Distributed Consensus-Based Online Monitoring of Robot Swarms With Temporal Logic Specifications

Ruixuan Yan, *Graduate Student Member, IEEE*, and Agung Julius, *Senior Member, IEEE*

***Abstract*—In this letter, we develop a distributed consensus-based online monitoring framework for a robot swarm with a fixed graph structure. Each agent can monitor whether the swarm satisfies specifications given in the form of Swarm Signal Temporal Logic (SwarmSTL) formulas. SwarmSTL formulas describe temporal properties of swarm-level features represented by generalized moments (GMs), e.g., centroid and variance. To deal with measurement noise, we propose a generalized moment consensus algorithm (GMCA) with Kalman filter (KF), allowing each agent to estimate the GMs. Besides, we prove the convergence properties of the GMCA and derive an upper bound for the error between an agent's estimate of the GMs and the actual GMs. This upper bound is derived to be dependent on the maximal allowed velocity but independent of the agents' exact motion. A set of distributed monitoring rules for SwarmSTL formulas are proposed based on the estimation error bound. As a result, the agents can monitor the satisfaction of SwarmSTL formulas over swarm features during execution. The distributed monitoring framework is applied to a supply transportation example, where the efficacy of KF in the GMCA is also shown.**

***Index Terms*—Agent-based systems, planning, scheduling and coordination, sensor networks.**

I. Introduction

A ROBOT swarm is a multi-agent system composed of a large number of robots that can accomplish complicated tasks through cooperation and coordination [1]. With the increase of complexity and functionality, ensuring the safety and correctness of robotic systems is a challenging task. Nowadays, real-time temporal logic formulas such as Metric Temporal Logic (MTL) and Signal Temporal Logic (STL) are widely used to express such safety and correctness properties [2], [3]. The *formal controller synthesis* part of the safety/correctness issue aims to construct control laws that result in executions satisfying temporal logic formulas [4]–[8]. The *formal verification* part of the safety/correctness issue aims to check whether all the possible executions satisfy specified temporal logic formulas [9], [10]. For instance, the formal verification of a robot swarm can be accomplished via formal verification of individual agents' executions [10]. Recently, *temporal logic monitoring* approaches, which use a finite number of observation traces, have been proposed to assess whether the system fulfills formal requirements [11]–[13]. Offline monitoring and online monitoring are two temporal logic monitoring techniques. Offline monitoring is performed when complete execution traces have been gathered. Studies have revealed the relationship between the efficiency of offline monitoring and the length of the execution traces and the size of the formula ϕ, from both theoretical and experimental perspectives [14], [15]. However, there are some situations where the monitoring task needs to be performed during the execution. For instance, consider the specification that "*If the temperature is higher than a threshold within the last minute, the robot swarm must proceed to the fire source, spread over the fire zone, and extinguish the fire within 10 minutes.*" for a robot swarm performing a fire monitoring task in a warehouse. In many scenarios, the robot swarm needs to adapt the plan based on its perception of the environment or itself. Online monitoring is the appropriate approach for such applications.

Naturally, when we describe a swarm, we usually use abstract features of the swarm, such as the centroid or the variance of the swarm, whereas the behaviors of the individual agents are less important. The objective of this letter is to design a distributed online monitoring algorithm for abstract features, which allows individual agents to monitor the satisfaction of abstract features with respect to swarm-level specifications. Distributed monitoring of abstract features can lead to a significant improvement in the computational efficiency and robustness compared with centralized monitoring. Moreover, distributed monitoring can assist distributed control of robot swarms with temporal logic specifications, especially for reactive missions. For example, if a robot swarm performs a task of supplies transportation, and it needs to drop the supplies only if its centroid reaches a target region within 3 s. With distributed monitoring, the agents can determine whether to drop the supplies by monitoring if the centroid of the swarm reaches the target region within 3 s.

Extensive studies have developed centralized or decentralized control algorithms for multi-agent systems subject to temporal logic specifications [7], [16], [17], with the goal of synthesizing controllers that can realize executions satisfying temporal logic formulas. By contrast, the goal of this letter is to monitor if the swarm executions satisfy a high-level specification in a distributed manner. In [18], the authors proposed *generalized moments (GMs)* to represent swarm features and *SwarmSTL* to describe swarm-level behaviors. Following this idea, we develop a distributed monitoring algorithm for robot swarms with a fixed graph structure, with which the agents can monitor whether the swarm features satisfy SwarmSTL specifications. We propose a generalized moment consensus algorithm (GMCA) with Kalman filter (KF) so that the agents can estimate the GMs. Distributed consensus problems have been addressed by many distributed average consensus algorithms (DACAs) [19]–[21],

Manuscript received 1 March 2022; accepted 20 June 2022. Date of publication 15 July 2022; date of current version 26 July 2022. This letter was recommended for publication by Associate Editor P. Ogren and Editor M. Vincze upon evaluation of the reviewers' comments. This work was supported by NSF under Grants CNS-1618369 and CMMI-1936578. *(Corresponding author: Ruixuan Yan.)*

The authors are with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: yanr5@rpi.edu; juliua2@rpi.edu).

Digital Object Identifier 10.1109/LRA.2022.3191236

TABLE I
COMPARISON OF PROBLEM SETTING WITH RELATED WORKS

Features	DACA [19]–[21]	DKCF [22], [23]	GMCA (**this paper**)
Estimated object	Swarm averaged state	Entire swarm state	Swarm averaged state
Addresses noisy measurements	No	Yes	Yes
Assumes coupled measurements	No	Yes	No

which require the agents' estimates (ζ^j) to track the average of the signals from individual agents (θ^j). Most of the existing work assume a special initialization $\zeta^j(0) = \theta^j(0)$ or the agents' measurements are noiseless [19]–[21]. Without these assumptions, the DACA will have a steady-state error. We get rid of these assumptions by incorporating KF into the GMCA so that each agent can estimate its own state and use its state estimate to perform the GMCA. We show the convergence properties of the GMCA on static swarms, which are then generalized to obtain an estimation error bound between agents' estimates and the actual GMs. Moreover, GMCA with KF is demonstrated to be performed simultaneously with other motion planning algorithms.

Distributed Kalman consensus filter (DKCF), which incorporates consensus procedures into the design of distributed KF, has been developed for sensor fusion and tracking in multi-agent systems [22], [23]. A major feature of the DKCF is that the agents' measurements are coupled. By contrast, in this letter, each agent only has a noisy measurement of its own state and uses the KF to estimate its own state, i.e., the measurements are decoupled. For clarity, we present the comparison of problem settings in Table I. The contributions of this letter are summarized as follows:

- We propose a novel distributed GMCA with KF for robot swarms with a fixed graph structure, where the KF is used to track the agents' own states;
- We derive an upper bound of the estimation error between the agents' estimates and the actual GMs and show the convergence properties of GMCA on static swarms;
- We propose distributed monitoring rules for SwarmSTL formulas based on the GMCA so that the agents can monitor whether the swarm satisfies SwarmSTL formulas.

II. PRELIMINARIES

A. Dynamic Model and Features of a Robot Swarm

The robot swarm works in a planar environment $S \subseteq \mathbb{R}^2$. The discrete-time kinematic model of an agent is defined as

$$s(k+1) = s(k) + u(k), \tag{1}$$

where $s \in S$ is an agent's state, $k \in T$ is the time slot, $T = \mathbb{Z}_{\geq 0}$, $u \in U$ is the control input that directly controls the velocity, and $U = \{u | \|u\|_\infty \leq u_{\max}\}$. Equivalently, we can write s as $s = [s_x, s_y]^T$ and u as $u = [u_x, u_y]^T$. Let N denote the size of the swarm, and $\boldsymbol{s} \in \boldsymbol{S} = S^N$ denote the swarm state, i.e. $\boldsymbol{s} = [(s^1)^T, \ldots, (s^N)^T]^T$, where $s^j = [s_x^j, s_y^j]^T$ is Agent j's state. The dynamic model of a swarm becomes

$$\boldsymbol{s}(k+1) = \boldsymbol{s}(k) + \boldsymbol{u}(k), \tag{2}$$

where $\boldsymbol{u} \in \boldsymbol{U} = U^N$, $\boldsymbol{u} = [(u^1)^T, (u^2)^T, \ldots, (u^N)^T]^T$, and $u^j = [u_x^j, u_y^j]^T$ is the control input of Agent j.

A swarm is commonly described by a collection of abstract features such as the swarm's centroid or variance, whereas the agent-level features are less significant. Agents' states can be considered as samples from a certain distribution, which can be recovered using an infinite number of moments [24]. Hence we define generalized moments to represent swarm features.

Definition 1: Let $P(s^j)$ denote a polynomial function of elements in s^j. We define generalized moment (GM) $T^P : \boldsymbol{S} \to \mathbb{R}$ to represent a swarm feature, which is expressed as [18]

$$T^P(\boldsymbol{s}) = \frac{1}{N} \sum_{j=1}^{N} P(s^j). \tag{3}$$

For instance, if $P_1(s^j) = s_x^j$, the mean of the agents' x positions is defined as $\bar{s}_x \triangleq T^{P_1}(\boldsymbol{s}) = \frac{1}{N}\sum_{j=1}^{N} s_x^j$. Similarly, for $P_2(s^j) = s_y^j$, the mean of the y positions is $\bar{s}_y \triangleq T^{P_2}(\boldsymbol{s}) = \frac{1}{N}\sum_{j=1}^{N} s_y^j$. Due to space limitation, the readers are referred to [18] for more details about the efficacy of GMs. This letter focuses on $q \in \mathbb{Z}_{>0}$ (positive integer set) generalized moments, which are denoted as $\boldsymbol{\eta} = \boldsymbol{T^p(s)} = [\eta_1, \ldots, \eta_q]^T$.

B. Swarm Signal Temporal Logic (SwarmSTL)

The syntax of SwarmSTL is expressed as follows [18]:

$$\phi := \top | \pi | \neg\phi | \phi_1 \wedge \phi_2 | \phi_1 \vee \phi_2 | \phi_1 \mathcal{U}_{[k_1,k_2]} \phi_2 | \phi_1 \mathcal{S}_{[k_1,k_2]} \phi_2, \tag{4}$$

where $\top$ is Boolean True, π is an atomic proposition defined as $\pi := \boldsymbol{a}^T\boldsymbol{\eta} < c$, $\neg, \wedge, \vee$ are Boolean operators representing "negation," "conjunction" and "disjunction," respectively, $\mathcal{U}$ reads as "*Until*," $\mathcal{S}$ reads as "*Since*," $k_1, k_2 \in T$, $\boldsymbol{a} \in \mathbb{R}^q$ and $\|\boldsymbol{a}\| = 1$, $c \in \mathbb{R}$. Additionally, we define two useful temporal operators from $\mathcal{S}$: $\diamondsuit_{[k_1,k_2]}\phi := \top \mathcal{S}_{[k_1,k_2]} \phi$ (reads "eventually ϕ in the past") and $\boxdot_{[k_1,k_2]}\phi := \neg\diamondsuit_{[k_1,k_2]}\neg\phi$ (reads "always ϕ in the past"). We define "$\Rightarrow$" as an implication operator, which means $\phi_1 \Rightarrow \phi_2 := \neg\phi_1 \vee \phi_2$. Due to space limitation, we refer the readers to [18] for more details on the expressiveness of SwarmSTL. Note that SwarmSTL is a special case of STL where the predicates are defined over generalized moments to express specifications of collective behaviors of robot swarms.

The Boolean semantics of SwarmSTL can qualitatively measure the satisfaction of ϕ over $\boldsymbol{\eta}$ at k, and $(\boldsymbol{\eta}, k) \models \phi$ means $\boldsymbol{\eta}$ satisfies ϕ at k. The robustness degree of satisfaction of ϕ over $\boldsymbol{\eta}$ at k is denoted as $r(\boldsymbol{\eta}, \phi, k)$, which can quantitatively measure the satisfaction of ϕ over $\boldsymbol{\eta}$ at k.

Definition 2: The robustness degree of satisfaction, $r(\boldsymbol{\eta}, \phi, k)$, can be calculated through the quantitative semantics [2]:

$$\begin{aligned}
r(\boldsymbol{\eta}, \pi, k) &= c - \boldsymbol{a}^T\boldsymbol{\eta}(k), \\
r(\boldsymbol{\eta}, \neg\phi, k) &= -r(\boldsymbol{\eta}, \phi, k), \\
r(\boldsymbol{\eta}, \phi_1 \wedge \phi_2, k) &= \min(r(\boldsymbol{\eta}, \phi_1, k), r(\boldsymbol{\eta}, \phi_2, k)), \\
r(\boldsymbol{\eta}, \phi_1 \vee \phi_2, k) &= \max(r(\boldsymbol{\eta}, \phi_1, k), r(\boldsymbol{\eta}, \phi_2, k)), \\
r(\boldsymbol{\eta}, \phi_1 \mathcal{U}_{[k_1,k_2]} \phi_2, k) &= \sup_{k' \in [k+k_1, k+k_2]} (\min(r(\boldsymbol{\eta}, \phi_2, k'), \\
&\qquad \inf_{k'' \in [k+k_1, k']} r(\boldsymbol{\eta}, \phi_1, k''))), \\
r(\boldsymbol{\eta}, \phi_1 \mathcal{S}_{[k_1,k_2]} \phi_2, k) &= \sup_{k' \in [k-k_2, k-k_1]} (\min(r(\boldsymbol{\eta}, \phi_2, k'), \\
&\qquad \inf_{k'' \in [k,'k-k_1]} r(\boldsymbol{\eta}, \phi_1, k''))).
\end{aligned} \tag{5}$$

III. PROBLEM STATEMENT AND APPROACH

We consider an agent as a node and the communication links between agents as edges. Hence a robot swarm can be viewed as a graph $G = \{D, E\}$, where D denotes the set of nodes,

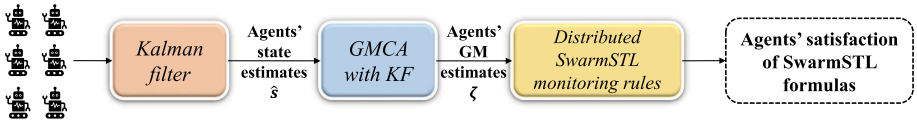


Fig. 1. The overall workflow for the distributed online monitoring algorithm.

and E denotes the set of edges. For instance, if Agent i can communicate with Agent j, then $(i, j) \in E$.

Assumption 1: Assume the communication between agents is asynchronous, i.e., at any k, only one agent is activate (initiates communication with another agent) and the probability of each agent being active is the same [19]. Also, each agent knows the fixed graph structure.

If the agents wish to monitor the satisfaction of SwarmSTL formulas, they need to first estimate the GMs and know the distance between their estimates and the actual GMs. Each agent can subsequently compute the satisfaction using the estimates, the distance, and the quantitative semantics. Essentially, we need to solve the following problems.

Problem 1: Design a distributed consensus algorithm for estimating the GMs with a guaranteed estimation error bound.

Problem 2: Design a set of distributed monitoring rules for SwarmSTL formulas based on the above algorithm such that each agent can compute the satisfaction of SwarmSTL formulas over its swarm features.

Eq. (3) indicates that a GM is the mean of $P(s^j)$. Hence Problem 1 can be posed as a distributed average consensus (DAC) problem. Previous works on DAC [19]–[21] will generate a steady-state error with the presence of measurement noise. Instead, we incorporate KF into the distributed GM consensus process. Each agent can estimate its own state using the KF and use its state estimate to perform the distributed GM consensus. We call this algorithm the GM consensus algorithm (GMCA) with KF. More details are presented in Section IV.

For Problem 2, we can use the estimation error bound from GMCA and the robustness degree of an agent's estimate with respect to π to compute an agent's confidence level of $\boldsymbol{\eta}$ satisfying π. By De Morgan's law and formula structure induction, we can derive the agent's confidence levels of satisfying other SwarmSTL formulas. More details are presented in Section V. The overall workflow for the distributed consensus-based online monitoring framework is shown in Fig. 1.

For clarity, we briefly review the KF. The measurement model of an agent is defined as

$$y(k) = s(k) + v(k), \tag{6}$$

where $y(k)$ is the measurement, $v(k)$ is the sensor noise. The trace of a matrix is denoted as $tr(\cdot)$, and the m-th largest eigenvalue of a matrix is denoted as $\lambda_m(\cdot)$.

Assumption 2: Assume $v(k)$ follows a Gaussian distribution with 0 mean and covariance matrix K_{v_a} that is time-invariant, and each agent knows that $E(\|v\|^2)$ is upper bounded by $v_{\max}$.

The state estimate of a swarm is denoted as $\hat{\boldsymbol{s}} = [(\hat{s}^1)^T, \ldots, (\hat{s}^N)^T]^T$, where $\hat{s}^j = [\hat{s}^j_x, \hat{s}^j_y]^T$ is the state estimate of the Agent j. The swarm measurement model is thus

$$\boldsymbol{y}(k) = \boldsymbol{s}(k) + \boldsymbol{v}(k), \tag{7}$$

where $\boldsymbol{y} = [(y^1)^T, \ldots, (y^N)^T]^T$, $\boldsymbol{v} = [(v^1)^T, \ldots, (v^N)^T]^T$, y^j and v^j are the measurement and the noise of Agent j, respectively. The covariance matrix of the state estimation error is $\Sigma = E[(\hat{\boldsymbol{s}} - \boldsymbol{s})(\hat{\boldsymbol{s}} - \boldsymbol{s})^T]$, and the covariance matrix of $\boldsymbol{v}$ is denoted as $K_v = E(\boldsymbol{v}\boldsymbol{v}^T)$, which is time-invariant. With Assumption 2, each agent knows that $E(\|\boldsymbol{v}\|^2) \leq N v_{\max}$.

Remark 1: In practice, we can use noise variance estimation techniques for KF such as [25] to obtain $v_{\max}$ without using Assumption 2.

Assumption 3: Assume each agent knows that $E(\|\hat{s}^j(0) - s^j(0)\|^2) \leq s_{\max}$.

The update of $\hat{\boldsymbol{s}}(k)$ and $\Sigma(k)$ is expressed as follows [26]:

$$\begin{aligned} K(k) &= \Sigma(k-1)(\Sigma(k-1) + K_v)^{-1}, \\ \hat{\boldsymbol{s}}(k) &= \hat{\boldsymbol{s}}(k-1) + \boldsymbol{u}(k-1) + K(k)(\boldsymbol{y}(k) \\ &\quad - \hat{\boldsymbol{s}}(k-1) - \boldsymbol{u}(k-1)), \\ \Sigma(k) &= (I_N - K(k))\,\Sigma(k-1), \end{aligned} \tag{8}$$

where $\hat{\boldsymbol{s}}(k) = E[\boldsymbol{s}(k)|\boldsymbol{Y}(k)]$, $\boldsymbol{Y}(k) = [(\boldsymbol{y}(0))^T, \ldots, (\boldsymbol{y}(k))^T]^T$, and I_N is the $N \times N$ identity matrix.

IV. Distributed Generalized Moments Consensus With Kalman Filter

This section presents how the agents use their state estimates from KF to update their estimates of GMs. For simplicity, we only discuss the consensus algorithm on one GM, and the same analysis can be applied to any GM. First, we derive some convergence properties and estimation error bounds, assuming that the swarm is stationary. These results are then generalized to obtain error bounds for our proposed GMCA.

A. Convergence Properties of GMCA on Static Swarms

In this section, we simplify Problem 1 by assuming that $\mathbf{u}(k) \equiv 0$, i.e.,

$$\boldsymbol{s}(k+1) = \boldsymbol{s}(k). \tag{9}$$

Hence the updates of $\hat{\boldsymbol{s}}(k)$ and $\Sigma(k)$ are described as

$$\begin{aligned} \hat{\boldsymbol{s}}(k) &= \hat{\boldsymbol{s}}(k-1) + K(k)\left(\boldsymbol{y}(k) - \hat{\boldsymbol{s}}(k-1)\right), \\ \Sigma(k) &= \left(I_N - \Sigma(k-1)(\Sigma(k-1) + K_v)^{-1}\right)\Sigma(k-1). \end{aligned} \tag{10}$$

In contrast to the special initialization $\zeta^j(0) = P(s^j(0))$ adopted in [19]–[21], each agent initializes its estimate of the GM as $\zeta^j(0) = P(\hat{s}^j(0))$. Asynchronous communication indicates that at each following k, there are two agents communicating with each other. The probability of each agent being active at k is $\frac{1}{N}$, where active means an agent can initiate communication with another agent. Let W be an $N \times N$ matrix associated with the robot swarm, where entry W_{ij} denotes the probability of Agent i communicating with Agent j.

Before proceeding into the details of the GM consensus process, we introduce some necessary notations. Let $\theta^j(k) = P(s^j(k))$ denote the polynomial function P applied to $s^j(k)$ and $\theta(k) = \boldsymbol{P}(\boldsymbol{s}(k)) = [\theta^1(k), \ldots, \theta^N(k)]^T$ denote the polynomial function $\boldsymbol{P}$ applied to $\boldsymbol{s}$. Similarly, if we apply P to the agents' state estimates $\hat{s}^j(k)$ and $\hat{\boldsymbol{s}}(k)$, we have the following notations: $\hat{\theta}^j(k) = P(\hat{s}^j(k))$, $\hat{\theta}(k) = \boldsymbol{P}(\hat{\boldsymbol{s}}(k)) = [\hat{\theta}^1(k), \ldots, \hat{\theta}^N(k)]^T$. The swarm's GM estimate at k is denoted as $\zeta(k) = [\zeta^1(k), \ldots, \zeta^N(k)]^T$, from which we know

TABLE II
THE VARIABLE NOTATIONS USED IN THE GMCA

Variable notation	Representation
s / $\boldsymbol{s}$	State of an agent / a swarm
u / $\boldsymbol{u}$	Input of an agent / a swarm
$\boldsymbol{\eta}$	Generalized moments
y / $\boldsymbol{y}$	Measurement of an agent / a swarm
v / $\boldsymbol{v}$	Sensor noise of an agent / a swarm
$\hat{s}$ / $\hat{\boldsymbol{s}}$	State estimate of an agent / a swarm
Σ	Covariance matrix of state estimation
K_v	Covariance matrix of noise
ζ	GM estimate of a swarm
θ / $\hat{\theta}$	Polynomial function P applied to $\boldsymbol{s}$ / $\hat{\boldsymbol{s}}$
$\bar{\eta}$ / $\bar{\theta}$	Actual GM
$\bar{\hat{\theta}}$	GM computed by swarm's state estimates

$\zeta(0) = \hat{\theta}(0)$. The update of $\hat{\theta}(k)$ is denoted as $\Delta\hat{\theta}(k) = \hat{\theta}(k) - \hat{\theta}(k-1)$. The mean of $\zeta(0)$, $\Delta\hat{\theta}(k)$ and $\theta(k)$ are denoted as $\bar{\zeta}(0) = \frac{\mathbf{1}^T}{N}\zeta(0), \Delta\bar{\hat{\theta}}(k) = \frac{\mathbf{1}^T}{N}\Delta\hat{\theta}(k), \bar{\eta}(k) = \bar{\theta}(k) = \frac{\mathbf{1}^T}{N}\theta(k)$, respectively. Table II shows a complete list of notations used in the GM consensus process.

The update scheme is that at $k+1$, if Agent i communicates with Agent j, then they update their estimates with

$$\zeta^i(k+1) = \frac{1}{2}\left(\zeta^i(k) + \zeta^j(k)\right) + \hat{\theta}^i(k+1) - \hat{\theta}^i(k),$$

$$\zeta^j(k+1) = \frac{1}{2}\left(\zeta^i(k) + \zeta^j(k)\right) + \hat{\theta}^j(k+1) - \hat{\theta}^j(k), \quad (11)$$

and the other agents update their estimates with

$$\zeta^p(k+1) = \zeta^p(k) + \hat{\theta}^p(k+1) - \hat{\theta}^p(k),\ (p \neq i, j). \quad (12)$$

Suppose $e^j = [0, \ldots, 1, \ldots, 0]^T \in \mathbb{R}^N$ with the j-th entry being 1 and all the other entries being 0, we can reformulate (11) and (12) in the vector form as

$$\zeta(k+1) = V(k) \cdot \zeta(k) + \Delta\hat{\theta}(k), \quad (13)$$

where $\zeta(0)$ is initialized as $\zeta(0) = [\zeta^1(0), \ldots, \zeta^N(0)]$, and $\zeta^j(0) = P(\hat{s}^j(0))$, and $V(k)$ has probability $\frac{1}{N}W_{ij}$ being

$$\mathcal{V}_{ij} = I_N - \frac{(e^i - e^j)(e^i - e^j)^T}{2}, \quad (14)$$

which is a symmetric doubly stochastic matrix. We denote the expectation of $V(k)$ as V, and the estimation error at k as $E(\|\zeta(k) - \bar{\eta}(k)\mathbf{1}\|_\infty)$, which can be decomposed as

$$E(\|\zeta(k) - \bar{\eta}(k)\mathbf{1}\|_\infty) = E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1} + \bar{\hat{\theta}}(k)\mathbf{1} -$$

$$\bar{\eta}(k)\mathbf{1}\|_\infty) \leq E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty) + E(\|\bar{\hat{\theta}}(k)\mathbf{1} - \bar{\eta}(k)\mathbf{1}\|_\infty), \quad (15)$$

where $\bar{\hat{\theta}}(k) = \frac{\mathbf{1}^T}{N}\hat{\theta}(k)$, and $\mathbf{1} \in \mathbb{R}^N$ denotes the vector of all ones. Hence we can prove the convergence of $E(\|\zeta(k) - \bar{\eta}(k)\mathbf{1}\|_\infty)$ via proving the convergence of $E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty)$ and $E(\|\bar{\hat{\theta}}(k)\mathbf{1} - \bar{\eta}(k)\mathbf{1}\|_\infty)$. Let's first analyze $E(\|\bar{\hat{\theta}}(k)\mathbf{1} - \bar{\eta}(k)\mathbf{1}\|_\infty)$. Using the fact that $P(s^j)$ is locally Lipschitz on S [27], it can be shown that $T^P(\boldsymbol{s})$ is locally Lipschitz on $\boldsymbol{S}$. This implies there is a constant $\mathcal{L}_1 \geq 0$ that

$$E(\|\bar{\hat{\theta}}(k)\mathbf{1} - \bar{\eta}(k)\mathbf{1}\|_\infty) = E(\|T^P(\hat{\boldsymbol{s}}(k)) - T^P(\boldsymbol{s}(k))\|_\infty),$$

$$\leq \mathcal{L}_1 E(\|\hat{\boldsymbol{s}}(k) - \boldsymbol{s}(k)\|_\infty), \quad (16)$$

where $E(\|\hat{\boldsymbol{s}}(k) - \boldsymbol{s}(k)\|_\infty) \leq E(\|\hat{\boldsymbol{s}}(k) - \boldsymbol{s}(k)\|) = \sqrt{tr(\Sigma(k))}$. Hence we can obtain an upper bound of $E(\|\bar{\hat{\theta}}(k)\mathbf{1} - \bar{\eta}(k)\mathbf{1}\|_\infty)$ by computing an upper bound of $tr(\Sigma(k))$. $\Sigma(k)$ can be reformulated as $\Sigma(k) = K_v(\Sigma(k-1) + K_v)^{-1}\Sigma(k-1)$, which leads to the following lemma.

Lemma 1: $\Sigma(k)$ follows the expression that

$$\Sigma(k) = K_v[K_v + k\Sigma(0)]^{-1}\Sigma(0), \quad (17)$$

and the state estimation error satisfies

$$E(\|\hat{\boldsymbol{s}}(k) - \boldsymbol{s}(k)\|_\infty) \leq \sqrt{\frac{N^2 s_{\max} v_{\max}}{(v_{\max} + k s_{\max})}}, \quad (18)$$

which converges to 0 asymptotically.

Proof 1: We can prove Lemma 1 by induction. At $k = 1$, we have $\Sigma(1) = K_v(K_v + \Sigma(0))^{-1}\Sigma(0) = ((K_v + \Sigma(0))K_v^{-1})^{-1}\Sigma(0) = (I + \Sigma(0)K_v^{-1})^{-1}\Sigma(0)$. Moreover, if $\Sigma(k) = K_v[K_v + k\Sigma(0)]^{-1}\Sigma(0) = (I + k\Sigma(0)K_v^{-1})^{-1}\Sigma(0)$ holds, we need to show $\Sigma(k+1) = K_v[K_v + (k+1)\Sigma(0)]^{-1}\Sigma(0)$ holds. This can be proved by

$$\begin{aligned}
&\Sigma(k+1)\\
&= K_v(\Sigma(k) + K_v)^{-1}(I + k\Sigma(0)K_v^{-1})^{-1}\Sigma(0),\\
&= K_v[(I + k\Sigma(0)K_v^{-1})\Sigma(k) + (K_v + k\Sigma(0))]^{-1}\Sigma(0),\\
&= K_v[(K_v + k\Sigma(0))K_v^{-1}\Sigma(k) + (K_v + k\Sigma(0))]^{-1}\Sigma(0),\\
&= K_v[(K_v + k\Sigma(0))((K_v + k\Sigma(0))^{-1}\Sigma(0) + I)]^{-1}\Sigma(0),\\
&= K_v\left[K_v + (k+1)\Sigma(0)\right]^{-1}\Sigma(0),\\
&= \left[\Sigma(0)^{-1} + (k+1)K_v^{-1}\right]^{-1}. \quad (19)
\end{aligned}$$

Therefore, $\Sigma(k) = K_v[K_v + k\Sigma(0)]^{-1}\Sigma(0)$, and we have

$$\begin{aligned}
\lambda_1(\Sigma(k)) &= \frac{1}{\lambda_N(\Sigma(0)^{-1} + k \times K_v^{-1})}\\
&\leq \frac{1}{1/\lambda_1(\Sigma(0)) + k/\lambda_1(K_v)},\\
&\leq \frac{1}{1/(N s_{\max}) + k/(N v_{\max})}, \quad (20)
\end{aligned}$$

which implies

$$tr(\Sigma(k)) \leq N\lambda_1(\Sigma(k)) \leq \frac{N^2 s_{\max} v_{\max}}{v_{\max} + k s_{\max}}. \quad (21)$$

The above results signify that $E(\|\hat{\mathbf{s}}(k) - \mathbf{s}(k)\|_\infty) \leq \sqrt{N^2 s_{\max} v_{\max}/(v_{\max} + k s_{\max})}$ and $E(\|\hat{\mathbf{s}}(k) - \mathbf{s}(k)\|_\infty)$ converges to 0 when $k \to \infty$.

Corollary 1: A direct result of Lemma 1 and (16) is that

$$E(\|\bar{\hat{\theta}}(k)\mathbf{1} - \bar{\eta}(k)\mathbf{1}\|_\infty) \leq \mathcal{L}_1\sqrt{\frac{N^2 s_{\max} v_{\max}}{(v_{\max} + k s_{\max})}}, \quad (22)$$

and $E(\|\bar{\hat{\theta}}(k)\mathbf{1} - \bar{\eta}(k)\mathbf{1}\|_\infty)$ converges to 0 asymptotically.

For brevity, let $\delta_{\max}(k) = (N^2 s_{\max} v_{\max})/(v_{\max} + k s_{\max})$. The next task is to prove $E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty)$ converges to 0. To prove this, we need the following proposition:

Proposition 1 [19]: The initial estimation error satisfies $E(\|\zeta(0) - \bar{\zeta}(0)\mathbf{1}\|^2) \leq E(\|\zeta(0)\|^2)$ with $\zeta(0)$ defined in (13). Also, the one-step estimation error of $\bar{\zeta}(0)$ satisfies $E(\|V(0)\zeta(0) - \bar{\zeta}(0)\mathbf{1}\|^2) = E(\|V(0)(\zeta(0) - \bar{\zeta}(0)\mathbf{1})\|^2) \leq \lambda_2^2(V)\|\zeta(0)\|^2$, and the k-step estimation error of $\bar{\zeta}(0)$ satisfies

$$E(\|V(k-1)\ldots V(0)\zeta(0) - \bar{\zeta}(0)\mathbf{1}\|^2) \leq \lambda_2^{2^{\wedge}k}(V)\|\zeta(0)\|^2. \quad (23)$$

Hereafter, we assume each agent knows that the initial estimation error is bounded, i.e. $E(\|\zeta(0) - \bar{\zeta}(0)\mathbf{1}\|_\infty) \leq \zeta_{\max}$.

Using Cauchy-Schwarz inequality and the fact that $E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty) \leq E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|)$, we have

$$E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty) \leq E(\|V(k-1)\ldots V(0)\zeta(0)+$$
$$V(k-1)\ldots V(1)\Delta\hat{\theta}(1) + \cdots + \Delta\hat{\theta}(k) - [\bar{\hat{\theta}}(k) - \bar{\hat{\theta}}(k-1)$$
$$+ \bar{\hat{\theta}}(k-1) - \bar{\hat{\theta}}(k-2)\ldots + \bar{\hat{\theta}}(1) - \bar{\zeta}(0) + \bar{\zeta}(0)]\mathbf{1}\|),$$
$$= E(\|V(k-1)\ldots V(0)\zeta(0) - \bar{\zeta}(0)\mathbf{1} + V(k-1)\ldots V(1)\Delta\hat{\theta}(1)$$
$$- \Delta\bar{\hat{\theta}}(1)\mathbf{1} + \cdots + \Delta\hat{\theta}(k) - \Delta\bar{\hat{\theta}}(k)\mathbf{1}\|),$$
$$\leq E(\|V(k-1)\ldots V(0)\zeta(0) - \bar{\zeta}(0)\mathbf{1}\|) + E(\|V(k-1)\ldots$$
$$V(1)\Delta\hat{\theta}(1) - \Delta\bar{\hat{\theta}}(1)\mathbf{1}\|) + \cdots + E(\|\Delta\hat{\theta}(k) - \Delta\bar{\hat{\theta}}(k)\mathbf{1}\|). \quad (24)$$

In order to prove $E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty)$ converges to 0, we need to compute an upper bound of the one-step GM estimation error $E(\|V(k-1)\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}\|)$.

Lemma 2: If the agents update their estimates using (13), then the one-step GM estimation error satisfies

$$E(\|V(k-1)\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}\|)$$
$$\leq \lambda_2(V)\mathcal{L}_2\sqrt{\delta_{\max}(k-1) + \delta_{\max}(k-2)},\ \mathcal{L}_2 \geq 0. \quad (25)$$

Proof 2: Let $\mathfrak{e}_{k-1} = V(k-1)\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}$. Using Jensen's inequality, we have the following result:

$$E(\|\mathfrak{e}_{k-1}\|) \leq \sqrt{E(\|\mathfrak{e}_{k-1}\|^2)},$$
$$= \sqrt{\sum_{\delta_v=-\infty}^{\infty}\sum_{i,j=1}^{N}\delta_v^T\mathcal{V}_{ij}^T\mathcal{V}_{ij}\delta_v\frac{1}{N}W_{ij}\cdot}$$
$$\sqrt{p[(\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}) = \delta_v]},$$
$$= \sqrt{\sum_{\delta_v=-\infty}^{\infty}\delta_v^T V^T V\delta_v p[(\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}) = \delta_v]}. \quad (26)$$

If we apply Proposition 1 to $\delta_v^T V^T V\delta_v$, we can write (26) as

$$E(\|V(k-1)\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}\|)$$
$$\leq \lambda_2(V)\sqrt{E(\|\Delta\hat{\theta}(k-1)\|^2)}, \quad (27)$$

where $p[(\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}) = \delta_v]$ denotes the probability of $\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1} = \delta_v$. As $\mathbf{P}(\mathbf{s})$ is locally Lipschitz on $\mathbf{S}$, $\exists\mathcal{L}_2 \geq 0$ such that

$$\sqrt{E(\|\Delta\hat{\theta}(k)\|^2)} = \sqrt{E(\|\mathbf{P}(\hat{\mathbf{s}}(k)) - \mathbf{P}(\hat{\mathbf{s}}(k-1))\|^2)},$$
$$\leq \mathcal{L}_2\sqrt{E(\|\hat{\mathbf{s}}(k) - \hat{\mathbf{s}}(k-1)\|^2)}. \quad (28)$$

Corollary 2: With Lemma 2 and Proposition 1, (24) becomes

$$E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty) \leq \lambda_2^k(V)\sqrt{N}\zeta_{\max} + \lambda_2^{k-1}(V)\mathcal{L}_2$$
$$\sqrt{\delta_{\max}(1) + \delta_{\max}(0)} + \cdots + \mathcal{L}_2\sqrt{\delta_{\max}(k) + \delta_{\max}(k-1)},$$
$$\leq \lambda_2^k(V)\sqrt{N}\zeta_{\max} + \mathcal{L}_2\sum_{k_t=1}^{k}\lambda_2^{k-k_t}(V)\sqrt{\delta_{\max}(k_t) + \delta_{\max}(k_t-1)}.$$

Also, $E(\|\zeta(k) - \bar{\hat{\theta}}(k)\mathbf{1}\|_\infty)$ converges to 0 asymptotically.

This is because when $k \to \infty$, $\delta_{\max}(k)$ and $\delta_{\max}(k-1)$ converge to 0. Also, V is a symmetric doubly stochastic matrix, thus $\lambda_2(V) < 1$ and $\lambda_2^k(V) \to 0$ when $k \to \infty$.

Theorem 3: If a robot swarm follows the dynamics in (9) and adopts (13) to update the GM estimates, then the agents' estimates of the GMs converge to the actual GMs asymptotically.

Proof 3: Straightforward from Corollary *1 and* Corollary *2.*

The estimation error bound for GMCA on a static swarm is

$$E(\|\zeta(k) - \bar{\eta}(k)\mathbf{1}\|_\infty) \leq \lambda_2^k(V)\sqrt{N}\zeta_{\max} +$$
$$\mathcal{L}_2\sum_{k_t=1}^{k}\lambda_2^{k-k_t}(V)\sqrt{\delta_{\max}(k_t) + \delta_{\max}(k_t-1)} + \mathcal{L}_1\sqrt{\delta_{\max}(k)}. \quad (29)$$

B. Estimation Error Bound for GMCA on Dynamic Swarms

This section generalizes the estimation error bound for static swarms to obtain the estimation error bound for dynamically moving swarms. The swarm follows the dynamic model in (2), and the updates of $\hat{\boldsymbol{s}}(k)$ and $\Sigma(k)$ are expressed as (8). Our goal is to obtain an upper bound of $E(\|\zeta(k) - \bar{\eta}(k)\mathbf{1}\|_\infty)$ with $\boldsymbol{u}(k) \neq 0$. With (8), we could write $\hat{\boldsymbol{s}}(k)$ as

$$\hat{\boldsymbol{s}}(k) = \hat{\boldsymbol{s}}(k-1) + \boldsymbol{u}(k-1)$$
$$+ K(k)(\boldsymbol{s}(k-1) - \hat{\boldsymbol{s}}(k-1) + \boldsymbol{v}(k)),$$

which leads to the following lemma.

Lemma 4: For GMCA on dynamic swarms, the one-step estimation error satisfies

$$E(\|V(k-1)\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}\|) \leq \lambda_2(V)\mathcal{L}_2$$
$$\sqrt{\delta_{\max}(k-1) + \delta_{\max}(k-2) + 2Nu_{\max}^2}. \quad (30)$$

Proof 4: For GMCA on dynamic swarms, (27) and (28) still hold. The upper bound for $\sqrt{E(\|\hat{\mathbf{s}}(k) - \hat{\mathbf{s}}(k-1)\|^2)}$ is

$$\sqrt{E(\|\hat{\mathbf{s}}(k) - \mathbf{s}(k) + \mathbf{s}(k-1) + \mathbf{u}(k-1) - \hat{\mathbf{s}}(k-1)\|^2)}$$
$$\leq \sqrt{tr(\Sigma(k)) + tr(\Sigma(k-1)) + \|\mathbf{u}(k-1)\|^2},$$
$$\leq \sqrt{\delta_{\max}(k) + \delta_{\max}(k-1) + 2Nu_{\max}^2}, \quad (31)$$

hence we have $E(\|V(k-1)\Delta\hat{\theta}(k-1) - \Delta\bar{\hat{\theta}}(k-1)\mathbf{1}\|) \leq \lambda_2(V)\mathcal{L}_2\sqrt{\delta_{\max}(k-1) + \delta_{\max}(k-2) + 2Nu_{\max}^2}$.

Corollary 3: Using Lemma 4, we could derive the dynamic estimation error bound with KF (DBKF) as below:

$$E(\|\zeta(k) - \bar{\eta}(k)\mathbf{1}\|_\infty) \leq \lambda_2^k(V)\sqrt{N}\zeta_{\max} + \mathcal{L}_2\sum_{k_t=1}^{k}\lambda_2^{k-k_t}(V)$$
$$\sqrt{\delta_{\max}(k_t) + \delta_{\max}(k_t-1) + 2Nu_{\max}^2} + \mathcal{L}_1\sqrt{\delta_{\max}(k)}.$$

For brevity, we denote

$$\rho(k) = \lambda_2^k(V)\sqrt{N}\zeta_{\max} + \mathcal{L}_2\sum_{k_t=1}^{k}\lambda_2^{k-k_t}(V)$$
$$\sqrt{\delta_{\max}(k_t) + \delta_{\max}(k_t-1) + 2Nu_{\max}^2} + \mathcal{L}_1\sqrt{\delta_{\max}(k)}, \quad (32)$$

which implies $\rho(k)$ depends on the maximal allowed velocity of the agents but is independent of the agents' exact motion. The above results imply that the GMCA with KF can be run in parallel with other motion planning algorithms. For q GMs denoted by

Algorithm 1: Generalized Moments Consensus Algorithm with Kalman Filter *(GMCAKF)*.

1: **for** k = 1,...,K **do**
2: Suppose Agent i is active. Agent i selects a neighbor j using the probability distribution defined by W
3: Agents i and j update the estimates of GMs with (11)
4: Other agents update their estimates of GMs with (12)
5: Each agent updates the DBKF with (32)
6: **end for**

$\boldsymbol{\eta}$, we let $\boldsymbol{\zeta}^j = [\zeta_1^j, \ldots, \zeta_q^j]^T$ denote Agent j's estimate of $\boldsymbol{\eta}$, and $\rho_i (1 \leq i \leq q)$ denote the estimation error bound of η_i, then the estimation error bound of $\boldsymbol{\eta}$ is denoted as $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_q]^T$. The description of GMCA with KF is presented in Algorithm 1. In Section V, we will explain how to compute an agent's confidence level of swarm features satisfying SwarmSTL formulas using the DBKF.

Remark 2: The GMCA with KF still works for robot swarms with dynamically changing graph structures if a less-than-one upper bound of $\lambda_2(V)$ is known.

C. Optimizing the Convergence Rate

From (29) we know the convergence rate of the estimation error is controlled by $\lambda_2(V)$. The fastest convergence rate can be achieved by solving the following Semidefinite Programming (SDP) problem [19]:

$$\begin{aligned} \text{minimize} \quad & \epsilon, \\ \text{subject to} \quad & W_{ij} \geq 0,\ W_{ij} = 0 \text{ if} (i,j) \notin E, \\ & V = \frac{1}{N} \sum_{i,j=1}^{N} W_{ij} \mathcal{V}_{ij},\ \sum_j W_{ij} = 1, \forall i, \\ & V - \frac{1}{N} \mathbf{1}\mathbf{1}^T \preceq \epsilon I_N, \epsilon \in \mathbb{R}, \end{aligned} \tag{33}$$

where $V - \frac{1}{N}\mathbf{1}\mathbf{1}^T \preceq \epsilon I_N$ means $(\epsilon I_N - V + \frac{1}{N}\mathbf{1}\mathbf{1}^T)$ is positive semidefinite. With Assumption 1, the agents can know the graph structure information contained in W and optimize $\lambda_2(V)$ independently. Here we use the SDP mode in the cvx[1] toolbox to solve (33).

V. Distributed Monitoring With GMCA

In this work, we consider the problem of distributed online monitoring as the agents move dynamically in the environment. The goal is to compute the confidence level of satisfying ϕ by using the past trajectories until the current time slot k_c. For a future time slot $k > k_c$, $\rho(k)$ can be computed by (32). At k_c, Agent j knows $\boldsymbol{\zeta}^j(k_c)$ and $\boldsymbol{\rho}(k_c)$, hence we have $\boldsymbol{\eta}(k_c) \in [\boldsymbol{\zeta}^j(k_c) - \boldsymbol{\rho}(k_c), \boldsymbol{\zeta}^j(k_c) + \boldsymbol{\rho}(k_c)]$. The Lipschitz continuity of $T^p(\boldsymbol{s})$ implies that $\boldsymbol{\eta}(k) \in [\boldsymbol{\eta}(k_c) - \mathcal{L}_1(k - k_c)u_{\max}, \boldsymbol{\eta}(k_c) + \mathcal{L}_1(k - k_c)u_{\max}]$. As $\boldsymbol{\rho(k)}$ is known from (32), we have $\boldsymbol{\zeta}^j(k) \in [\boldsymbol{\eta}(k) - \boldsymbol{\rho}(k), \boldsymbol{\eta}(k) + \boldsymbol{\rho}(k)]$. By combining the above ranges for $\boldsymbol{\eta}(k_c), \boldsymbol{\eta}(k)$, and $\boldsymbol{\zeta}^j(k)$, we have $\boldsymbol{\zeta}^j(k) \in [\boldsymbol{\zeta}^j(k_c) - \boldsymbol{\rho}(k_c) - \mathcal{L}_1(k - k_c)u_{\max} - \boldsymbol{\rho}(k), \boldsymbol{\zeta}^j(k_c) + \boldsymbol{\rho}(k_c) + \mathcal{L}_1(k - k_c)u_{\max} + \boldsymbol{\rho}(k)]$. Correspondingly, we can compute the range of robustness of $\boldsymbol{\zeta}^j(k)$ with respect to π as $r(\boldsymbol{\zeta}^j, \pi, k) = M^j(k)$, where $M^j(k) = \{c - \boldsymbol{a}^T\boldsymbol{\zeta}^j(k) | \boldsymbol{\zeta}^j(k) \in [\boldsymbol{\zeta}^j(k_c) - \boldsymbol{\rho}(k_c) - \mathcal{L}_1(k - k_c)u_{\max} - \boldsymbol{\rho}(k), \boldsymbol{\zeta}^j(k_c) + \boldsymbol{\rho}(k_c) + \mathcal{L}_1(k - k_c)u_{\max} + \boldsymbol{\rho}(k)]\}$. The robustness of $\boldsymbol{\zeta}^j$ for π at k, depending on whether $k \leq k_c$, is expressed as

$$m^j(k) = \begin{cases} r(\boldsymbol{\zeta}^j, \pi, k) \text{ if } k \leq k_c, \\ M^j(k), \quad \text{otherwise.} \end{cases} \tag{34}$$

With (34), Agent j's confidence level of the swarm features satisfying SwarmSTL formulas, i.e., the distributed monitoring rules, are presented in Lemma 5.

Lemma 5: Agent j's confidence level of $\boldsymbol{\eta}$ satisfying ϕ at k is expressed as follows:

$$Pr_j((\boldsymbol{\eta}, k) \models \pi) \geq \begin{cases} 1 - \frac{\max\{\rho_i(k)\}}{m^j(k)}, & \text{if } m^j(k) > \rho_i(k) \\ & \forall i, \text{ and } k \leq k_c, \\ 1 - \frac{\max\{\rho_i(k)\}}{\min\limits_{m^j(k) \in M^j(k)} m^j(k)}, & \text{if } \min\limits_{m^j(k) \in M^j(k)} \\ & m^j(k) > \rho_i(k)\ \forall i, \text{ and } k > k_c, \\ 0, & \text{otherwise,} \end{cases}$$

$$\begin{aligned} & Pr_j\left((\boldsymbol{\eta}, k) \models \phi_1 \wedge \phi_2\right) \geq Pr_j((\boldsymbol{\eta}, k) \models \phi_1) \\ & + Pr_j((\boldsymbol{\eta}, k) \models \phi_2) - 1, \\ & Pr_j\left((\boldsymbol{\eta}, k) \models \phi_1 \vee \phi_2\right) \geq 1 - \min\{1 - Pr_j((\boldsymbol{\eta}, k) \models \phi_1), \\ & 1 - Pr_j((\boldsymbol{\eta}, k) \models \phi_2)\}, \\ & Pr_j\left((\boldsymbol{\eta}, k) \models \phi_1 \mathcal{S}_{[k_1,k_2]} \phi_2\right) \geq 1 - \min_{k' \in [k-k_2, k-k_1]} \{1 - \\ & Pr_j((\boldsymbol{\eta}, k') \models \phi_2) + \sum_{k''=k'}^{k-k_1} 1 - Pr_j((\boldsymbol{\eta}, k'') \models \phi_1)\}, \\ & Pr_j\left((\boldsymbol{\eta}, k) \models \phi_1 \mathcal{U}_{[k_1,k_2]} \phi_2\right) \geq 1 - \min_{k' \in [k+k_1, k+k_2]} \{1 - \\ & Pr_j((\boldsymbol{\eta}, k') \models \phi_2) + \sum_{k''=k+k_1}^{k'} 1 - Pr_j((\boldsymbol{\eta}, k'') \models \phi_1)\}, \end{aligned} \tag{35}$$

where $Pr_j((\boldsymbol{\eta}, k) \models \phi)$ denotes Agent j's confidence level of $\boldsymbol{\eta}$ satisfying ϕ at k.

Proof 5: Let $\boldsymbol{\theta}_i = [\theta_i^1, \ldots, \theta_i^N]^T (1 \leq i \leq q)$ and $\eta_i = \frac{\mathbf{1}^T}{N}\boldsymbol{\theta}_i$, where $\theta_i^j = P_i(s^j)$, and $\zeta_i = [\zeta_i^1, \ldots, \zeta_i^N]^T$ denote the agents' estimates of η_i. Using Markov's inequality, we know the estimation error of η_i satisfies $Pr_j(\|\zeta_i(k) - \eta_i(k)\mathbf{1}\|_\infty \leq m^j(k)) \geq 1 - E(\|\zeta_i(k) - \eta_i(k)\mathbf{1}\|_\infty)/m^j(k) \geq 1 - \rho_i(k)/m^j(k)$. For q GMs, we take the maximum of ρ_i as the estimation error bound for $\boldsymbol{\eta}$. Hence for an atomic proposition π, Agent j's confidence level of $\boldsymbol{\eta}$ satisfying π at $k \leq k_c$ is $Pr_j(\boldsymbol{\eta}, k) \models \pi) \geq 1 - \max_{1 \leq i \leq q}\{\rho_i(k)\}/m^j(k)$ if $m^j(k) > \rho_i(k), \forall i$. Otherwise, it is 0. The confidence level of π over $\boldsymbol{\eta}$ at $k > k_c$ can be proved analogously. With the confidence level of π, Agent j's confidence levels of $\boldsymbol{\eta}$ satisfying other operators at k are straightforward to show using formula structure induction.

Remark 3: There is a trade-off between the motion planning and the GMCA. Increasing the maximal allowed velocity $u_{\max}$ would increase $\rho(k)$ and the time to reach the desired confidence level, which can be observed in (32) and (35).

Remark 4: There are two potential sources of conservatism in the agents' estimates: the estimation error bound in (32) and

[1][Online]. Available: http://cvxr.com/cvx/doc/sdp.html

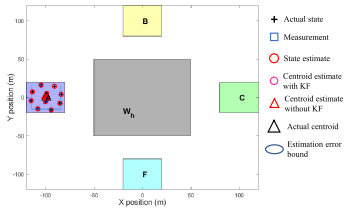


Fig. 2. A snapshot of simulation for the transportation task.

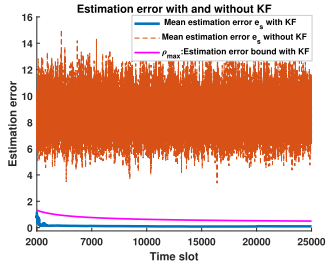


Fig. 3. Progression of $\rho_{\max}$ and e_s of the algorithms with and without KF.

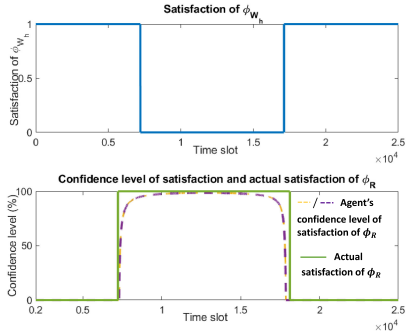


Fig. 4. Progression of agents' confidence levels of satisfaction of ϕ_R, and the actual satisfaction of ϕ_{W_h}, ϕ_R on $\bar{s}$. The solid green line is the actual satisfaction of ϕ_R, and the dashed lines represent agents' confidence levels of satisfaction of ϕ_R.

the confidence level of satisfaction in (35), which is based on the Markov's inequality.

VI. Case Study

We now use a swarm transporting supplies example to demonstrate how the distributed monitoring algorithm works. Suppose a virus outbreak happens in four regions A, B, C, F, the swarm needs to transport medical supplies from the warehouse W_h to these regions back and forth. Specifically, the swarm first transports the supplies from W_h to A. Next, it returns to W_h to take another supply and transports it to B. It subsequently follows the same idea to transport the supplies to C and F. The swarm follows a trajectory in the sequence of $W_h \to A \to W_h \to B \to W_h \to C \to W_h \to F \to W_h$. A snapshot of the simulation is shown in Fig. 2. In the simulation, we set $N = 10$, $u_{\max} = 0.01$, and adopt a flocking model [28] and a potential field approach to accomplish the motion planning. The motion planning of the swarm is based on the following rules:

1) cohesion: $u_1^j(k) = u_1^j(k-1) + \alpha_1 \sum_{i=1, i\neq j}^{N} (s^i(k) - s^j(k))$;
2) separation: $u_2^j(k) = u_2^j(k-1) - \alpha_2 \sum_{i=1, i\neq j}^{N} \frac{s^i(k) - s^j(k)}{\|s^i(k) - s^j(k)\|}$;
3) alignment: $u_3^j(k) = u_3^j(k-1) + \frac{\alpha_3}{N-1} \sum_{i=1, i\neq j} u^i(k-1)$;
4) attraction: $u_4^j(k) = \alpha_4(\mathcal{G} - s^j(k))$;

where α_1 to α_4 are positive tuning parameters, $u_t^j(k) = 0, \forall k = 1, \ldots, 4$ when $k < 0$, $\mathcal{G} \in S$ is the goal position. Aggregating the above rules gives $\hat{u}^j(k) = [\hat{u}_x^j(k), \hat{u}_y^j(k)]^T = \sum_{t=1}^{4} u_t^j(k)$. Agent j's velocity is then calculated as $u^j(k) = \hat{u}^j(k) * u_{\max} / \max\{\hat{u}_x^j(k), \hat{u}_y^j(k)\}$. A video of the simulation process is uploaded to https://tinyurl.com/swarmstlmonitor.

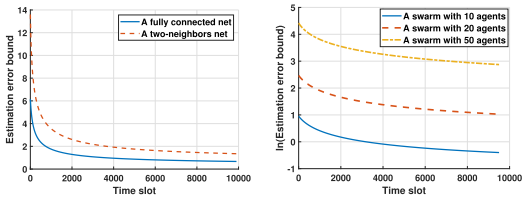
As the robot swarm needs to transport the supplies from W_h to the four regions back and forth, it cannot stay in W_h for too long. Hence we propose a SwarmSTL formula $\phi_R := \Diamond_{[1000,2000]} \phi_{W_h} \Rightarrow \Diamond_{[0,800]} \neg \phi_{W_h}$, which specifies the robot swarm to stay in W_h for at most 1000 time slots, where $\phi_{W_h} := \bar{s}_x \geq -50 \wedge \bar{s}_x \leq 50 \wedge \bar{s}_y \geq -50 \wedge \bar{s}_y \leq 50$. This interpretation can be obtained using the rule that $\phi_1 \Rightarrow \phi_2 := \neg\phi_1 \vee \phi_2$. ϕ_R reads as "If $\exists k' \in [1000, 2000]$ such that ϕ_{W_h} is satisfied at $k - k'$, then $\exists k'' \in [0, 800]$ such that $\neg\phi_{W_h}$ is satisfied at $(k - k'')$". The agents need to perform the consensus on the centroid $\bar{s} = [\bar{s}_x, \bar{s}_y]^T$. At $k = 0$, each agent sets $\boldsymbol{\zeta}^j(0) = [\zeta_x^j(0), \zeta_y^j(0)]^T = [\hat{s}_x^j(0), \hat{s}_y^j(0)]^T$, and computes the optimal $\lambda_2(V)$ by solving the SDP problem in (33).

The polynomial function P applied to the agents' state estimates is denoted as $\hat{\boldsymbol{s}}_x = [\hat{s}_x^1, \ldots, \hat{s}_x^N]^T$, $\hat{\boldsymbol{s}}_y = [\hat{s}_y^1, \ldots, \hat{s}_y^N]^T$, and the swarm's estimate of the centroid is denoted as $\zeta_x = [\zeta_x^1, \ldots, \zeta_x^N]^T$, $\zeta_y = [\zeta_y^1, \ldots, \zeta_y^N]^T$. At $k+1$, the agents use the KF update in (8) to update $\hat{\boldsymbol{s}}_x$ and $\hat{\boldsymbol{s}}_y$, and then update their estimates of the centroid using

$$\zeta_{x(y)}(k+1) = V(k)\zeta_{x(y)}(k) + \hat{\boldsymbol{s}}_{x(y)}(k+1) - \hat{\boldsymbol{s}}_{x(y)}(k). \tag{36}$$

Simultaneously, the agents compute the estimation error bound $\rho_{\max} = \max\{\rho_x, \rho_y\}$ and the confidence level of $\bar{s}$ satisfying ϕ_R using the monitoring rules in (35), where ρ_x, ρ_y are the estimation error bounds of $\bar{s}_x, \bar{s}_y$, respectively. The purpose of using the flocking model and the potential field method is to show the agents can perform the distributed monitoring and the flocking and motion planning tasks simultaneously because $\rho_{\max}$ is independent of the agents' exact motion.

To show the efficacy of the KF in the GM consensus process, we also perform a centroid consensus algorithm without KF, i.e. using $\boldsymbol{y}$ to update ζ_x and ζ_y. The actual mean estimation error is defined as $e_s = \frac{1}{N} \sum_{j=1}^{N} \|\boldsymbol{\zeta}^j - \bar{s}\|$. Fig. 3 shows the progression of $\rho_{\max}$ and e_s of the distributed centroid consensus algorithm with and without KF, from which we can observe e_s of the consensus algorithm with KF is smaller than e_s of the consensus algorithm without KF, and $\rho_{\max}$ decreases as k increases. The satisfaction of $\bar{s}$ with respect to ϕ_{W_h} is shown in Fig. 4 (top). Along the y axis, 1 represents that $\bar{s}$ satisfies ϕ_{W_h}, and 0 represents that $\bar{s}$ violates ϕ_{W_h}. To be more informative, we only show the satisfaction for $k \in [1000, 25000]$. Using the satisfaction of ϕ_{W_h} over $\bar{s}$, we can compute the satisfaction of ϕ_R over $\bar{s}$. The agents' confidence levels of $\bar{s}$ satisfying ϕ_R and the actual satisfaction of $\bar{s}$ over ϕ_R are shown in Fig. 4 (bottom), where the dashed lines represent the agents' confidence levels that may have overlap. When $k \in [8000, 17120]$, the swarm is outside of W_h, so $\bar{s}$ satisfies ϕ_R. The agents' confidence levels of satisfaction are also higher than 95%. In the simulation, the swarm actually stays in W_h longer than 1000 time slots. Hence



(a) Network topology effect. (b) Swarm size effect.

Fig. 5. Effect of network topology and swarm size on the estimation error bound.

we can observe that $\bar{s}$ satisfies ϕ_{W_h} for $k \in [17120, 24000]$ and violates ϕ_R for $k \in [18120, 25000]$, where the agents' confidence levels of satisfaction are also 0. These results show that the agents can monitor the satisfaction of swarm features with respect to SwarmSTL formulas using the GMCA with KF and the distributed monitoring rules.

Furthermore, we investigate the influence of the network topology and the swarm size on the estimation error bound. The estimation error bounds for a fully connected network and a two-neighbors network where each agent can communicate with its two nearest neighbors are shown in Fig. 5(a). Apparently, as an agent has more neighbors to share information, the estimation error bound will decrease faster. The estimation error bounds for a swarm with 10, 20, or 50 agents are shown in Fig. 5(b), in which we observe that at the same k, increasing the swarm size would increase the estimation error bound, which matches the pattern in (32).

VII. Conclusion

This letter investigates the challenge of distributed monitoring of satisfaction of SwarmSTL formulas over swarm features. A GMCA with KF is developed to allow each agent to estimate the GMs. The convergence properties of GMCA on static swarms are presented and an upper bound of the estimation error between agents' estimates and the actual GMs is derived. Additionally, the GMCA with KF can be performed in conjunction with other motion planning and control algorithms. The monitoring rules for SwarmSTL formulas are also proposed based on the estimation error bounds. The outcome of this work is that the agents can monitor whether the swarm satisfies a SwarmSTL formula in a distributed manner. The proposed method is applied to a swarm transporting supplies example, where the efficacy of the KF is also shown.

References

[1] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: A review from the swarm engineering perspective," *Swarm Intell.*, vol. 7, no. 1, pp. 1–41, 2013.

[2] A. Donze and O. Maler, "Robust satisfaction of temporal logic over real-valued signals," in *Proc. Int. Conf. Formal Model. Anal. Timed Syst.*, 2010, pp. 92–106.

[3] S. Karaman and E. Frazzoli, "Vehicle routing problem with metric temporal logic specifications," in *Proc. IEEE 47th Conf. Decis. Control*, 2008, pp. 3953–3958.

[4] C. I. Vasile, X. Li, and C. Belta, "Reactive sampling-based path planning with temporal logic specifications," *Int. J. Robot. Res.*, vol. 39, no. 8, pp. 1002–1028, 2020.

[5] L. Lindemann, G. J. Pappas, and D. V. Dimarogonas, "Reactive and risk-aware control for signal temporal logic," in *IEEE Trans. Autom. Control*, early access, Oct. 15, 2021, doi: 10.1109/TAC.2021.3120681.

[6] Y. E. Sahin, P. Nilsson, and N. Ozay, "Multirobot coordination with counting temporal logics," *IEEE Trans. Robot.*, vol. 36, no. 4, pp. 1189–1206, Aug. 2020.

[7] Y. Kantaros and M. M. Zavlanos, "STyLuS*: A temporal logic optimal control synthesis algorithm for large-scale multi-robot systems," *Int. J. Robot. Res.*, vol. 39, no. 7, pp. 812–836, 2020.

[8] T. Zheng, Z. Liu, and H. Lin, "Complex pattern generation for swarm robotic systems using spatial-temporal logic and density feedback control," in *Proc. IEEE Amer. Control Conf.*, 2020, pp. 5301–5306.

[9] C. Dixon, A. F. Winfield, M. Fisher, and C. Zeng, "Towards temporal verification of swarm robotic systems," *Robot. Auton. Syst.*, vol. 60, no. 11, pp. 1429–1441, 2012.

[10] A. F. Winfield, J. Sa, M.-C. Fernandez-Gago, C. Dixon, and M. Fisher, "On formal specification of emergent behaviours in swarm robotic systems," *Int. J. Adv. Robotic Syst.*, vol. 2, no. 4, 2005, Art. no. 39.

[11] A. Donze, T. Ferrere, and O. Maler, "Efficient robust monitoring for STL," in *Proc. Int. Conf. Comput. Aided Verification*, 2013, pp. 264–279.

[12] J. V. Deshmukh, A. Donze, S. Ghosh, X. Jin, G. Juniwal, and S. A. Seshia, "Robust online monitoring of signal temporal logic," *Formal Methods Syst. Des.*, vol. 51, no. 1, pp. 5–30, 2017.

[13] P. Thati and G. Roşu, "Monitoring algorithms for metric temporal logic specifications," *Electron. Notes Theor. Comput. Sci.*, vol. 113, pp. 145–162, 2005.

[14] G. E. Fainekos and G. J. Pappas, "Robustness of temporal logic specifications for continuous-time signals," *Theor. Comput. Sci.*, vol. 410, no. 42, pp. 4262–4291, 2009.

[15] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems*. Berlin, Germany: Springer, 2004, pp. 152–166.

[16] S. Moarref and H. Kress-Gazit, "Automated synthesis of decentralized controllers for robot swarms from high-level temporal logic specifications," *Auton. Robots*, vol. 44, no. 3, pp. 585–600, 2020.

[17] R. Yan and A. Julius, "A decentralized B&B algorithm for motion planning of robot swarms with temporal logic specifications," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7389–7396, Oct. 2021.

[18] R. Yan, Z. Xu, and A. Julius, "Swarm signal temporal logic inference for swarm behavior analysis," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 3021–3028, Jul. 2019.

[19] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006.

[20] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Dynamic consensus on mobile networks," in *Proc. IFAC World Congr.*, 2005, pp. 1–6.

[21] M. Zhu and S. Martínez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[22] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proc. IEEE 46th Conf. Decis. Control*, 2007, pp. 5492–5498.

[23] X. He, C. Hu, Y. Hong, L. Shi, and H.-T. Fang, "Distributed Kalman filters with state equality constraints: Time-based and event-triggered communications," *IEEE Trans. Autom. Control*, vol. 65, no. 1, pp. 28–43, Jan. 2020.

[24] V. John, I. Angelov, A. Oncul, and D. Thevenin, "Techniques for the reconstruction of a distribution from a finite number of its moments," *Chem. Eng. Sci.*, vol. 62, no. 11, pp. 2890–2904, 2007.

[25] S. Park, M.-S. Gil, H. Im, and Y.-S. Moon, "Measurement noise recommendation for efficient Kalman filtering over a large amount of sensor data," *Sensors*, vol. 19, no. 5, 2019, Art. no. 1168.

[26] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Basic Eng.*, vol. 82, pp. 35–45, 1960.

[27] H. K. Khalil, *Nonlinear Systems*, vol. 3. Englewood Cliffs, NJ, USA: Prentice Hall, 2002.

[28] C. W. Reynolds, "Flocks, herds, and schools: A distributed behavioral model," *Comput. Graph.*, vol. 21, no. 4, pp. 25–34, 1987.