$, to synthesize its own control inputs.

Alg. 1 illustrates the proposed receding horizon control procedure that each agent $i$ uses to synthesize its own control inputs in the time length $\tau$. In Alg. 1, at each time step $t$, each agent $i$ computes (1) a finite agent-level trajectory $s_i$ with a length of $H$ that satisfies $\phi_i$ at time-step $t$ and (2) computes finite estimated system-level trajectories $\zeta_i$ with a length of $H$ that satisfies $\phi_{\text{s}}$ at time-step $t$ while taking into consideration that the actual system-level trajectory $\eta$ satisfies $\phi_{\text{s}}$ with a minimum probability $\gamma_{\min}$.

**Algorithm 1:** Distributed Differentially Private Receding Horizon Control for MTL Specifications

**Input:** A positive large number $M$
$\max_{i \in \mathcal{Z}} \{H(\phi_{\text{s}}), H(\phi_i)\}$ and the time length $\tau$
The minimum confidence level $\gamma_{\min}$
The minimum robustness degree $r_{\min}$
The adjacency matrix $\boldsymbol{D}$
The number of the agents $|\mathcal{Z}|$
The initial state covariance matrix $\boldsymbol{\Sigma}[0]$ and the noise covariance matrix $\mathcal{K}$
Lipschitz constants $\mathcal{L}_1$ and $\mathcal{L}_2$, $s_{\max}$, $\zeta_{\max}$, and $v_{\max}$

Agent $i$ calculates $\lambda(\boldsymbol{V})$
**while** $t < \tau$ *and Diff-MILP is feasible* **do**
  Agent $i$ calculates the Kalman gain matrices $\boldsymbol{K}_{i*}[k]$ for all $k \in \{t, t+1, ..., t+H-1\}$ using Eq. (8)
  **if** *agent $i$ is active* **then**
    Agent $i$ communicates with agent $l \in \mathcal{Z}_i$ to acquire $\zeta_l[t]$
    $\zeta_i[t] \leftarrow \frac{\zeta_i[t] + \zeta_l[t]}{2}$
  **end if**
  $r_{\min} \leftarrow P[j], \ \forall j \in \{0, ..., H-1\}$
  $\gamma_{\min} \leftarrow Co[j], \ \forall j \in \{0, ..., H-1\}$
  Compute $\mathbf{U}_i[t] = \big[\boldsymbol{u}_i^0[t], \boldsymbol{u}_i^1[t], ..., \boldsymbol{u}_i^{H-1}[t]\big]$ by solving Diff-MILP
**end while**

**Remark 2.** *Because each agent $i$ has access to its own actual trajectory $s_i$ at each time step $t$, we enforce the MILP constraints related to each $r_i(s_i[0:H-1], \phi_i, j) > r_{\min}$ directly using $s_i$ for each agent $i$, where $j \in [0, H-1]$.*

In Diff-MILP, we implement the constraint $r_i(s_i[0:H-1], \phi_i, j) > r_{\min}$ (Eq. (20)) using the MILP technique introduced in [15]. We use the positive large number $M$

in this MILP problem where we use big-$M$ formulation to encode the MTL formulas. $\mathbb{P}_i((\eta[0:H-1],j) \vDash \phi_s) > \gamma_{\min}$ (Eq. (21)) enforces that the actual system-level trajectory $\eta[j]$ satisfies $\phi_s$ with the minimum probability $\gamma_{\min}$ for all $j \in [0, H-1]$. For implementing the constraint $\mathbb{P}_i((\eta[0:H-1],j) \vDash \phi_s) > \gamma_{\min}$, we enforce $r_i(\zeta_i[0:H-1], \phi_s, j) > r_{\min}$ and we calculate the estimated error bound $\rho_i[j]$ using Eq. (14) for all $j \in [0, H-1]$. Then, we encode the constraint $\mathbb{P}_i((\eta,t) \vDash \pi) \geq 1 - \frac{\rho[j]}{r_{\min}}$, and after that we recursively encode the constraints listed in Lemma 1 according to $\phi_s$ for all $j \in [0, H-1]$.

In Alg. 1, each agent $i$ sets $\zeta_i[0]$ equal to $\hat{s}_i[0]$. At $t = 0$, each agent $i$ calculates $\lambda(\boldsymbol{V})$ by solving optimization problem (33) in [13] (Line 1 in Alg. 1) to calculate the error upper bound $\rho[j]$ for all $j \in [0, H-1]$ at each time step $t$. At each time step $t$, each agent $i$ calculates the Kalman gain matrices $\boldsymbol{K}_{i*}[t]$ in the time horizon $[t, t+H-1]$ using Eq. (8) and (9) (Line 3 in Alg. 1). If at time step $t$, agent $i$ is active, then agent $i$ communicates with one of its neighboring agents $l \in \mathcal{Z}_i$ with a uniform random probability $\frac{1}{|\mathcal{Z}_i|}$ to acquire $\zeta_l[t]$ and update $\zeta_i[t]$ (Lines 4-7 in Alg. 1). At Line 8 in Alg. 1, $P$ is a $1 \times H$ vector of variables that represent the minimum required robustness degree at each time step in the time horizon $[0, H-1]$. Here, we choose a robustness degree of value $r_{\min}$ for all $j \in [0, H-1]$ (Line 8 in ALg. 1). At Line 9 in Alg. 1, $\mathcal{C}o$ is a $1 \times H$ vector of variables that represents the minimum required confidence level of each agent $i$ in satisfying $\phi_s$ at each time step in the time horizon $[0, H-1]$. Here, we choose a confidence level of value $\gamma_{\min}$ for all $j \in [0, H-1]$. At Line 10 of Alg. 1, at each time step $t$, each agent $i$ calculates a sequence of control inputs $\mathbf{U}_i[t] = [\boldsymbol{u}_i^0[t], \boldsymbol{u}_i^1[t], ..., \boldsymbol{u}_i^{H-1}[t]]$ in the control horizon $H$ by solving Diff-MILP. Here, $\boldsymbol{u}_i^{H-1}[t]$ represents the predicted vector of control inputs calculated at the future time step $t+H-1$ by agent $i$, and the current time step is $t$.

## VI. Case Study

In this section, we implement the proposed approach for a MAS consisting of 8 agents in two different scenarios. In the MAS, the set of nodes is $\mathcal{C} = \{c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8\}$ and the set of edges is $\mathcal{E} = \{e_{1,2}, e_{1,5}, e_{2,3}, e_{2,6}, e_{3,4}, e_{3,7}, e_{4,8}\}$. We consider a 2-dimensional planar environment $\mathcal{S} \subseteq \mathbb{R}^2$ in which we have the following areas: (1) $R_1$ and $R_2$ are square areas both centered at $(0,0)$ with the edge length equal to 16 and 10, respectively. 2) $R_3$, $R_4$, $R_5$, and $R_6$ are rectangular areas all with the length and width of 6 and 2 which are centered at $(-2.5, 5)$, $(-2.5, -1)$, $(2.5, -5)$, and $(2.5, 1)$, respectively. We also denote the four quadrants of the 2D plane (starting from the positive quadrant going clockwise) by $\tilde{Q}_1$, $\tilde{Q}_2$, $\tilde{Q}_3$, and $\tilde{Q}_4$, respectively. We define the two scenarios as follows. In both scenarios, we specify the system-level specification as $\phi_s := \mathbf{G}_{[0,100]}(\eta \in R_2)$ which reads as "*the centroid of the MAS should always be in the area $R_2$ in the next 100 time steps*".

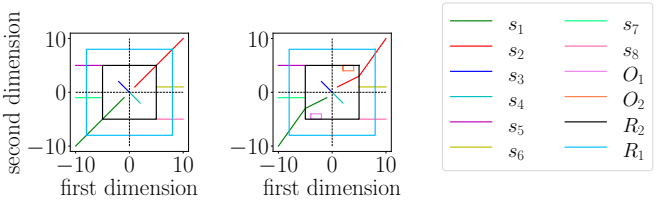


Fig. 2: The trajectories of the eight agents in Scenario I (Left) and Scenario II (Right).

**Scenario I:** In this scenario, we define the agent-level specifications as $\phi_{1,\mathrm{I}} := \big(\mathbf{F}_{[0,80]}(s_1 \in R_1)\big) \wedge \big(\mathbf{G}_{[0,100]}(s_1 \in \tilde{Q}_3)\big) \wedge \big(\mathbf{G}_{[0,100]} \neg(s_1 \in R_4)\big)$, $\phi_{2,\mathrm{I}} := \big(\mathbf{F}_{[0,80]}(s_2 \in R_1)\big) \wedge \big(\mathbf{G}_{[0,100]}(s_2 \in \tilde{Q}_1)\big) \wedge \big(\mathbf{G}_{[0,100]} \neg(s_2 \in R_6)\big)$, $\phi_{3,\mathrm{I}} := \big(\mathbf{F}_{[0,80]}(s_3 \in R_1)\big) \wedge \big(\mathbf{G}_{[0,100]}(s_3 \in \tilde{Q}_4)\big) \wedge \big(\mathbf{G}_{[0,100]} \neg(s_3 \in R_3)\big)$, $\phi_{4,\mathrm{I}} := \big(\mathbf{F}_{[0,80]}(s_4 \in R_1)\big) \wedge \big(\mathbf{G}_{[0,100]}(s_4 \in \tilde{Q}_2)\big) \wedge \big(\mathbf{G}_{[0,100]} \neg(s_4 \in R_5)\big)$, $\phi_{5,\mathrm{I}} := \big(\mathbf{G}_{[0,100]}(s_5 \in R_3)\big)$, $\phi_{6,\mathrm{I}} := \big(\mathbf{G}_{[0,100]}(s_6 \in R_6)\big)$, $\phi_{7,\mathrm{I}} := \big(\mathbf{G}_{[0,100]}(s_7 \in R_4)\big)$, and $\phi_{8,\mathrm{I}} := \big(\mathbf{G}_{[0,100]}(s_8 \in R_5)\big)$.

$\phi_{1,\mathrm{I}}$ reads as "*agent 1 should eventually reach area $R_1$ in the period of $[0,80]$, always stay in the third quadrant in the period of $[0,100]$, and never enter the area $R_4$ in the time span of $[0,100]$*". Similarly, the other agent-level specifications can be also translated into natural language.

**Scenario II:** This scenario is similar to Scenario I except that agents 1 and 2 must avoid collision with the obstacles $O_1$ and $O_2$ for safety purposes. Obstacles $O_1$ and $O_2$ are rectangular areas with a length of 3 and width of 1 centered at $(2.5, 4.5)$ and $(-2.5, -4.5)$, respectively. Here, we have $\phi_{1,\mathrm{II}} := \phi_{1,\mathrm{I}} \wedge \big(\mathbf{G}_{[0,100]} \neg(s_1 \in O_1)\big)$ and $\phi_{2,\mathrm{II}} := \phi_{2,\mathrm{I}} \wedge \big(\mathbf{G}_{[0,100]} \neg(s_2 \in O_2)\big)$.

For choosing the control horizon, we calculate the minimum necessary lengths of the given STL specifications and choose the largest one which is $H(\phi_1) = 100$ and we choose $\tau = 600$s. Also, we set $M = 1000$, $\gamma_{\min} = 0.9$, and $r_{\min} = 0.1$. In addition, we add the Gaussian noise to the outputs $y_i$ with the differential privacy parameters $\epsilon \in [\log(6), \log(10)]$ and $\delta \in [0.1, 0.4]$.

The left plot in Fig. 2 represents the trajectories of the eight agents in Scenario I. Fig. 3a represents the obtained results for the estimated system-level trajectories in comparison with $\eta$ (Left) and the actual agent-level trajectories of the eight agents (Right) in the first dimension of $\mathcal{S}$ in Scenario I. Both the estimated system-level trajectories and the actual agent-level trajectories in the second dimension follow the same trend as the first dimension for agents 1, 2, 3, and 4. For agents 5, 6, 7, and 8, the estimated system-level trajectories follow the same trend as the first dimension, and the actual agent-level trajectories are horizontal lines with the values within the coordinates of the designated regions $R_3$, $R_4$, $R_5$, and $R_6$, respectively. As can be seen in the top plot in Fig. 3a, $\phi_s$ is satisfied with the probability of 1 by the actual system-level trajectory $\eta$ which is higher than $\gamma_{\min} = 0.9$. According to top plot in Fig. 3a, $(\eta[t])_1 \in R_2$ for all $t \in [0, 600]$. Fig. 3a also shows that when the time goes to infinity, the estimated system-level trajectories converge to the actual system-level

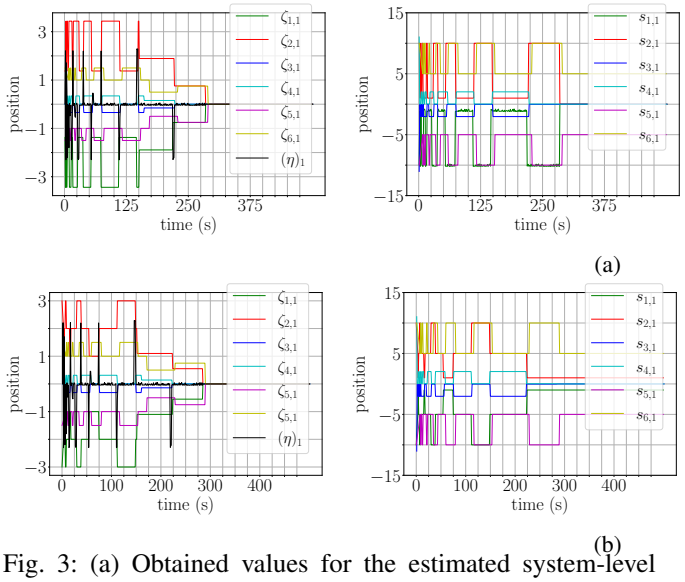


Fig. 3: (a) Obtained values for the estimated system-level trajectory $\zeta_i$ (Left) and the actual agent level trajectories $s_i$ (Right) with $i \in \{1, ..., 6\}$ for the first dimension of $\mathcal{S}$ in Scenario I. The results for agents 7 and 8 are not shown due to overlapping with the results of agents 5 and 6. (b) Obtained values for the estimated system-level trajectory $\zeta_i$ (Left) and the actual agent level trajectories $s_i$ (Right) with $i \in \{1, ..., 6\}$ for the first dimension of $\mathcal{S}$ in Scenario II. The results for agents 7 and 8 are not shown due to overlapping with the results of agents 5 and 6.

trajectory. The bottom plot in Fig. 3a shows that the agent-level specifications are also satisfied. For example, $s_2[t] \in \tilde{Q}_1$ for all $t \in [0, 600]$, and the time window of $[219, 284]$ is the longest time window that $s_2[t] \notin R_1$ which means that $\phi_{2,\mathrm{I}}$ is satisfied at all $t \in [0, 600]$. Similarly, $s_1[t] \in \tilde{Q}_3$ for all $t \in [0, 600]$, and the time window of $[222, 284]$ is the longest time window that $s_1[t] \notin R_1$ which means that $\phi_{1,\mathrm{I}}$ is satisfied at all $t \in [0, 600]$. The results for agents 7 and 8 are not shown in Fig. 3a since the obtained results for agents 7 and 8 are exactly the same as the obtained results for agents 5 and 6.

The right plot in Fig. 2 shows the paths of the eight agents in Scenario II. Fig. 3b illustrates the obtained results for the estimated system-level trajectories in comparison with $\eta$ (Left) and the actual agent-level trajectories of the eight agents (Right) in the first dimension of $\mathcal{S}$ in Scenario II. The top and the bottom plots in Fig. 3b show that $\phi_{\mathrm{s}}$ is satisfied by $\eta$ with the probability of 1 and all of the agent-level specifications are satisfied by the agents, respectively.

A comparison between the left and the right plot in Fig. 2 shows that changing the agent-level specifications of agents 1 and 2 do not interfere with satisfying $\phi_{\mathrm{s}}$. This shows that the proposed framework provides flexibility for the agents to satisfy different agent-level specifications without affecting the system-level specification. Moreover, the comparison between Scenario I and II shows that the proposed framework can be used for safety purposes such as avoiding collision between the agents and avoiding obstacles while collaborating to satisfy a task.

## VII. Conclusion

We introduced a distributed receding horizon control for multi-agent systems with metric temporal logic specifications. Future research directions include investigating control synthesis when agents share partial outputs instead of noisy ones, and integrating learning-based techniques [17] to handle unknown system dynamics for improved adaptability and robustness.

## References


[1] M. Kegeleirs, G. Grisetti, and M. Birattari, "Swarm slam: Challenges and perspectives," *Frontiers in Robotics and AI*, vol. 8, 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2021.618268

[2] G. Huang, Z. Zhang, and W. Yan, "Distributed control of discrete-time linear multi-agent systems with optimal energy performance," *Frontiers in Control Engineering*, vol. 2, 2022. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fcteg.2021.797362

[3] J. Chen, R. Sun, and H. Kress-Gazit, "Distributed control of robotic swarms from reactive high-level specifications," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, 2021, pp. 1247–1254.

[4] R. Patton, C. Kambhampati, A. Casavola, P. Zhang, S. Ding, and D. Sauter, "A generic strategy for fault-tolerance in control systems distributed over a network," *European Journal of Control*, vol. 13, no. 2, pp. 280–296, 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0947358007708245

[5] Z. Xu, K. Yazdani, M. T. Hale, and U. Topcu, "Differentially private controller synthesis with metric temporal logic specifications," 2019. [Online]. Available: https://arxiv.org/abs/1909.13294

[6] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," *CoRR*, vol. abs/1407.1338, 2014. [Online]. Available: http://arxiv.org/abs/1407.1338

[7] S. Seshia and D. Sadigh, "Towards verified artificial intelligence," *ArXiv*, vol. abs/1606.08514, 2016.

[8] E. Asarin, A. Donzé, O. Maler, and D. Nickovic, "Parametric identification of temporal properties," in *Runtime Verification*, S. Khurshid and K. Sen, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 147–160.

[9] E. Plaku and S. Karaman, "Motion planning with temporal-logic specifications: Progress and challenges," *AI Commun.*, vol. 29, pp. 151–162, 2016.

[10] J. Le Ny and G. J. Pappas, "Differentially private filtering," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 341–354, 2014.

[11] N. Baharisangari, J.-R. Gaglione, D. Neider, U. Topcu, and Z. Xu, "Uncertainty-aware signal temporal logic inference," in *Software Verification: 13th International Conference, VSTTE 2021, New Haven, CT, USA, October 18–19, 2021, and 14th International Workshop, NSV 2021, Los Angeles, CA, USA, July 18–19, 2021, Revised Selected Papers*. Berlin, Heidelberg: Springer-Verlag, 2021, p. 61–85. [Online]. Available: https://doi.org/10.1007/978-3-030-95561-8_5

[12] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 03 1960. [Online]. Available: https://doi.org/10.1115/1.3662552

[13] R. Yan and A. Julius, "Distributed consensus-based online monitoring of robot swarms with temporal logic specifications," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9413–9420, 2022.

[14] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[15] V. Raman, A. Donzé, M. Maasoumy, R. M. Murray, A. Sangiovanni-Vincentelli, and S. A. Seshia, "Model predictive control with signal temporal logic specifications," in *53rd IEEE Conference on Decision and Control*, 2014, pp. 81–87.

[16] J. H. Lee and N. L. Ricker, "Extended kalman filter based nonlinear model predictive control," in *1993 American Control Conference*, 1993, pp. 1895–1899.

[17] C. K. Verginis, Z. Xu, and U. Topcu, "Non-parametric neuro-adaptive coordination of multi-agent systems," in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '22. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2022, p. 1747–1749.