Resilient Multi-Robot Multi-Target Tracking

Ragesh K. Ramachandran[1], Nicole Fronda[2], James A. Preiss[1], Zhenghao Dai[1] and Gaurav S. Sukhatme[1,3]

***Abstract*—We address the problem of ensuring resource availability in a networked multi-robot system performing distributed target tracking. Specifically, we consider a multi-target tracking scenario where the targets are driven by exogenous inputs that are unknown to the robots performing the tracking task. Robots track the positions of targets using a form of the Distributed Kalman Filter (DKF). We use the trace of each robot's sensor measurement noise covariance matrix as a measure of its sensing quality. When a robot's sensing quality deteriorates, the team's communication graph is modified by adding edges such that the robot with deteriorating sensor quality may share information with other robots to improve the team's target tracking ability. This computation is performed centrally and is designed to work without a large change in the number of active inter-robot communication links. Our method generates coordinates for the robots such the new communication graph can be realized in 3D. To achieve this, we propose two mixed integer semi-definite programming formulations, namely an 'agent-centric' strategy and a 'team-centric' strategy. We implement both formulations and a greedy, baseline strategy in simulation. Our simulation results show that the team-centric approach outperforms both agent-centric and greedy methods. Additionally, we show the effectiveness of our method in real-world settings through a multi-robot experiment performed in real-time.**

***Note to Practitioners*—This paper is motivated by the need to track multiple targets by means of a multi-robot team. When robots in the team experience degradation in their sensing quality our method reconfigures the team's communication graph by repositioning robots. This allows the robot with deteriorated sensing quality to benefit from sensor measurements from its neighbors. In previous work, we solved this problem under the strong assumption the targets moved under inputs that were known to the tracking team. In this paper we remove this assumption. Our experiments show comparable tracking accuracy for both settings, suggesting that this strong assumption is not always necessary for good tracking results. Our method is straightforward to implement in Python using off-the-shelf optimization solvers. We assume that the team performs target tracking using a distributed algorithm, but has access to a powerful centralized base station to solve the computationally expensive underlying optimization problems. Developing a fully decentralized method and finding better ways to solve the underlying optimization problems are potential directions for future work.**

***Index Terms*—Multi robot system, resilience, distributed multi-target tracking**

[1]Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA. email: rageshku|jpreiss|zhenghad|gaurav@usc.edu

[2]Department of Electrical and Computer Engineering, Oregon State University email: frondan@oregonstate.edu

[3]G.S. Sukhatme holds concurrent appointments as a Professor at USC and as an Amazon Scholar. This paper describes work performed at USC and is not associated with Amazon.

This work was supported in part by the Army Research Laboratory as part of the Distributed and Collaborative Intelligent Systems and Technology (DCIST) Collaborative Research Alliance (CRA).

I. Introduction

MULTI-TARGET tracking plays a central role in many civilian and military applications such as autonomous driving, disaster response, convoy protection *etc.*. Tracking multiple targets using a robot team is potentially attractive for scalability [1]. As a result, there has been burgeoning interest in the multi-robot multi-target tracking problems [2]–[5]. In a multi-robot setting, even though each robot may be limited in terms of sensing, computational capabilities and field of view of the environment, the target tracking task could be performed collaboratively by exchanging information among team members. A necessary condition for this is the existence of an underlying communication network. Collaborative distributed tracking avoids the need for a central data fusion center entrusted with the task of combining data collected from individual robots to estimate the states of the targets. A common approach for distributed multi-robot target tracking is the distributed Kalman filter [3]; we adopt this method in our work. We envision a scenario in which a team of robots tracks and estimates the state of a constant number of targets in an environment. We assume that each tracker robot is equipped with a single sensor allowing it to uniquely and distinctively identify all targets in its field of view. Measurements from the sensor on each tracker robot constitute the inputs to a distributed Kalman filter (DKF). The performance of each tracker robot is determined using a metric that quantifies its ability to obtain accurate measurements from its sensor - specifically, we use the trace of the sensor's measurement noise covariance matrix as a measure. A tracker robot's performance is considered to be reduced if the trace of its sensor measurement noise covariance matrix increases. The robot team is monitored by a base station which intervenes in the team's activities only when the performance of a robot in the team is reduced. We focus on strategies to mitigate the effect of a robot's sensor quality deterioration on the team's overall tracking performance by modifying the topology of the underlying communication network and target state measurement fusion weights. Figure 1 illustrates the settings discussed in the paper.

In a control theoretic sense, the state estimation error of the DKF depends upon the observability properties of the underlying communication network [6]. Significant prior research has gone into understanding and quantifying the effect of network topology on a network's observability properties [7]–[9]. These findings inspired us to investigate the idea of employing network reconfiguration to mitigate the effect of sensor quality deterioration on multi-target tracking performance.

This article builds on the abstract framework developed in [10] by adapting it to multi-robot target tracking. Similar

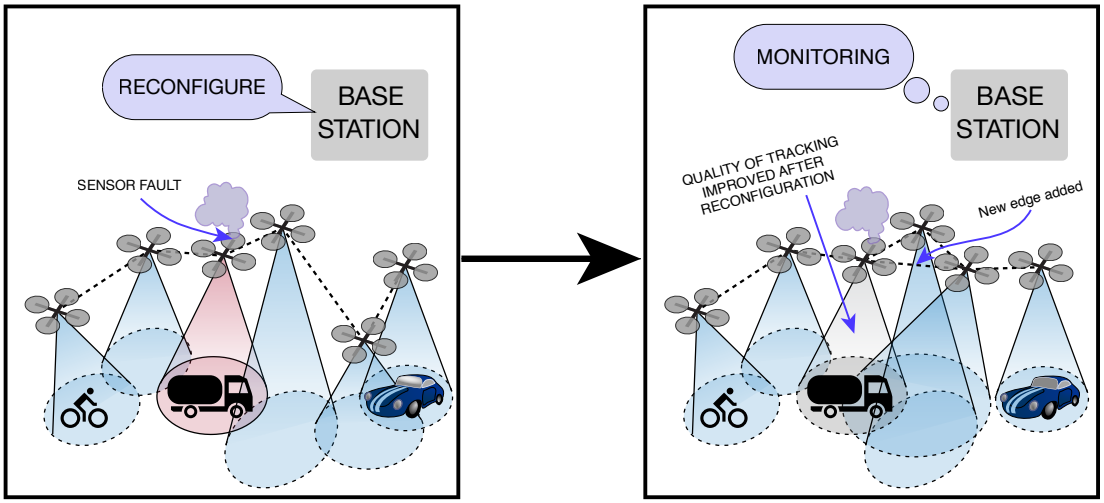


Fig. 1: Reconfiguration of communication network topology in a multi-target tracking scenario. (*Left*) One of the tracker robots in the team experiences a sensor fault due to external environmental factors, thereby affecting the team's overall tracking performance. The base station (which monitors the team) generates a new communication graph for the robots and commands the tracker robots to reconfigure themselves according to the new communication graph. (*Right*) The team has reconfigured to realize the new communication graph in 3D. Consequently, the team's tracking performance has improved compared to its tracking performance that existed immediately after the robot's sensor fault. The base station continues to monitor the team.

to [10], we propose a two-stage strategy to alleviate the impact of a robot's sensor quality deterioration on target tracking performance. When a robot's sensing deteriorates (as measured by its measurement noise covariance matrix), the first stage of our method updates the team's communication network topology and a set of measurement fusion weights for each robot such that the new topology is adjacent to the original with improved target tracking performance. We consider two network topologies to be adjacent to each other if they have identical nodes and the norm of the difference in their adjacency matrices is below a predefined value. Based on the sensing quality, our method identifies the neighbors of each robot and a set of associated weights such that each robot can optimally fuse its sensor measurements with the measurements obtained by its neighbors thereby improving the tracking performance of the team. The method depends on establishing a relationship between the sensing quality of each robot and the overall tracking performance of the team. Common metrics employed to quantify performance of Kalman filter-based tracking systems are measures on the state error covariance matrix such as its *trace* , *determinant*, *condition number etc.* [11]. In our previous work [12], we have considered a special case of the tracking problem - one where a (single) target is driven by exogenous inputs that are known to the tracker robots. In this simplified setting, the state error covariance matrix can be expressed explicitly as a function of the robot's sensor noise covariance matrix. Thus the optimal sensor fusion weights can be computed by directly optimizing the trace of the target state error covariance matrix. This is no longer true when the exogenous inputs are unknown to the tracking team - the case considered in this paper. In Section III-B, we show that the sensing quality of the robot is only implicitly related to the tracking performance of the team through a set of equations. In spite of this, we show that a viable heuristic approach - where the sensor fusion weights are computed based on the quality of the robots' sensors - works well both in simulation and experiment providing evidence that the weights computed in this manner improve the tracking performance of the robot team. The second stage of our approach generates a set of coordinates that maximizes the coverage of the robots over the targets' domain in the face of sensor quality deterioration, and realizes the new communication network topology in three-dimensional space. We link the coverage metric to the tracking performance metric by relating robot sensor parameters modeling spatial resolution to the measurement noise covariance matrix. The framework considered here accounts for sensor quality deterioration; (near) complete sensor failure can be incorporated into our framework by modeling it as a sensor with very high measurement noise covariance. In addition, we assume that sensor faults do not result in any bias in the sensor. In other words, the sensor measurement noise is always characterized by a zero-mean probability distribution. Further, we assume that robots can detect sensor faults and estimate their quality deterioration (*e.g.,* by fault detection [13] and degradation estimation [14], [15] techniques in the literature).

Stage one of our method uses *mixed integer semi-definite programs* (MISDPs) to formulate and solve the problem of constructing a communication graph topology and associated information fusion weights that improve the tracking performance of the team. We consider two MISDPs: *agent-centric configuration generation* (**ACCG**) and *team-centric configuration generation* (**TCCG**). The first (agent-centric) maximizes the trace of the inverse of *"Intermediate posteriori" target state estimation error covariance matrix* (**TISEECM**) associated with the robot which experienced the sensor quality degradation and the second (team-centric) minimizes the average of the trace of **TISEECM** over all the robots. Our

procedure in stage two solves an optimization problem to compute the coordinates of the robots that maximizes the coverage of the robots given measurement noise over the target's configuration space under communication constraints dictated by the graph topology computed in stage one.

Although resilience in multi-robot systems is a well studied research topic [16], the idea of resilience through reconfiguration during task execution is recent. Resilience in multi-robot networked systems has been extensively studied in the context of developing network connectivity conditions required to filter out the adverse influence of non-cooperating robots in the team [17]–[19]. Researchers have also developed algorithms that can perform distributed state estimation using sensor networks, when a fraction of the sensors in the the network are manipulated by an attacker according to a known attacker model [20]–[22]. [23] introduces a planning approach based on set function optimization to handle failures in a multi-robot team performing target tracking. All these works focus on developing *offline* resilient strategies for the multi-robot team under the assumption that a certain fraction of the robots in the team goes rogue or is under the influence of an attacker with a known model. In contrast, we do not assume any knowledge of the number of failures and propose an *online* solution to encode resilience in a multi-robot team performing multi-target tracking where the robots experience sensor degradation due to *unknown* external factors.

The recent survey paper by Prorok *et al.* [24] presents a taxonomy of the various approaches developed for building resilient multi-robot systems. According to the survey paper, the strategies presented in [17]–[19] are classified as *Pre-operative approaches*. Pre-operative approaches are defined as those strategies where the decision about the team's actions are made apriori [24]. On the other hand, since the strategies presented in this article identify the sensor quality deterioration and adapt without information on the origin or type of disturbance, they are categorized as *Intra-operative approaches*.

In recent years, researchers have employed the framework of *random finite sets* [25] for solving multi-target tracking problems [26], [27]. Under the random finite set framework, the evolution of target state densities are tracked instead of individual target states. In [28], we extended our framework developed in [10] to encode resilience in a team of robots performing multi-target tracking using the random finite set framework. In addition, our work in [29] described a resilience framework for a team of networked heterogeneous robots under complete sensor failures. In [29], the robots are tasked with monitoring an external process of interest using their sensors and sensors on their one-hop neighbors. To tackle this problem, we introduced a new notion of observability - *one-hop observability* - which quantified a robot's ability to estimate the state of an external process using its sensor measurements and the sensing information obtained from its one-hop neighbors.

This paper is an extension of the conference paper [12]. Compared to our previous work, we consider a more general model in this work where the robot trackers are unaware of the external inputs driving the targets. Compared to [12], we consider a more realistic tracking and coverage model where a robot's sensor quality affects both its tracking and coverage. By bringing in the concept of resilience by reconfiguration into multi-robot target tracking, we make the following contributions:

- We pose the resilient multi-robot multi-target tracking problem as a mixed integer semi-definite program (MISDP) in two different ways: *agent-centric configuration generation* (**ACCG**) and *team-centric configuration generation* (**TCCG**).
- We compare the solutions obtained from **ACCG** and **TCCG** against each other and with a greedy approach. The results show that with respect to the target tracking error **TCCG** outperforms both **ACCG** and greedy approach.
- We extend to the formation synthesis problem presented in [10] to produce robot positions that maximize the team's visual coverage given visual sensor deterioration over the target's configuration space while satisfying the constraints imposed by the communication graph topology.
- The approaches are validated using extensive simulations and their applicability to real world scenarios is demonstrated through a multi-robot experiment.

The rest of the paper is structured as follows. Section II outlines the notation used and gives the necessary background to understand the paper. Section III formally introduces the problem addressed in this paper. Section IV delineates the configuration generation and formation synthesis strategies developed here. Section V discusses the experimental results. Finally, Section VII summarizes and concludes the paper.

II. NOTATIONS AND PRELIMINARIES

Boldface lowercase and uppercase symbols represent vectors and matrices respectively. Small letter symbols indicate scalar quantities. Calligraphic symbols denote sets. Unless otherwise specified, the variables with a *bar* are associated with the robot trackers. Likewise, variables with a *tilde* correspond to the targets. $\mathbb{R}$ and $\mathbb{Z}^+$ denote the set of real numbers and positive integers respectively. For any positive integer $z \in \mathbb{Z}^+$, $[z]$ denotes the set $\{1, 2, \cdots, z\}$. The standard Euclidean 2-norm is denoted by $\|\cdot\|$. For a matrix $\mathbf{M} \in \mathbb{R}^{m_1 \times m_2}, m_1, m_2 \in \mathbb{Z}^+$, $\|\mathbf{M}\|_F$ denotes the Frobenius norm of the matrix. Additionally, $\mathrm{Tr}(\mathbf{M})$ denotes the trace of $\mathbf{M}$. The symbols $\mathbf{1}$ and $\mathbf{0}$ denote the vector or matrix of ones and zeros, respectively. The superscripts indicating the dimensions of matrices and vectors are omitted when they are clear from the context. We use $\mathbf{e}_i \in \mathbb{R}^{m_1}$ to denote the standard unit normal i^{th} basis vector of the Euclidean space. For a vector $\mathbf{v}$, $\mathrm{Diag}(\mathbf{v})$ gives a matrix with the elements of $\mathbf{v}$ along its diagonal. Conversely, $\mathrm{diag}(\mathbf{M})$ outputs the vector containing the diagonal elements of $\mathbf{M}$. Also, $\mathbf{M}^\top$ or $(\mathbf{M})^\top$, $\mathbf{M}^{-1}$ and $\mathbf{M}^\dagger$ represent its transpose, inverse, and Moore-Penrose pseudo-inverse, respectively. Note that we use $(\mathbf{M})_{i,j}$ to denote the (i, j) element of $\mathbf{M}$. We use $\mathbf{I}_{m_1}$ to denote the $m_1 \times m_1$ identity matrix. $[\mathbf{M}_1; \mathbf{M}_2; \cdots; \mathbf{M}_m]$ represents the vertically concatenated matrix build from the matrices $\{\mathbf{M}_1, \mathbf{M}_2, \cdots, \mathbf{M}_m\}$. $\mathbf{M}_1 \otimes \mathbf{M}_2$ results in a matrix obtained by taking the *Kronecker product* [30] between them. Similarly, $\mathbf{M}_1 \oplus \mathbf{M}_2$ yields the block diagonal matrix with $\mathbf{M}_1$ and $\mathbf{M}_2$ along its diagonal. $\mathcal{S}_+^{m_1}$ and $\mathcal{S}_{++}^{m_1}$ denotes the space of $m_1 \times m_1$ symmetric positive semi-definite matrices and

TABLE I: Notations

Symbol	Description
N	Number of robots
$\mathcal{D}$	Tracking domain
$\bar{i}$	Label of robot $i \in \{1, ..., N\}$
$\bar{x}^i$	x-coordinate of $\bar{i}$ w.r.t to global frame
$\bar{y}^i$	y-coordinate of $\bar{i}$ w.r.t to global frame
$\bar{z}^i$	z-coordinate of $\bar{i}$ w.r.t to global frame
$\bar{\theta}^i$	Orientation of $\bar{i}$ w.r.t to global frame
$\bar{\mathbf{x}}^i_\tau$	Pose $[x^i\ y^i\ \theta^i]^T$ of $\bar{i}$ at time τ
L	Number of targets
$\{\bar{\mathbf{x}}_{[N]}\}$	Tracker position set
$\tilde{j}$	Label of target $j \in \{1, ..., L\}$
$\{\tilde{\mathbf{x}}_{[L]}\}$	Target position set
$\mathcal{G}(k)$	Undirected robot communication graph at time step k
$\mathcal{V}$	Vertex set of $\mathbb{G}[k]$, $\{1, ..., N\}$ (robot indices)
$\mathcal{E}[k]$	Edge set of $\mathbb{G}[k]$ (pairs of robots that can communicate)
$\mathbf{A}[k]$	Weighted adjacency matrix associated with $\mathbb{G}(k)$
$\mathcal{N}_{(i)}[k]$	Neighbors of $\bar{i}$ (can communicate with $\bar{i}$) at time step k
$\tilde{\mathbf{x}}_k$	Targets' states vector
$\tilde{\mathbf{F}}_k$	Targets' state transition matrix
$\tilde{\mathbf{G}}_k$	Targets' input matrix
$\tilde{\mathbf{u}}_k$	Targets' exogenous input
$\bar{\mathbf{z}}^i_k$	Measurement vector obtained by $\bar{i}$
$\bar{\mathbf{H}}^i_k$	Measurement matrix of $\bar{i}$ at k
$\bar{\mathbf{q}}^i_k$	Sensing information vector
$\bar{\mathbf{\Omega}}^i_k$	Sensor information matrix
$\hat{\mathbf{x}}^i_{k\|k}$	Posteriori targets' state estimate
$\hat{\mathbf{P}}^i_k$	Targets' state estimation posteriori error covariance matrix
$\check{\mathbf{P}}^i_{k\|k}$	**TISEECM**
$\bar{\tilde{\mathbf{q}}}^i_k$	Information vector
$\bar{\tilde{\mathbf{\Omega}}}^i_k$	Information matrix
δ_c	Communication radius
δ_s	Safe inter-robot distance
$\delta^{\bar{i}}_f$	Field of view radius
$\mathcal{H}$	Coverage functional

positive definite matrices, respectively. A weighted undirected graph with non-negative edge weights $\mathcal{G}$ is defined using the triplet $\left(\mathcal{V}, \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}, \mathbf{A} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}_{\geq 0}\right)$, where $\mathbf{A}$ is the weighted adjacency matrix of the graph. Also, $\overline{\mathcal{E}} = (\mathcal{V} \times \mathcal{V}) \setminus \mathcal{E}$ denotes the edge complement of $\mathcal{G}$. Moreover, we use $\mathcal{G}[k]$ to "time stamp" a dynamic graph and denote the adjacency matrix of an unweighted graph using $\mathbf{A}_u$. The matrix $\mathbf{A}_u$ is defined as:

$$(\mathbf{A}_u)_{i,j} = \begin{cases} 1 & \text{if } (i,j) \in \mathcal{E} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

We define the neighbor set of vertex i according to $\mathcal{G}[k]$ as $\mathcal{N}_{(i)}[k] \triangleq \{j \mid (i,j) \in \mathcal{E}[k]\}$. A matrix $\mathbf{M}$ is doubly stochastic if its rows and columns sum to unity [30]. Mathematically, $\mathbf{M1} = \mathbf{1}$ and $\mathbf{1}^{\mathrm{T}}\mathbf{M} = \mathbf{1}^{\mathrm{T}}$. Due to Lemma 2.9 of [31], any doubly stochastic matrix has unity as one of its eigenvalues and all its other eigenvalues have a magnitude less than unity. Table I gives the list of the frequently used notation in the paper.

III. Problem Statement

We consider a team of $N \in \mathbb{Z}^+$ robots whose labels belong to $[N]$. The team is tasked with tracking a set of $L \in \mathbb{Z}^+$ moving targets labeled $\{1, 2, \cdots, L\}$ for a time period of T epochs. We refer to the robot team that tracks the moving targets as the *tracker team* and the robots as *trackers*. The robot with label $i \in [N]$ is indicated as $\bar{i}$. Similarly, the target with the label $j \in [L]$ is indicated as $\tilde{j}$. Let $\bar{\mathbf{x}}^i$ denote the triplet position vector $[\bar{x}^i, \bar{y}^i, \bar{z}^i]^\top \in \mathbb{R}^3$ of $\bar{i}$, and $\{\bar{\mathbf{x}}_{[N]}\}$ denote the set $\{\bar{\mathbf{x}}^i : i \in [N]\}$ of all tracker positions. In similar manner, $\tilde{\mathbf{x}}_j = [\tilde{x}^j, \tilde{y}^j, \tilde{z}^j]^\top \in \mathcal{D} \subset \mathbb{R}^3$ denotes position of target $\tilde{j}$ and $\{\tilde{\mathbf{x}}_{[L]}\}$ represent the set of all target positions. $\mathcal{D}$ represents a closed and bounded region where the targets are constrained to navigate. We assume that the trackers are equipped with localization capabilities which enable them to determine their own location with reasonable accuracy.

Since we consider the scenario where the tracker team performs distributed target tracking, trackers need to communicate among themselves. Let the dynamic undirected graph $\mathcal{G}[k] = (\mathcal{V}, \mathcal{E}[k], \mathbf{A}_u[k])$ model the communication network of the tracker team at the k^{th} time step ($k \in [T]$). Note that we use *time step*, *time*, and *epoch* interchangeably in this paper. The node set $\mathcal{V}$ is isomorphic to the tracker team label set $[N]$. An edge (i, j) is included in the edge set $\mathcal{E}[k]$ if $\bar{i}$ communicates with $\bar{j}$ at time k. Let $\delta_c \in \mathbb{R}_{>0}$ denote the size of each tracker's communication radius. In addition, we assume that the radius of the field of view associated with the i^{th} tracker is a non-decreasing function of its position's z-coordinate, denoted as $\delta^{\bar{i}}_f(z^i)$ or as $\delta^{\bar{i}}_f$. We assume that the trackers are equipped with sufficient computational capabilities enabling them to perform their part in the distributed target tracking task.

A. Target dynamics and measurement model

Let the dynamics of target $\tilde{i}$ at time step k be described by the following standard linear state space equation

$$\mathbf{x}^{\tilde{i}}_{k+1} = \mathbf{F}^{\tilde{i}}_k \mathbf{x}^{\tilde{i}}_k + \mathbf{G}^{\tilde{i}}_k \mathbf{u}^{\tilde{i}}_k + \mathbf{w}^{\tilde{i}}_k, \tag{2}$$

where $\mathbf{x}^{\tilde{i}}_k \in \mathbb{R}^{s^{\tilde{i}}_a}$ and $\mathbf{u}^{\tilde{i}}_k \in \mathbb{R}^{u^{\tilde{i}}_a}$ are the state and the input vectors of the target respectively. Here, $\mathbf{F}^{\tilde{i}}_k \in \mathbb{R}^{s^{\tilde{i}}_a \times s^{\tilde{i}}_a}$ and $\mathbf{G}^{\tilde{i}}_k \in \mathbb{R}^{s^{\tilde{i}}_a \times u^{\tilde{i}}_a}$ are the state transition matrix and input matrix of appropriate dimensions respectively. $\mathbf{w}^{\tilde{i}}_k \in \mathbb{R}^{s^{\tilde{i}}_a}$ is the zero-mean normally distributed random vector with the covariance matrix $\mathbf{Q}^{\tilde{i}}_k \in \mathcal{S}^{s^{\tilde{i}}_a}_{++}$. If we stack the state and input vectors of individual targets to form the stacked state vector $\tilde{\mathbf{x}} \in \mathbb{R}^{|\tilde{\mathbf{x}}|}$, $|\tilde{\mathbf{x}}| = \sum_{\tilde{i}=1}^{L} s^{\tilde{i}}_a$ and input vector $\tilde{\mathbf{u}} \in \mathbb{R}^{|\tilde{\mathbf{u}}|}$, $|\tilde{\mathbf{u}}| = \sum_{\tilde{i}=1}^{L} u^{\tilde{i}}_a$ respectively, then the dynamics of targets can be simplified as:

$$\tilde{\mathbf{x}}_{k+1} = \tilde{\mathbf{F}}_k \tilde{\mathbf{x}}_k + \tilde{\mathbf{G}}_k \tilde{\mathbf{u}}_k + \tilde{\mathbf{w}}_k, \tag{3}$$

where

$$\tilde{\mathbf{F}}_k = \mathbf{F}^{\tilde{1}}_k \oplus \mathbf{F}^{\tilde{2}}_k \oplus \cdots \oplus \mathbf{F}^{\tilde{L}}_k$$

and

$$\tilde{\mathbf{G}}_k = \mathbf{G}^{\tilde{1}}_k \oplus \mathbf{G}^{\tilde{2}}_k \oplus \cdots \oplus \mathbf{G}^{\tilde{L}}_k.$$

Also, $\tilde{\mathbf{w}}_k = \left[(\mathbf{w}^{\tilde{1}}_k); (\mathbf{w}^{\tilde{2}}_k); \cdots, (\mathbf{w}^{\tilde{L}}_k)\right]$ with the covariance matrix $\tilde{\mathbf{Q}}_k = \mathbf{Q}^{\tilde{1}}_k \oplus \mathbf{Q}^{\tilde{2}}_k \oplus \cdots \oplus \mathbf{Q}^{\tilde{L}}_k$. Each tracker can obtain measurements about the state of the targets present within its field of view. The measurement model associated with

tracker $\bar{i}$ obtaining measurements on the state target $\tilde{j}$ when $\tilde{j}$ is contained in the tracker's sensing region is expressed as

$$\mathbf{z}_k^{(\bar{i},\tilde{j})} = \mathbf{H}_k^{(\bar{i},\tilde{j})}\mathbf{x}_k^{\tilde{j}} + \mathbf{v}_k^{(\bar{i},\tilde{j})}, \tag{4}$$

where $\mathbf{z}_k^{(\bar{i},\tilde{j})} \in \mathbb{R}^{m_{\bar{i},\tilde{j}}}$ and $\mathbf{H}_k^{(\bar{i},\tilde{j})} \in \mathbb{R}^{s_a^{\tilde{j}} \times m_{\bar{i},\tilde{j}}}$ are the measurement vector and output matrix of tracker $\bar{i}$ while obtaining measurements about the state of target $\tilde{j}$, respectively. Also, $\mathbf{v}_k^{(\bar{i},\tilde{j})} \in \mathbb{R}^{m_{\bar{i},\tilde{j}}}$ is a zero-mean Gaussian sensing noise vector with a covariance matrix $\mathbf{R}_k^{(\bar{i},\tilde{j})} \in \mathcal{S}_{++}^{m_{\bar{i},\tilde{j}}}$ modeling the sensor noise characteristics of $\bar{i}$ while tracking the target $\tilde{j}$.

Let $\mathcal{T}^{\bar{i}}[k] = \{l_1, l_2, \cdots, l_{|\mathcal{T}|}\} \subseteq [L]$ denote the set containing the indices of the targets in present in the field of view of tracker $\bar{i}$ at the sampling time k. If we define $\bar{\mathbf{z}}_k^{\bar{i}} = \left[\mathbf{z}_k^{(\bar{i},l_1)}; \mathbf{z}_k^{(\bar{i},l_2)}; \cdots; \mathbf{z}_k^{(\bar{i},l_{|\mathcal{T}|})}\right]$, $\mathbf{H}_k^{\bar{i}} = \left[\mathbf{e}_{l_1} \otimes \mathbf{H}_k^{(\bar{i},l_1)}; \mathbf{e}_{l_2} \otimes \mathbf{H}_k^{(\bar{i},l_2)}; \cdots; \mathbf{e}_{l_{|\mathcal{T}|}} \otimes \mathbf{H}_k^{(\bar{i},l_{|\mathcal{T}|})}\right]$, then the measurement equation of tracker $\bar{i}$ can be expressed as

$$\bar{\mathbf{z}}_k^i = \bar{\mathbf{H}}_k^i \tilde{\mathbf{x}}_k + \bar{\mathbf{v}}_k^i, \tag{5}$$

where $\bar{\mathbf{v}}_k^i = [\mathbf{v}_k^{(\bar{i},l_1)}; \mathbf{v}_k^{(\bar{i},l_2)}; \cdots; \mathbf{v}_k^{(\bar{i},l_{|\mathcal{T}|})}]$ is the augmented zero-mean Gaussian random vector and $\bar{\mathbf{R}}_k^i = \mathbf{R}_k^{(\bar{i},1)} \oplus \mathbf{R}_k^{(\bar{i},2)} \oplus \cdots \oplus \mathbf{R}_k^{(\bar{i},L)}$ is its associated covariance matrix.

Throughout the paper we make the following assumption, which is a necessary condition for the existence of a distributed state estimator [32].

Assumption 1. *$(\bar{\mathbf{H}}_k, \tilde{\mathbf{F}}_k)$ is observable for all $k \in 0 \cup [T]$, where $\bar{\mathbf{H}}_k = [\bar{\mathbf{H}}_k^1; \bar{\mathbf{H}}_k^2; \cdots; \bar{\mathbf{H}}_k^N]$. In other words, the system is collectively observable.*

B. Decentralized and distributed Kalman filter for tracking unknown inputs ***DDKFU***

In this subsection, we describe the decentralized distributed Kalman filter computation employed by the trackers to collectively track the targets maneuvering in an environment according to the dynamics described in Section III-A. The main advantage of employing a decentralized distributed Kalman filter compared to a centralized one is the fact that a decentralized Kalman filter eliminates the need for central data fusion center (an entity that communicates with all the robots and collects their sensing information) to perform the data fusion [32]. The decentralized distributed Kalman filter is a well studied topic and various versions of the filter can be found in both in controls and robotics literature [3], [6], [33]. There is also a sizeable amount of literature on distributed Kalman filters that depend on centralized data fusion, which we do not consider in this paper. Interested readers can refer to [34]–[36] and the references therein. In this section of the paper, we primarily follow the formulation presented in [37], [38]. These works consider the problem of constructing Kalman filters when the exogenous inputs to the process are unknown.

Additionally, we make the following assumption, which is a necessary condition required for the existence of an unbiased state estimator when the exogenous inputs to the target dynamics are unknown [37], [38].

Assumption 2. *The rank of $\bar{\mathbf{H}}_k\tilde{\mathbf{G}}_k$ equals the rank of $\tilde{\mathbf{G}}_k$,*

We are now in a position to describe the distributed Kalman filtering algorithm used in this paper. The algorithm consists primarily of two steps: 1) *consensus update* and 2) *individual update*. These steps can be visualized as processes happening in a set of nested loops. The inner loop handles the operations associated with the consensus step, and the outer loop executes the operations related to the individual update step. When robot $\bar{i}$ receives measurement $\bar{\mathbf{z}}_k^i$ according to Equation 5, it computes two quantifies

$$\bar{\mathbf{q}}_k^i\{0\} = (\bar{\mathbf{H}}_k^i)^\top (\bar{\mathbf{R}}_k^i)^{-1} \bar{\mathbf{z}}_k^i$$
$$\bar{\mathbf{\Omega}}_k^i\{0\} = (\bar{\mathbf{H}}_k^i)^\top (\bar{\mathbf{R}}_k^i)^{-1} \bar{\mathbf{H}}_k^i.$$

We refer to these quantities as *sensing information vector* (SIV) and *sensor information matrix* (SIM) respectively. The nomenclature for these terms stems from the fact that they encode the information about a robot's sensors and sensor measurements obtained through its sensors. Once these quantities are computed, the robots in the tracker team exchange information among their neighbors and exponentially reach a consensus on the average of the quantities over the trackers. The robots exchange information according to the following equations:

$$\bar{\mathbf{q}}_k^i\{l+1\} = \sum_{j \in \mathcal{N}_{(i)}[k] \cup \{i\}} \left(\bar{\mathbf{A}}[k]\right)_{i,j} \bar{\mathbf{q}}_k^j\{l\} \tag{6}$$

$$\bar{\mathbf{\Omega}}_k^i\{l+1\} = \sum_{j \in \mathcal{N}_{(i)}[k] \cup \{i\}} \left(\bar{\mathbf{A}}[k]\right)_{i,j} \bar{\mathbf{\Omega}}_k^j\{l\}, \tag{7}$$

where $[\bar{\mathbf{A}}[k]]_{i,j}$ is the (i,j) entry of a doubly stochastic matrix $\bar{\mathbf{A}}[k]$ which has the same structure as the unweighted adjacency matrix $(\mathbf{A}_u[k])$ of $\mathcal{G}[k]$ except for the diagonal elements. Specifically, $\bar{\mathbf{A}}[k]$ is non-zero along its diagonal and its off-diagonal elements are non-zero if and only if the corresponding elements of $\mathbf{A}_u[k]$ are unity. In theory, the SIV and SIM of the robots converge to the respective averages only when l tends to infinity. However, it has been shown that this consensus protocol enjoys the additional property of an exponential rate of convergence [31]. As a result, a reasonable level of consensus on the SIV and SIM can be achieved by propagating Equation 6 and Equation 7 for a sufficient number of consensus steps η. Therefore, it is necessary that the consensus update can be executed at a much faster time scale than the system dynamics (Equation 2).

After η steps the consensus update, each tracker executes its individual update using the following the equations.

State Prediction:

$$\hat{\mathbf{x}}_{k|k-1}^i = \tilde{\mathbf{F}}_{k-1}\hat{\mathbf{x}}_{k-1}^i \tag{8}$$
$$\check{\mathbf{P}}_{k|k-1}^i = N(\tilde{\mathbf{F}}_{k-1}\bar{\mathbf{P}}_{k-1}^i(\tilde{\mathbf{F}}_{k-1})^\top + \mathbf{Q}_{k-1}). \tag{9}$$

Intermediate posteriori state error covariance:

$$\check{\mathbf{P}}_{k|k}^i \triangleq ((\check{\mathbf{P}}_{k|k-1}^i)^{-1} + \bar{\mathbf{\Omega}}_k^i)^{-1} \tag{10}$$

Input estimate:

$$\boldsymbol{\theta}_k = (\mathbf{I} - \bar{\boldsymbol{\Omega}}_k^i \check{\mathbf{P}}_{k|k}^i)\bar{\mathbf{q}}_k^i \tag{11}$$

$$\boldsymbol{\Theta}_k = \bar{\boldsymbol{\Omega}}_k^i - \bar{\boldsymbol{\Omega}}_k^i \check{\mathbf{P}}_{k|k}^i \bar{\boldsymbol{\Omega}}_k^i \tag{12}$$

$$\hat{\mathbf{u}}_{k-1}^i = (\tilde{\mathbf{G}}_{k-1}^\top \boldsymbol{\Theta}_k \tilde{\mathbf{G}}_{k-1})^{-1} (\tilde{\mathbf{G}}_{k-1}^\top \boldsymbol{\theta}_k - \tilde{\mathbf{G}}_{k-1}^\top \bar{\boldsymbol{\Omega}}_k^i \hat{\mathbf{x}}_{k|k-1}^i) \tag{13}$$

Local innovation or measurement update:

$$\hat{\mathbf{x}}_{k|k}^{i*} = \hat{\mathbf{x}}_{k|k-1}^i + \tilde{\mathbf{G}}_{k-1}\hat{\mathbf{u}}_{k-1}^i \tag{14}$$

$$\hat{\mathbf{x}}_{k|k}^i = \hat{\mathbf{x}}_{k|k}^{i*} + \check{\mathbf{P}}_{k|k-1}^i(\boldsymbol{\theta}_k - \boldsymbol{\Theta}_k \hat{\mathbf{x}}_{k|k}^{i*}) \tag{15}$$

$$\boldsymbol{\Upsilon}_k = \tilde{\mathbf{G}}_{k-1}(\tilde{\mathbf{G}}_{k-1}^\top \bar{\boldsymbol{\Omega}}_k^i \tilde{\mathbf{G}}_{k-1})^{-1}\tilde{\mathbf{G}}_{k-1}^\top \tag{16}$$

$$\check{\mathbf{P}}_{k|k}^{i*} = \check{\mathbf{P}}_{k|k-1}^i - \boldsymbol{\Upsilon}_k \bar{\boldsymbol{\Omega}}_k^i \check{\mathbf{P}}_{k|k-1}^i - \check{\mathbf{P}}_{k|k-1}^i \bar{\boldsymbol{\Omega}}_k^i \boldsymbol{\Upsilon}_k \tag{17}$$

$$\bar{\mathbf{P}}_k^i = \frac{1}{N}(\mathbf{I} - \check{\mathbf{P}}_{k|k-1}^i \bar{\boldsymbol{\Omega}}_k^i)(\check{\mathbf{P}}_{k|k}^{i*} - \check{\mathbf{P}}_{k|k-1}^i \bar{\boldsymbol{\Omega}}_k^i \boldsymbol{\Upsilon}_k), \tag{18}$$

where $\bar{\mathbf{q}}_k^i \triangleq \bar{\mathbf{q}}_k^i\{\eta\}$ and $\bar{\boldsymbol{\Omega}}_k^i \triangleq \bar{\boldsymbol{\Omega}}_k^i\{\eta\}$.

From the above equation it is clear that sensor measurement noise covariance matrices of the robots are implicitly related to the state estimate error covariance matrix. This relationship motivated us to investigate the idea of improving the tracking performance by optimally fusing the sensor information matrices.

For ease of readability, we move the details associated with *decentralized and distributed Kalman filter for tracking known inputs* (**DDKFK**) to Appendix C.

C. Tracking under sensor quality deterioration

In this subsection, we formally introduce the problem which is analyzed and solved in this paper. As mentioned in Section I, we consider the problem of mitigating the impact of sensor quality degradation on target tracking performance through appropriate reconfiguration of the tracker team. We will give a precise definition of what we mean by *tracker team reconfiguration* and *sensor quality deterioration* after we introduce some additional terminology and notation.

We term the tuple $(\mathcal{G}[k], \bar{\mathbf{A}}[k])$ as the *configuration* of the tracker team at the k^{th} time step and symbolize it by $\mathcal{C}[k]$. The matrix $\bar{\mathbf{A}}[k]$ is a doubly stochastic matrix whose elements are used to perform the information fusion computations outlined in Equation 6 and Equation 7 for the consensus step. During tracking operation for a T time steps, suppose n_f detrimental events occur independently to random trackers in the tracker team. Each event inflicts undesirable effects on the tracker's sensor, which results in sensor quality degradation. At a particular time k, we say that tracker i's sensor quality is deteriorated if the average trace of the measurement covariance matrices over the tracked targets at that instant has increased with respect to the previous instant. In other words, if $Tr(\bar{\mathbf{R}}_k^i)/|\mathcal{T}^i[k]| > Tr(\bar{\mathbf{R}}_{k-1}^i))/|\mathcal{T}^i[k-1]|$, then i's sensor quality deteriorated at time k. Recall that we assume the sensor is unbiased even after its quality deteriorates.

We consider a sequence $\mathcal{F} = [k_1, k_2, \cdots, k_p, \cdots, k_{n_f}]$, where $k_p \in [T]$ indicates the time step when the p^{th} sensor fault occurred. Consequently, we specify that $\mathcal{C}[k_p - 1]$ is the configuration of the tracker team before p^{th} detrimental event or sensor fault occurred. Now, we formally define the problems studied in this paper. The first problem (Problem 1) deals with reconfiguration of the tracker team such that target tracking performance is optimal in some reasonable sense. Complementing the first problem, the second problem addresses the issue of realizing the graph topology in 3-dimensional space while maximizing the tracker team's coverage over $\mathcal{D}$.

Problem 1. ***Configuration generation or reconfiguration:*** *Given that:*

- *tracker i experienced deterioration at some time k_p,*
- *$\bar{\mathbf{H}}_{k_p^+}^i$ is the sensor noise covariance matrix immediately after the sensor fault event, and*
- *$\mathcal{C}[k_p - 1]$ is the tracker configuration prior to the event,*

determine a new configuration $\mathcal{C}[k_p]$ such that

1) *$\mathcal{G}[k_p]$ is a connected graph,*
2) *$\|\mathbf{A}_u[k_p] - \mathbf{A}_u[k_p - 1]\|_F \leq 2e$, where $e \in \mathbb{Z}^+$ is the allowable number of edges to be modified in $\mathcal{G}[k_p - 1]$ to obtain $\mathcal{G}[k_p]$, and*
3) *tracking performance is optimized.*

Graph connectivity is an essential requirement for any distributed computation over a network and thus is enforced in Problem 1 [31]. The second condition enables the user to control the communication load on the generated configuration by tuning the parameter e. Finally, the third condition ensures good tracking performance. As noted in Section I, we take a heuristic approach to ensure good tracking performance for the team. In the forthcoming section, we describe the approach for solving these problems.

Problem 2. ***Formation synthesis:*** *Given a tracker team configuration $\mathcal{C}[k_p]$ and the tracker team's sensor model parameters, generate coordinates that best realizes the given configuration and maximizes tracker team's coverage over $\mathcal{D}$, subject to various constraints. These constraints ensures that the robot are at safe distance from each other and they are confined to $\mathcal{D}$. We defer the exact details of this problem to Section IV-B.*

IV. Methodology

In this section, we detail our strategies for solving Problem 1 and Problem 2. As indicated earlier in Section I, a base station monitors the activities of the tracker team. When a sensor fault occurs, the base station computes a new formation using the available information and directs the tracker team to reconfigure. Akin to our framework in [10], the base station in this paper also uses a two-step procedure to compute a new configuration and generate a set of robot coordinates that realize the computed configuration in three-dimensional space. As in our earlier work, we refer to these steps as *configuration generation* and *formation synthesis*. An illustration of the overall base station decision making process is outlined in Figure 2. We emphasize that configuration generation and formation synthesis steps are solutions to Problem 1 and Problem 2 respectively. The subsequent subsections delineate the steps in detail. Additionally, for ease of reference, we have included the agent-centric configuration generation and team-centric configuration generation formulations when the external

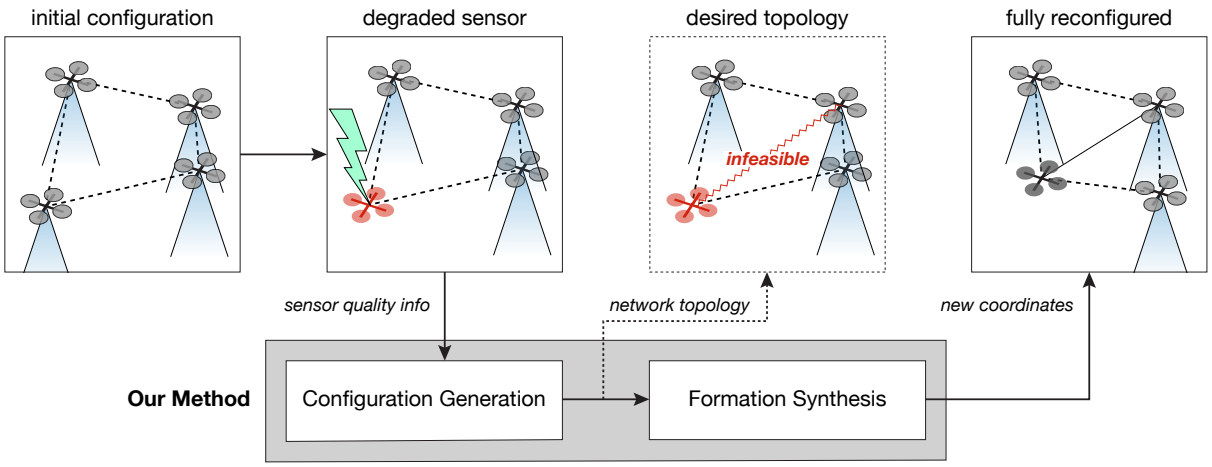


Fig. 2: Outline of our strategy. When a robot's sensor quality is affected, *configuration generation* modifies the communication graph. Then, *formation synthesis* generates the physical locations for the robots that support the desired graph topology.

inputs to targets are known to the trackers presented in [12]. We use the abbreviations **ACCGK** and **TCCGK** to denote our agent-centric configuration generation and team-centric configuration generation formulations when inputs are known, respectively.

A. Configuration generation

Our two strategies to solve Problem 1 are detailed in this subsection. As mentioned earlier, we refer to these strategies as *agent-centric configuration generation* (**ACCG**) and *team-centric configuration generation* (**TCCG**). Both strategies are based on solving a mixed integer semi-definite program. In the literature, many network topology design problems are addressed by posing them as MISDPs [39], [40]. Although it is well known that integer programming problems are NP-hard [41], reasonably sized problems can be solved using *branch and bound* [41] type algorithms by exploiting good heuristics. In the agent-centric configuration method, the idea is primarily to find a new configuration such that trace of information matrix (inverse of TSEECM) associated with the tracker in the tracker team that experienced a sensor quality deterioration is maximized. On the contrary, the team-centric method aims to minimize the average of the trace of TSEECMs associated with all the trackers. Since the optimization over the covariance matrices obtained after $\eta > 1$ steps consensus results in a mixed integer non-linear programming problem, we use the TSEECM obtained after one step consensus to formulate and solve the resulting MISDPs for both strategies. Before we dive into the details of the MISDP formulations, we state a theorem that serves as the basis for the connectivity inequality constraint in MISDPs [28].

Theorem 1. *If a graph containing self loops at every node is equipped with a weighted adjacency matrix* $\mathbf{A}$ *that is doubly stochastic, then any graph isomorphic to this graph with or without self loops is connected if and only if*

$$\frac{1}{n}\mathbf{1}\mathbf{1}^T + \mathbf{I} \succ \mathbf{A} \tag{19}$$

Proof. See Appendix A for the proof. □

*1) **ACCG**:* The following MISDP encodes our agent-centric configuration generation approach:

$$\underset{\substack{\mathbf{A}\in\mathcal{S}_+^N,\ \mu\in\mathbb{R}_{>0},\\ \mathbf{\Pi}\in\{0,1\}^{n\times n}}}{\text{maximize}} \quad (\mathbf{e}_i^N)^{\mathrm{T}}\mathbf{A}\begin{bmatrix}\mathrm{Tr}(\bar{\mathbf{\Omega}}^1_{k_p}[0])\\ \mathrm{Tr}(\bar{\mathbf{\Omega}}^2_{k_p}[0])\\ \vdots \\ \mathrm{Tr}(\bar{\mathbf{\Omega}}^n_{k_p}[0])\end{bmatrix} + \mathrm{Tr}((\check{\mathbf{P}}^i_{k|k-1})^{-1}) \tag{20}$$

$$\text{subject to} \quad \mathbf{A}\cdot\mathbf{1}^n = \mathbf{1}^n \tag{21}$$

$$\frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}} + (1-\mu)\mathbf{I} \succeq \mathbf{A}, \mu \ll 1 \tag{22}$$

$$diag(\mathbf{\Pi}) = \mathbf{1}^n \tag{23}$$

$$\mathbf{\Pi} = \mathbf{\Pi}^T \tag{24}$$

$$\mathbf{A} = \mathbf{A}^T \tag{25}$$

$$[\mathbf{A}]_{i,i} > 0\ \forall\ i \in [n] \tag{26}$$

$$[\mathbf{A}]_{i,j} \geq 0 \forall\ (i,j) \in [n]^2,\ i \neq j \tag{27}$$

$$[\mathbf{A}]_{i,j} \leq \mathbf{\Pi}_{i,j} \forall\ (i,j) \in [n]^2,\ i \neq j \tag{28}$$

$$\|\mathbf{\Pi} - \bar{\mathbf{A}}[k_p - 1]\|_F^2 \leq 2e. \tag{29}$$

The decision variables $\mathbf{A}$ and $\mathbf{\Pi}$ model the doubly stochastic matrix used for consensus protocol and the adjacency matrix of the generate configuration respectively. Constraint 21 and Constraint 25 to Constraint 28 ensure that $\mathbf{A}$ is a doubly stochastic matrix that is structurally equivalent to $\mathbf{\Pi}$. In light of Theorem 1, Constraint 22 enforces the generated configuration to possess a connected graph. Finally, Constraint 29 encodes the second condition in Problem 1 into the MISDP. If i represents the label of the robot that suffered sensor quality deterioration at k_p, then with some simple algebraic manipulation it is easy to see that Equation 20 is equal to $Tr(\mathbf{\Omega}^i_{k_p}(1))$ or $Tr((\bar{\mathbf{P}}^i_{k_p}(1))^{-1})$.

*2) **ACCGK**:* The following MISDP describes the agent-centric approach developed in [12].

$$\underset{\substack{\mathbf{A}\in\mathcal{S}_+^N,\ \mu\in\mathbb{R}_{>0},\\ \mathbf{\Pi}\in\{0,1\}^{n\times n}}}{\text{maximize}} \quad (\mathbf{e}_i^N)^{\mathrm{T}}\mathbf{A}\begin{bmatrix}\mathrm{Tr}(\bar{\bar{\mathbf{\Omega}}}^1_{k_p}[0])\\ \mathrm{Tr}(\bar{\bar{\mathbf{\Omega}}}^2_{k_p}[0])\\ \vdots\\ \mathrm{Tr}(\bar{\bar{\mathbf{\Omega}}}^n_{k_p}[0])\end{bmatrix} \tag{30}$$

subject to

Constraint 21 – Constraint 29.

*3) **TCCG**:* Consider the following MISDP formulation encoding the team-centric configuration generation strategy that minimizes $\frac{1}{n}\sum_i^n \mathrm{Trace}(\mathbf{P}^i_{k_p}(1))$, where $\mathbf{P}^i_{k_p}(1)) = (\mathbf{\Omega}^i_{k_p}(1))^{-1}$.

$$\underset{\substack{\mathbf{A}\in\mathcal{S}_+^n,\ \mu\in\mathbb{R}_{>0},\\ \mathbf{\Pi}\in\{0,1\}^{n\times n}\bar{\mathbf{P}},\bar{\mathbf{\Delta}}\in\mathcal{S}_+^{n\times s_a},\\ \mathbf{\Delta_1},\mathbf{\Delta_2},\cdots,\mathbf{\Delta_n}\in\mathcal{S}_+^{s_a}}}{\text{minimize}} \quad \frac{1}{n}\mathrm{Tr}(\bar{\mathbf{P}}) \tag{31}$$

$$\text{subject to} \quad \begin{bmatrix}\bar{\mathbf{P}} & \mathbf{I}\\ \mathbf{I} & \bar{\mathbf{\Delta}}\end{bmatrix} \succeq 0 \tag{32}$$

$$(\mathbf{A}\otimes\mathbf{I})\begin{bmatrix}\bar{\mathbf{\Omega}}^1_{k_p}[0]\\ \bar{\mathbf{\Omega}}^1_{k_p}[0]\\ \vdots\\ \bar{\mathbf{\Omega}}^1_{k_p}[0]\end{bmatrix} + \begin{bmatrix}(\check{\mathbf{P}}^{\mathbf{1}}_{\mathbf{k|k-1}})^{-1}\\ (\check{\mathbf{P}}^{\mathbf{2}}_{\mathbf{k|k-1}})^{-1}\\ \vdots\\ (\check{\mathbf{P}}^{\mathbf{N}}_{\mathbf{k|k-1}})^{-1}\end{bmatrix} = \begin{bmatrix}\mathbf{\Delta_1}\\ \mathbf{\Delta_2}\\ \vdots\\ \mathbf{\Delta_N}\end{bmatrix} \tag{33}$$

Constraint 21 – Constraint 29,

Where $\bar{\mathbf{\Delta}} \triangleq \mathbf{\Delta}_1 \oplus \mathbf{\Delta}_2 \cdots \oplus \mathbf{\Delta}_n$ and, $\mathbf{A}$and$\mathbf{I}$ are matrices of the equal size. Constraint 33 is essentially Equation 7 for $L = 1$ written compactly as a single equation for the whole tracker team. Therefore, $\mathbf{\Delta}_i$ should match the information matrix $\mathbf{\Omega}^i_{k_p}(1)$. The following lemma proves that minimizing Equation 31 minimizes $\frac{1}{n}\sum_i^n \mathrm{Trace}(\mathbf{P}^i_{k_p}(1)))$.

Lemma 1. *The objective $\frac{1}{n}Tr(\bar{\mathbf{P}})$ is an upper bound on $\frac{1}{n}\sum_i^n Trace(\mathbf{P}^i_{k_p}(1)))$*

Proof. See Appendix B for the proof. □

*4) **TCCGK**:* We describe the team-centric configuration generation approach developed in [12] using the following MISDP.

$$\underset{\substack{\mathbf{A}\in\mathcal{S}_+^n,\ \mu\in\mathbb{R}_{>0},\\ \mathbf{\Pi}\in\{0,1\}^{n\times n}\bar{\mathbf{P}},\bar{\mathbf{\Delta}}\in\mathcal{S}_+^{n\times s_a},\\ \mathbf{\Delta_1},\mathbf{\Delta_2},\cdots,\mathbf{\Delta_n}\in\mathcal{S}_+^{s_a}}}{\text{minimize}} \quad \frac{1}{n}\mathrm{Tr}(\bar{\mathbf{P}}) \tag{34}$$

$$\text{subject to} \quad \begin{bmatrix}\bar{\mathbf{P}} & \mathbf{I}\\ \mathbf{I} & \bar{\mathbf{\Delta}}\end{bmatrix} \succeq 0 \tag{35}$$

$$\mathbf{A}\otimes\mathbf{I}\begin{bmatrix}\bar{\bar{\mathbf{\Omega}}}^1_{k_p}[0]\\ \bar{\bar{\mathbf{\Omega}}}^1_{k_p}[0]\\ \vdots\\ \bar{\bar{\mathbf{\Omega}}}^1_{k_p}[0]\end{bmatrix} = \begin{bmatrix}\mathbf{\Delta_1}\\ \mathbf{\Delta_2}\\ \vdots\\ \mathbf{\Delta_N}\end{bmatrix} \tag{36}$$

Constraint 21 – Constraint 29.

B. Formation synthesis

We now describe a procedure to assign a physical location to each robot such that the team's coverage performance is maximized over $\mathcal{D}$. We also impose constraints so that connected robot pairs remain within communication distance δ_c of each other, and the distance between all robot pairs exceed δ_s to ensure that no two robots collide.

We calculate coverage performance $\mathcal{H}$ following [42] as

$$\mathcal{H} = \mathcal{H}_c - \mathcal{H}_o \tag{37}$$

$$\text{where} \quad \mathcal{H}_c = \sum_{i\in\mathcal{E}}\int_{\mathcal{V}_i} Surv_i(q)\phi(q)dq \tag{38}$$

$$\mathcal{H}_o = \sum_{i\in\mathcal{E}}\int_{\tilde{\mathcal{V}}_i} Surv_i(q)\phi(q)dq, \tag{39}$$

where $\mathcal{V}_i$ indicates the region of dominance of robot i, and $\tilde{\mathcal{V}}_i$ indicates the region covered by robot i where another robot in the team has superior sensing performance. Note that we set the region of dominance of a robot according to its field of view instead of the conic Voronoi diagram as in [42] for ease of computation. Additionally, we set the density function $\phi(q)$ as a constant to encode uniform importance of each point in the space.

$Surv_i(q)$ quantifies the surveillance quality of robot i for point q and is a function of its perspective quality $Pers_i$ and loss of resolution Res_i at that point:

$$Surv_i(q) = Pers_i(q)Res_i(q) \tag{40}$$

$$Pers_i(q) = \frac{\omega}{\omega-\lambda_i}\left(\frac{\bar{z}^i}{\|q-X_i\|} - \frac{\lambda_i}{\omega}\right) \tag{41}$$

$$Res_i(q) = \left(\frac{\lambda_i}{\omega}\right)^{\kappa} exp\left(-\frac{(\|q-X_i\|-R)^2}{2\sigma^2}\right), \tag{42}$$

where $\omega \triangleq \sqrt{\lambda_i^2 + (\bar{\delta_f^i})^2}$. Here X_i refers to the projected ground position of robot i as $[x^i\ y^i]^T$, and λ_i refers to the focal length of the robot's camera sensor. The parameters κ and σ model spatial resolution variability of the robot's camera sensor. R indicates the desired range of the sensor.

We set the parameter σ as $|Det(\bar{\mathbf{R}}^i_k - \bar{\mathbf{R}}^i_0)|$ to model an inverse relationship between sensor noise and spatial resolution. Thus, sensor deterioration through increasing $\bar{\mathbf{R}}^i_k$ does not only affect tracking performance step, but also indirectly affects the coverage performance.

This leads to the following constrained optimization problem for coverage performance:

$$\underset{\{\bar{\mathbf{x}}_{[N]}\}}{\text{maximize}} \quad \mathcal{H} \tag{43}$$

$$- \|Avg(\mathcal{X}_{[N]}) - Avg(\mathcal{X}_{[L]})\| \tag{44}$$

$$- \|Avg(\mathcal{X}_{[N]}) - Avg(\mathcal{X}'_{[N]})\| \tag{45}$$

$$\text{subject to} \quad \delta_s \le \|\mathcal{X}_i - \mathcal{X}_j\| \le \delta_c \qquad \forall\ (i,j)\in\mathcal{E} \tag{46}$$

$$\delta_s \le \|\mathcal{X}_i - \mathcal{X}_j\| \qquad \forall\ (i,j)\in\bar{\mathcal{E}} \tag{47}$$

$$B^{\min} \le \mathcal{X}_i \le B^{\max} \qquad \forall\ i\in V, \tag{48}$$

where $\mathcal{X}_i$ indicates the position of robot i as $[x^i\ y^i\ z^i]^T$. The terms $B^{\min}$, $B^{\max} \in \mathbb{R}^3$ are the minimum and maximum

extents of an axis-aligned bounding box, with the operator $\leq$ applied elementwise in Equation 48. We add the term in Equation 44 to the objective function to minimize the difference between the centroid of the trackers ($Avg(\mathcal{X}_{[N]})$) and the centroid of the targets ($Avg(\mathcal{X}_{[L]})$. We use the first tracker's target estimates for calculating the target centroid, but the convergence of the distributed DKF allows for any tracker's estimates to be used. Similarly, we add Equation 45 as a term in the objective function to minimize the difference between the new tracker positions and the previous tracker positions, which are denoted by $\mathcal{X}'_{[N]}$.

1) Simulated annealing: Simulated annealing (SA) is a well-known probabilistic technique for approximating the global optimum of an optimization problem with a large search space, such as our proposed coverage performance problem 43. Our description of SA assumes a minimization problem to follow the classic energy-based exposition. In searching for a global minimum, SA can escape poor local minima by occasionally randomly taking a step that *increases* the value of the objective function.

Algorithm 1: Simulated annealing ([43], [44])

Input: x_0 : initial guess
Output: Local optimal solution
Function *Simulated annealing*(x_0)
  $x \leftarrow x_0$ // set initial guess as the solution
  repeat
    $x' \leftarrow$ Propose(x) ;
    if $E(x') < E(x)$ **then**
      $x \leftarrow x'$;
    else
      $x \leftarrow x'$ w/ probability $\exp(-T(E(x') - E(x)))$
    $T \leftarrow$ Cooling(T)
  until *stopping criterion met*;
  return x;
Function *Propose(x)*
  sample $j \sim$ Uniform($[n]$), $d \sim$ Uniform($[3]$) ;
  sample $\delta \sim$ Uniform($[-\delta_{\max}, \delta_{\max}]$);
  $x' = x$;
  $x'_{j,d} \leftarrow x'_{j,d} + \delta$;
  return x';

We recap the algorithmic framework of SA in Algorithm 1. Propose(x) generates a new value x', an adjacent solution to x. $E(x)$ is the objective or *energy* function. T is the *temperature*, or the probability of taking a step that increases the minimization objective. Cooling(T) models a cooling schedule which decreases T over time. For additional details on SA refer to [43], [44].

In the implementation of Propose(x), we first randomly select a tracker whose position we will modify to create a new formation. We then randomly select a direction in which to translate the tracker's position. The distance for the translation is uniformly sampled between $(-\delta_s, \delta_s)$. We use exponential cooling (Cooling(T) $= \gamma T$ for $0 \ll \gamma < 1$), and a fixed number of steps as our stopping criterion. These were straightforward choices that yielded good solutions in our simulations and experiments.

We capture the objective and constraints in 43 in the energy function $E(x)$ through penalty functions:

$$E(x) = -\mathcal{H}_c + p_H(\mathcal{H}_o) + \sum_{(i,j)\in\bar{\mathcal{E}}} p_H(\delta_c - \|\mathcal{X}_i - \mathcal{X}_j\|_2) \tag{49}$$

$$+ \sum_{(i,j)\in\mathcal{E}} [p_H(\delta_s - \|\mathcal{X}_i - \mathcal{X}_j\|_2) + p_H(\|\mathcal{X}_i - \mathcal{X}_j\|_2 - \delta_c)] + \sum_{i\in V}\sum_{d\in[3]} p_H(\mathcal{X}_{i,d} - B_d^{\max}) + p_H(B_d^{\min} - X_{i,d}), \tag{50}$$

where p_H denotes a penalty function satisfying the property

$$\lim_{H\to\infty} p_H(y) = \begin{cases} \infty & : y > 0 \\ 0 & : y < 0. \end{cases} \tag{51}$$

The "hardness" parameter $H > 0$ increases over iterations, analogous to the decay of T. As in [10], we use the exponential penalty $p_H(y) = e^{Hy}$.

We emphasize that using penalty functions in $E(x)$ does not guarantee that the solution will satisfy the hard constraints 46–48. It is therefore necessary to perform a final feasibility check on the SA output after termination. If the SA output is not feasible according to 46–48, we restart the algorithm. In our experiments, we run SA for 20,000 steps. We choose γ such that T decays from 1 to 10^{-8}, and the growth constant for H such that H increases from 1 to 10^3. We let $\delta_{max} = \delta_s/10$. Note that tunings of T and H are sensitive to the overall scale of the distances $\mathbf{D}$, δ_s, δ_c involved in the problem (≈ 1 meter in our experiments).

V. SIMULATION RESULTS

In this section, we describe validation of our methodology using multiple simulation experiments of a multi-robot team tracking targets with unknown external inputs. The targets follow two-dimensional single integrator dynamics, given by the parameters

$$\mathbf{F}_k = \begin{bmatrix} 1 & 0 & dt & 0 \\ 0 & 1 & 0 & dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \ \mathbf{G}_k = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{52}$$

and input vector $\mathbf{u}_k = [v_k^x, v_k^y]$. Targets are initialized in a grid formation in the simulation space, a bounding box of $x \in [-30, 30]$, $y \in [-30, 30]$. At each time step, we apply input to each target such that it travels in a straight line to the edge of the bounding box before reversing direction.

For the distributed Kalman filter, we initiate the same $\bar{\mathbf{H}}_k^i$ and $\mathbf{R}_k^i$ for each robot in the tracker team. We used $L = 15$ for the consensus step. Parameters chosen for the configuration generation and formation synthesis problems were $e = 1$, $d_s = 10$, $d_{mc} = 10$, and $d_{sen}^i = 30 \forall\ i \in [n]$, with a bounding box of $x \in [-50, 50]$, $y \in [-50, 50]$, and $z \in [-10, 10]$.

To simulate deteriorating sensor quality for a robot *i*, we modified its covariance matrix $\mathbf{R}_k^i$ by adding a random positive semi-definite matrix of the same size. These random semi-positive definite matrices were generated by multiplying a matrix sampled from a uniform distribution between 0 and 1 by its transpose and then multiplying by some fixed scalar. The determinant of the modified $\mathbf{R}_k^i$ matrix was then added to the σ_i value of the robot when calculating its individual surveillance quality for the formation synthesis step.

We generated 10 different deterioration event sequences for robot teams of $n \in \{7, 10, 15, 20\}$ where a random robot was chosen at every f time step of the simulation to experience sensor deterioration. Each trial was initialized with the trackers positioned in a line and communicating according to the corresponding line graph. Figure 3 illustrates a single simulation trial for a team of 7 tracking 4 targets at different sensor deterioration events before and after reconfiguration. In the supplementary material, we also include a video of a team of 7 robots tracking 5 targets, 4 of which have pre-determined trajectories and 1 which follows user commands.

To quantify performance for both the **ACCG** and **TCCG** strategies, we plotted the average improvement of *maximum* $Tr(\bar{\mathbf{P}}_\mathbf{k}^\mathbf{i})$, *maximum estimation error*, and *average* $\mathcal{H}$ of the robot team over a *baseline* scenario where reconfiguration is not applied after failure. We also evaluate a *greedy* strategy in which a single edge is added connecting the robot whose sensing quality has deteriorated to the robot with lowest $Tr(\bar{\mathbf{P}}_\mathbf{k}^\mathbf{i})$ at the time of the deterioration event. Figure 4 shows the aggregate results of these simulations over all trials at different failure events. The term 'Delta' used in the charts in this figure refers to the difference between the evaluated scenario and the *baseline* for the given metric. Both **ACCG** and **TCCG** approaches show greater improvement in maximum estimation error and covariance over the baseline scenario than the *greedy* approach. However, the trade-off between minimizing estimation errors and covariance is a reduction in overall coverage of the tracking area. All approaches show decreasing $\mathcal{H}$ after each failure event as edges are added and robots are pulled closer together, increasing overlapping fields of view.

Additionally, Figure 5 compares the failure node estimation errors and $Tr(\bar{P}_k^i)$ of the **ACCG** and **TCCG** approaches to the previously developed **ACCGK** and **TCCGK** approaches. The difference in error and covariance between the scenario in which inputs are unknown and the scenario in which inputs are known is small, which indicates that both *agent-centric* and *team-centric* configuration generation strategies are appropriate for both tracking situations. The primary reason for this stems from the fact that the target dynamics described in Equation 52 satisfies the condition described in Assumption 2 which ensures the converges of Kalman filter.

VI. Experiments

To validate our framework in a real-time setting, we implement it on the Crazyswarm multi-quadrotor platform [45] using two quadrotors as targets and five as trackers. Aside from demonstrating that our methods are sufficiently fast and robust to use in real time, our hardware experiments test the accuracy of our distributed Kalman filter (DKF) on targets following real-world quadrotor dynamics. A photograph of our experiment is shown in Figure 7.

Due to complexity and resource constraints, we simulate the DKF on the base station PC instead of running it onboard the quadrotors' embedded processors. We emphasize that, while the implementation runs on the PC, it obeys the same communication constraints as it would if running onboard the tracker quadrotors. That is, while the base station facilitates communication between all quadrotors, only quadrotor pair that would directly communicate according to the weighted adjacency matrix $\mathbf{A}$ share information through the base station during the consensus step. Additionally, when targets move outside of a tracker's FOV, information of that target is not shared with the tracker by the base station.

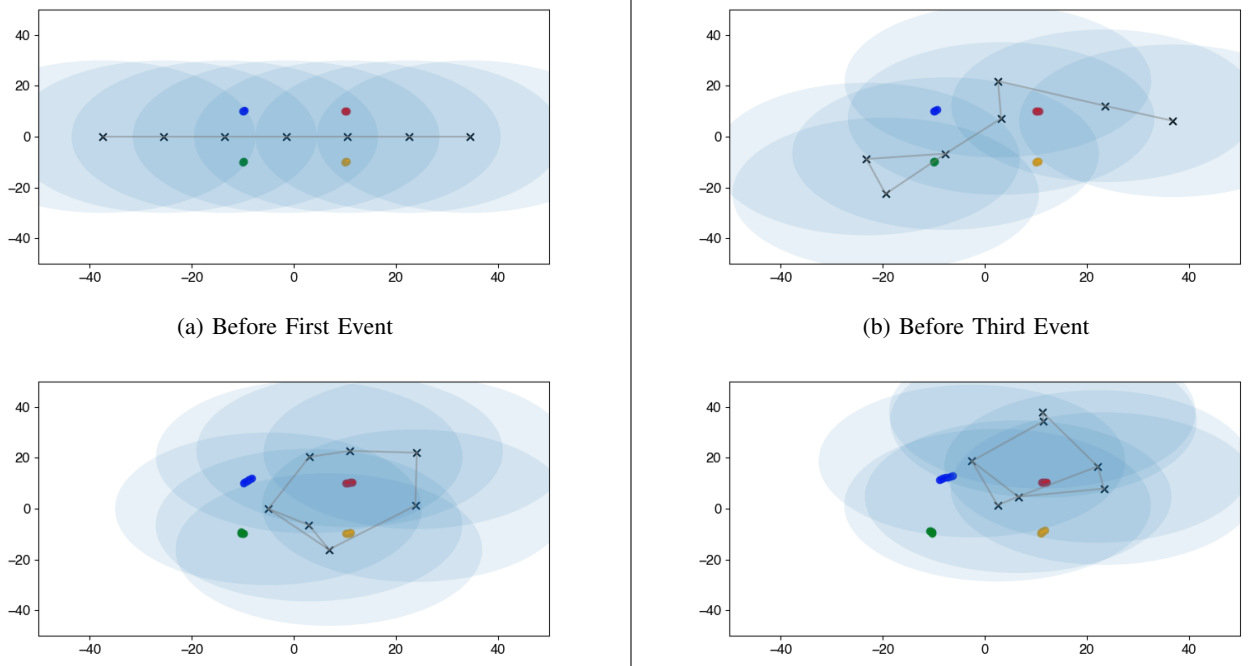
Our real-time implementation uses three concurrent processes $(P_{OPT}, P_{HW}, P_{DKF})$, devoted to optimization, hardware I/O, and Kalman filtering respectively; and four queues (Q_M, Q_E, Q_T, Q_C), devoted to target position measurements, DKF state estimates, tracker network topology commands, and tracker position commands respectively. The data flow between these elements is illustrated in Figure 6.

The optimization process P_{OPT} "drives" the experiment. At the beginning of its main loop, P_{OPT} fetches the most recent DKF state estimate from Q_E. It then simulates sensor degradation events and executes our centralized method to design a new network topology and formation using the same method described in Section V. We restricted the formation of the trackers to be within bounding box $x \in [-5, 5]$, $y \in [-5, 5]$, and $z \in [1.05, 2.25]$. Targets were restricted to the same x,y bounds, and restricted in height to $z \in [0, 1]$. P_{OPT} then places the new topology and the new positions of the formation on Q_T and Q_C respectively.

The hardware process P_{HW} continually sends low-level commands for the targets to move in concentric counter-rotating circles. Additionally, in each loop, P_{HW} checks Q_C for new tracker coordinates. If present, it sends high-level "go to and stop" commands to each tracker quadrotor. P_{HW} also receives position measurements from the motion capture system and places them on Q_M.

The distributed Kalman filter process P_{DKF} simulates our distributed Kalman filter algorithm (Section III-B) running onboard the trackers. In each loop, P_{DKF} checks Q_T for a new network topology and updates the DKF simulation state accordingly. P_{DKF} then reads the most recent motion capture measurement from Q_M. Each tracker updates its local state estimate using the measurements received, and exchanges information with its neighbors. Then, it uses the information exchange after consensus to update its estimate of the tracker state. Finally, it publishes the complete state estimate of each tracker on Q_E. As described in Section III-B, state estimates from all trackers will have converged, and a single estimate from any tracker can be used for planning.

Since our formation synthesis does not address the need for collision-free formation change trajectories, we employ buffered Voronoi collision avoidance [46] running onboard the quadrotors. The buffered Voronoi method is decentralized, and hence provides lower-quality solutions than centralized methods



(a) Before First Event

(b) Before Third Event

(c) After First Event

(d) After Third Event

Fig. 3: Screenshots of a simulation in which a seven quadrotor team tracks four targets below them. Each quadrotor's field of vision is colored in light blue. The quadrotor's themselves are depicted with an 'x'. The target trajectories are depicted with a '+'. The figures in the top row depict the formation of the quadrotors before the occurrence of a sensor deterioration event. The corresponding figures on the bottom portray the formation after 1) sensor deterioration is detected, 2) a new communication edge is chosen, and 3) the robots move to their new locations.

	max $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})$		max Estimation Error		$\mathcal{H}$	
	Average Delta	St. Dev Delta	Average Delta	St. Dev Delta	Average Delta	St. Dev Delta
ACCG	2.012	2.873	0.686	3.146	0.012	0.477
TCCG	2.369	3.738	0.855	4.121	0.103	0.631
greedy	1.884	2.222	0.621	2.810	-0.112	0.477

TABLE II: Summary statistics (mean and standard deviation) for the improvement in *maximum* $Tr(\mathbf{P})$, *maximum estimation error*, $\mathcal{H}$ over the *baseline* by strategy.

such as [47], but quality is reasonable when there are no environmental obstacles and the initial and final configurations are close. The distance term (45) in our formation synthesis objective therefore helps make collision avoidance easier.

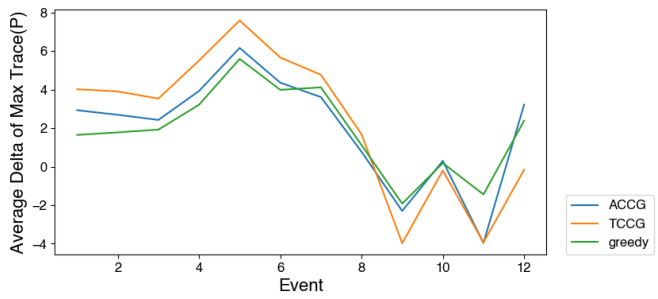
We monitored the estimation errors and sensing covariance of the trackers during the experiment. Three sensing failure events were simulated. After each failure event, the quadrotor team is able to recover and maintain good estimation of the target trajectories. Figure 9 depicts the results of the experiment. Figure 8 illustrates one of the true target trajectories and its trajectory as estimated by the tracker team.

Sensing failures result in a spike in covariance and estimation error. Spikes in estimation error also occur when the targets move out of the field of view of the one or more of the trackers. This type of event triggers a formation synthesis step without an optimization step. The sensing covariance at time points when targets are not in the field of view is not largely affected as a sensing failure has not occurred.

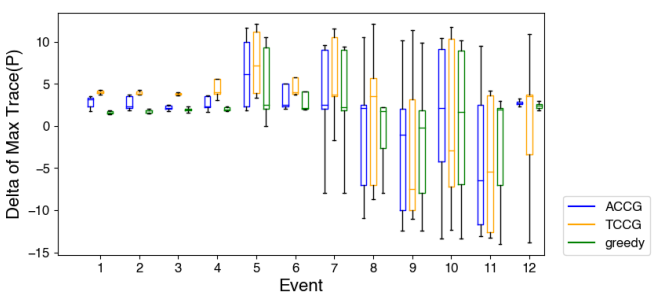
The combination of sensing failures and missed observations due to the target moving outside of the field of view result in instances of imperfect tracking, as shown in Figure 8. However, the team is able to trace the target trajectory with minimal errors over the duration of the experiment.

VII. Conclusion

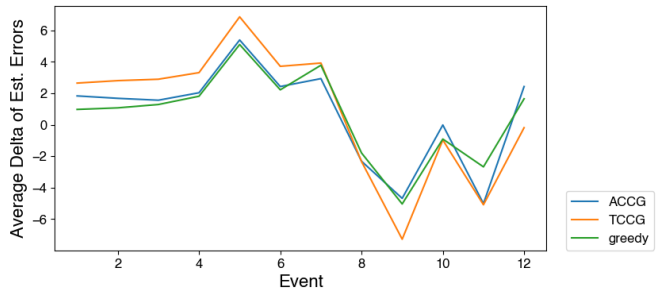
In this work, we proposed a novel strategy that allows a team of robots tasked with tracking a set of targets whose exogenous inputs are unknown, to reconfigure themselves in response to deterioration in the sensing quality in one member of the team. The reconfigured team mitigates the impact of the sensor quality deterioration on team's tracking and coverage performance. The strategies, which we term as *agent-centric* and *team-centric*, were validated in simulation and were compared to each other and a greedy strategy. Additionally, the strategies were compared to the agent-centric and team-centric strategies developed in our previous work [12], where any exogenous inputs driving the targets were known to the tracking robots. In terms of the estimation error in tracking



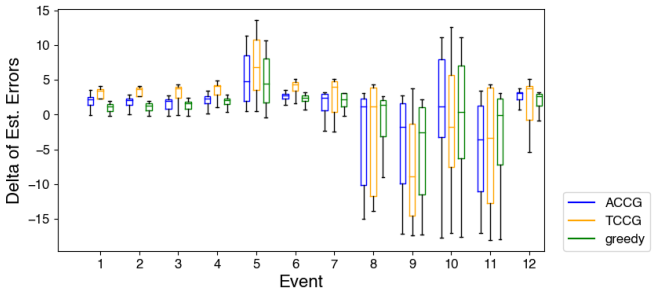
(a) Average of *max* $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})$ - *max* $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})_{baseline}$



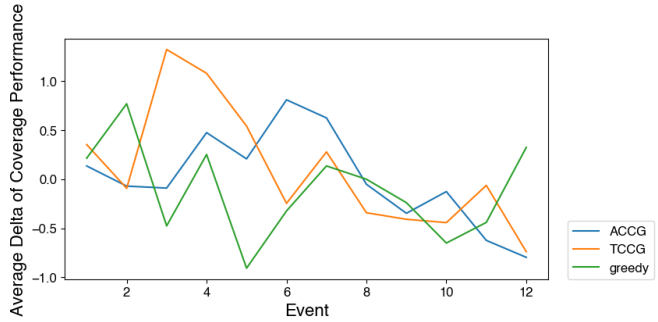
(b) Distribution of *max* $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})$ - *max* $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})_{baseline}$



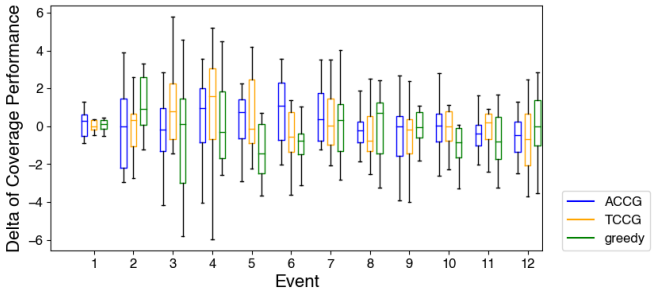
(c) Average of *max estimation error* - *baseline max estimation error*



(d) Distribution of *max estimation error* - *baseline max estimation error*



(e) Average of $\mathcal{H}$ - $\mathcal{H}_{baseline}$



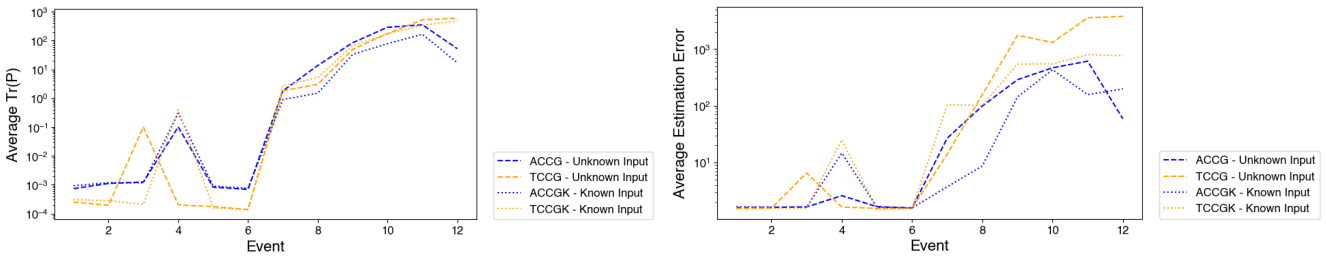
(f) Distribution of $\mathcal{H}$ - $\mathcal{H}_{baseline}$

Fig. 4: Comparison of the average improvement of *maximum* $Tr(\mathbf{P})$ and *maximum estimation error* over the *baseline* scenario between **ACCG** (blue), **TCCG** (orange), and the *greedy* (green) approaches over all simulations and team sizes. For each failure event, both the **ACCG** and **TCCG** strategies outperform the *greedy* strategy in both estimation error and covariance, though variance of their performance increases after several failure events, as shown by the length of the boxplots. All approaches show decreasing trend in $\mathcal{H}$ as edges are added after each failure event. Summarized statistics of these results are provided in Table II. The simulations are described in Section V.

the state of the targets, we infer that the team-centric approach outperforms both the agent-centric and greedy approaches. When comparing the agent-centric and greedy approaches we find that both gives similar results. This is due to the fact that akin to the greedy strategies the agent centric methods takes myopic decisions enforcing the affected robot to connect to a robot in the team with better sensing quality. Moreover, the results of the team- and agent-centric approaches proposed in this article are comparable to the ones proposed in our previous work [12]. Furthermore, we demonstrated that our strategy can be implemented to run in real time and applied to real-world scenarios through multi-robot experiments using quadrotors. Future work involves developing a decentralized version of our strategy which are scalable with the number of robots in the team.

APPENDIX A
PROOF OF THEOREM 1

Let $\mathbf{L} = \mathbf{I} - \mathbf{A}$, then since $\mathbf{A}$ is doubly stochastic $\mathbf{L}\mathbf{1}^n = \mathbf{0}^n$ and $\mathbf{L}^T\mathbf{1}^n = \mathbf{0}^n$. Also, as the spectrum of $\mathbf{A}$ is real and less than or equal to one in magnitude, the spectrum of $\mathbf{L}$ is real and less than or equal to zero. Now, from the above statement we conclude that $\mathbf{L}$ is a positive semi-definite matrix. Furthermore, note that $\mathbf{L}$ can be interpreted as the Laplacian of a weighted undirected graph $\mathcal{G}_L$ having the same topology of the graph associated with $\mathbf{A}$ except for self loops. Since the connectivity properties of an undirected graph does not depend on the existence of self loops, the original graph (the graph associated with $\mathbf{A}$) is connected if and only if $\mathcal{G}_L$ is connected. From [48, Proposition 1], we infer that $\mathcal{G}_L$ is connected if and only if $\mathbf{L} + \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}} \succ 0$. Therefore, substituting $\mathbf{L} = \mathbf{I} - \mathbf{A}$ in the equation yields $\frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}} + \mathbf{I} \succ \mathbf{A}$.



(a) Average $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})$ (b) Average *maximum estimation error*

Fig. 5: Average $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})$ and *maximum estimation error* between **ACCG**, **TCCG** and **ACCGK**, **TCCGK** strategies. The performance of the approaches where inputs are unknown are comparable to their counterpart approaches in which target inputs are known to the team.

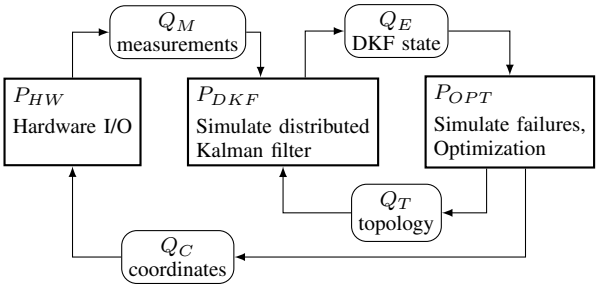


Fig. 6: Data flow between concurrent processes (sharp corners) and queues (round corners) in our real-time implementation.

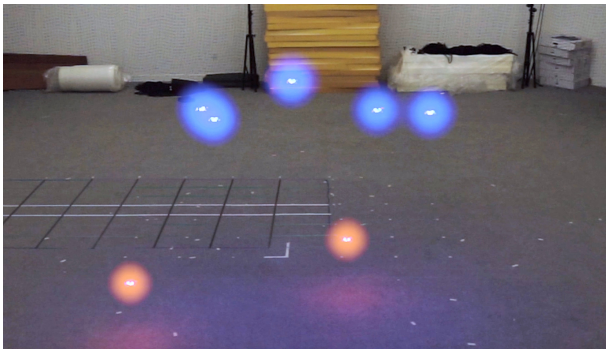


Fig. 7: Enhanced photograph of our real-time implementation controlling five trackers (blue) and two targets (orange). See Section VI for details.

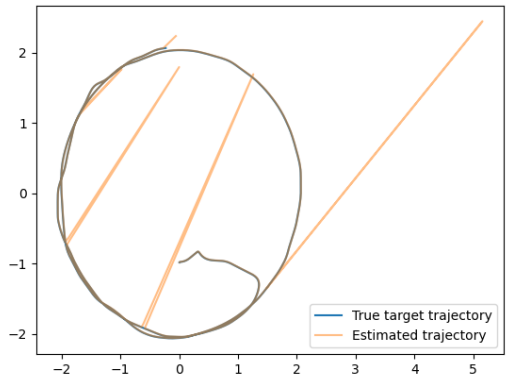


Fig. 8: Kalman filter-estimated trajectory and true trajectory of one of the target quadrotors. The combination of sensing failures and missed observations due to the target moving outside of the field of view result in instances of imperfect tracking, but the team is generally able to follow the target with minimal error.

Appendix B
Proof of Lemma 1

According to Schur complement lemma [49, Chapter 2], the *linear matrix inequality* (LMI) [49]

$$\begin{bmatrix} \mathbf{Q} & \mathbf{S} \\ \mathbf{S}^T & \mathbf{R} \end{bmatrix} \succeq 0 \tag{53}$$

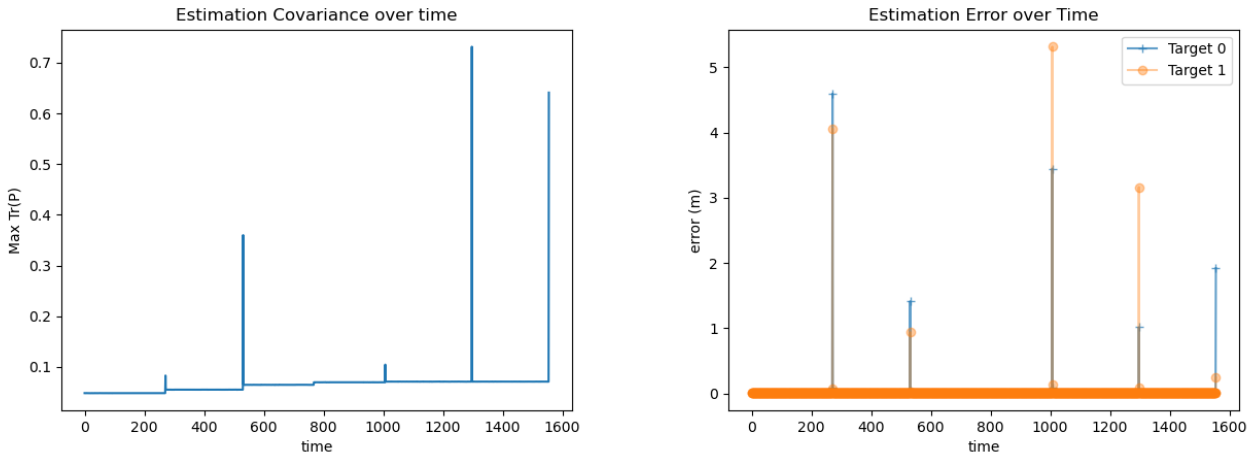
is equivalent to the conditions $\mathbf{R} \succeq 0$, $\mathbf{Q} - \mathbf{S}\mathbf{R}^{-1}\mathbf{S}^T \succeq 0$. Therefore, the LMI (32) is equivalent to $\bar{\mathbf{P}} - \bar{\mathbf{\Delta}}^{-1} \succeq 0$. Also, it is straightforward to see that Trace$(\bar{\mathbf{P}}) \geq Tr(\bar{\mathbf{\Delta}}^{-1})$. Since $\bar{\mathbf{\Delta}}$ is the block diagonal matrix containing the posterior information matrices of all tracking robots, $\bar{\mathbf{\Delta}}^{-1}$ is a block diagonal matrix with $\{\mathbf{P}^1_{k_p}(1), \mathbf{P}^2_{k_p}(1), \cdots, \mathbf{P}^n_{k_p}(1)\}$ along its diagonal. This is mathematically written as

$$\bar{\mathbf{\Delta}}^{-1} = \begin{bmatrix} \mathbf{P}^1_{k_p}(1) & 0 & \cdots & 0 \\ 0 & \mathbf{P}^2_{k_p}(1) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{P}^n_{k_p}(1). \end{bmatrix} \tag{54}$$

Therefore, $\frac{1}{n}Tr(\bar{\mathbf{P}}) \geq \frac{1}{n}Tr(\bar{\mathbf{\Delta}}^{-1})$ is equivalent to $\frac{1}{n}Tr(\bar{\mathbf{P}}) \geq \frac{1}{n}\sum_i^n Tr(\mathbf{P}^i_{k_p}(1))$.

Appendix C
Decentralized and Distributed Kalman filter for tracking known inputs (**DDKFK**)

For the sake of completeness, we describe the decentralized and distributed Kalman filter delineated in our previous work [12], which solves multi-target tracking problems when the external inputs to the targets are known to the trackers. For ease of readability, we move the details associated with **DDKFK** to appendix The formulation presented primarily follow the outlined in [6], [50] and the references therein. Similar to the formulation described in Section III-B, the distributed Kalman



(a) Average $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})$ of tracker team

(b) The *maximum estimation error* of tracker team for each target

Fig. 9: Average $Tr(\bar{\mathbf{P}}^{\mathbf{i}}_{\mathbf{k}})$ for the full state, and *maximum estimation error* for each target made by the five quadrotor tracker team in our experiment. Sensing failures occurred at times $t \in \{500, 1300, 1550\}$, resulting in a spike in covariance and estimation error. At times $t \in \{250, 1000\}$, the targets move out of the field of view of the one or more of the trackers. These events cause a spike in estimation error, and also triggers a formation synthesis step without an optimization step. The sensing covariance at these time points is not largely affected as a sensing failure has not occurred.

filter delineated here also contains the same two steps namely: 1) *individual update* and 2) *consensus update*. Contrary to formulation in the previous subsection, the robots preform the individual update before the consensus step. Upon receiving the measurement vector $\bar{\mathbf{z}}^i_k$, the robot $\bar{i}$ makes a local estimate about the states of the targets using the following standard Kalman filter equations.

Prediction:

$$\hat{\mathbf{x}}^i_{k|k-1} = \tilde{\mathbf{F}}_{k-1}\hat{\mathbf{x}}^i_{k-1} + \tilde{\mathbf{G}}_{k-1}\tilde{\mathbf{u}}_{k-1} \tag{55}$$

$$\bar{\mathbf{P}}^i_{k|k-1} = \tilde{\mathbf{F}}_{k-1}\bar{\mathbf{P}}^i_{k-1}(\tilde{\mathbf{F}}_{k-1})^\top + \mathbf{Q}_{k-1}. \tag{56}$$

Local innovation or measurement update:

$$\bar{\mathbf{K}}^i_k = \bar{\mathbf{P}}^i_{k|k-1}(\bar{\mathbf{H}}^i_k)^{\mathrm{T}}\left(\bar{\mathbf{H}}^i_k\bar{\mathbf{P}}^i_{k|k-1}(\bar{\mathbf{H}}^i_k)^{\mathrm{T}} + \bar{\mathbf{H}}^i_k\right)^{-1} \tag{57}$$

$$\hat{\mathbf{x}}^i_{k,0} = \hat{\mathbf{x}}^i_{k|k-1} + \bar{\mathbf{K}}^i_k(\bar{\mathbf{z}}^i_k - \bar{\mathbf{H}}^i_k\hat{\mathbf{x}}^i_{k|k-1}) \tag{58}$$

$$\bar{\mathbf{P}}^i_{k,0} = \bar{\mathbf{P}}^i_{k|k-1} - \bar{\mathbf{K}}^i_k\bar{\mathbf{H}}^i_k\bar{\mathbf{P}}^i_{k|k-1}. \tag{59}$$

In the consensus step, the each tracker fuses its local targets' states estimate by communication with its local neighbors using a consensus protocol. Each tracker computes the *information vector* (IV) $\bar{\bar{\mathbf{q}}}^i_k\{0\} = (\bar{\mathbf{P}}^i_{k,0})^{-1}\hat{\mathbf{x}}^i_{k,0}$ and the corresponding information matrix (IM) $\bar{\bar{\mathbf{\Omega}}}^i_k\{0\} = (\bar{\mathbf{P}}^i_{k,0})^{-1}$ prior to initiating the consensus protocol. After computing these quantities, the tracker instigates a consensus protocol similar to Equation 6 and Equation 7 to arrive at a refined state estimate of the tracked targets. The trackers interchange information as follows:

$$\bar{\bar{\mathbf{q}}}^i_k\{l+1\} = \sum_{j\in\mathcal{N}_{(i)}[k]\cup i}[\bar{\mathbf{A}}[k]]_{i,j}\bar{\bar{\mathbf{q}}}^j_k\{l\} \tag{60}$$

$$\bar{\bar{\mathbf{\Omega}}}^i_k\{l+1\} = \sum_{j\in\mathcal{N}_{(i)}[k]\cup i}[\bar{\mathbf{A}}[k]]_{i,j}\bar{\bar{\mathbf{\Omega}}}^j_k\{l\}. \tag{61}$$

After performing η epochs in the consensus protocol, the posteriori estimates of the targets' state vector $\hat{\mathbf{x}}^i_{k|k}$ and the posteriori estimation error covariance matrix $\bar{\mathbf{P}}^i_k$ of $\bar{i}$ can be computed as $\hat{\mathbf{x}}^i_{k|k} = (\bar{\bar{\mathbf{\Omega}}}^i_k\{\eta\})^{-1}\bar{\bar{\mathbf{q}}}^i_k\{\eta\}$ and $\bar{\mathbf{P}}^i_k = (\bar{\bar{\mathbf{\Omega}}}^i_k\{\eta\})^{-1}$. Observe that the distributed Kalman filter based formulation presented here comprise of fewer computations compared to the ones described in Section III-B. Also, the information vector and matrix associated with the standard Kalman filter contains complete information about the state of the targets. In other words, given a information vector and matrix, the corresponding the targets' state estimate and estimation error covariance matrix can be uniquely determined. On the contrary, the SIV and SIM introduced in Section III-B alone is insufficient to calculate the corresponding state estimate and estimation error covariance matrix.

References

[1] C. Robin and S. Lacroix, "Multi-robot target detection and tracking: taxonomy and survey," *Autonomous Robots*, vol. 40, no. 4, p. 729–760, 2016.

[2] K. Hausman *et al.*, "Cooperative multi-robot control for target tracking with onboard sensing," *The International Journal of Robotics Research*, vol. 34, no. 13, pp. 1660–1677, 2015.

[3] R. Olfati-Saber and P. Jalalkamali, "Collaborative target tracking using distributed kalman filtering on mobile sensor networks," in *Proceedings of the 2011 American Control Conference*. IEEE, 2011, pp. 1100–1105.

[4] R. K. Williams and G. S. Sukhatme, "Observability in topology-constrained multi-robot target tracking," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 1795–1801.

[5] P. Dames, P. Tokekar, and V. Kumar, "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1540–1553, 2017.

[6] Q. Liu *et al.*, "On kalman-consensus filtering with random link failures over sensor networks," *IEEE Transactions on Automatic Control*, vol. 63, no. 8, pp. 2701–2708, 2017.

[7] F. Pasqualetti, S. Zampieri, and F. Bullo, "Controllability metrics, limitations and algorithms for complex networks," in *2014 American Control Conference*, June 2014, pp. 3287–3292.

[8] D. Leitold, Á. Vathy-Fogarassy, and J. Abonyi, "Controllability and observability in complex networks–the effect of connection types," *Scientific reports*, vol. 7, no. 1, p. 151, 2017.

[9] R. K. Ramachandran and S. Berman, "The effect of communication topology on scalar field estimation by large networks with partially accessible measurements," in *2017 American Control Conference (ACC)*, May 2017, pp. 3886–3893.

[10] R. K. Ramachandran, J. A. Preiss, and G. S. Sukhatme, "Resilience by reconfiguration: Exploiting heterogeneity in robot teams," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2019, pp. 6518–6525.

[11] C. Yang, L. Kaplan, and E. Blasch, "Performance measures of covariance and information matrices in resource management for target state estimation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 3, pp. 2594–2613, 2012.

[12] R. K. Ramachandran, N. Fronda, and G. S. Sukhatme, "Resilience in multi-robot target tracking through reconfiguration," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4551–4557.

[13] A. B. Sharma, L. Golubchik, and R. Govindan, "Sensor faults: Detection methods and prevalence in real-world datasets," *ACM Transactions on Sensor Networks (TOSN)*, vol. 6, no. 3, p. 23, 2010.

[14] S. Arosh *et al.*, "Fitness function based sensor degradation estimation using h infinity filter," *Procedia Computer Science*, vol. 58, pp. 172–177, 2015.

[15] L. Jiang *et al.*, "Sensor degradation detection in linear systems," in *Engineering Asset Management*. Springer, 2006, pp. 1252–1260.

[16] L. Guerrero-Bonilla, A. Prorok, and V. Kumar, "Formations for resilient robot teams," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 841–848, April 2017.

[17] H. J. LeBlanc *et al.*, "Resilient asymptotic consensus in robust networks," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 4, pp. 766–781, April 2013.

[18] H. Zhang and S. Sundaram, "Robustness of information diffusion algorithms to locally bounded adversaries," in *2012 American Control Conference (ACC)*. IEEE, 2012, pp. 5855–5861.

[19] H. Zhang, E. Fata, and S. Sundaram, "A notion of robustness in complex networks," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 3, pp. 310–320, Sep. 2015.

[20] Y. Chen, S. Kar, and J. M. F. Moura, "Resilient distributed estimation: Sensor attacks," *IEEE Transactions on Automatic Control*, vol. 64, no. 9, pp. 3772–3779, 2019.

[21] Y. Chen, S. Kar, and J. M. F. Moura, "Resilient distributed parameter estimation with heterogeneous data," *IEEE Transactions on Signal Processing*, vol. 67, no. 19, pp. 4918–4933, 2019.

[22] S. Xiao *et al.*, "Distributed resilient estimator design for positive systems under topological attacks," *IEEE Transactions on Cybernetics*, pp. 1–11, 2020.

[23] L. Zhou *et al.*, "Resilient active target tracking with multiple robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 129–136, 2018.

[24] A. Prorok *et al.*, "Beyond robustness: A taxonomy of approaches towards resilient multi-robot systems," 2021.

[25] R. P. Mahler, *Statistical multisource-multitarget information fusion*. Artech House, Inc., 2007, vol. 685.

[26] R. P. S. Mahler, "Multitarget bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, Oct 2003.

[27] O. Erdinc, P. Willett, and Y. Bar-Shalom, "Probability hypothesis density filter for multitarget multisensor tracking," in *2005 7th International Conference on Information Fusion*, vol. 1, 2005, pp. 8 pp.–.

[28] R. K. Ramachandran, N. Fronda, and G. Sukhatme, "Resilience in multi-robot multi-target tracking with unknown number of targets through reconfiguration," *IEEE Transactions on Control of Network Systems*, pp. 1–1, 2021.

[29] R. K. Ramachandran *et al.*, "Resilient monitoring in heterogeneous multi-robot systems through network reconfiguration," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 126–138, 2022.

[30] R. A. Horn and C. R. Johnson, Eds., *Matrix Analysis*. New York, NY, USA: Cambridge University Press, 1986.

[31] F. Bullo, *Lectures on Network Systems*, 1st ed. Kindle Direct Publishing, 2019, with contributions by J. Cortes, F. Dorfler, and S. Martinez. [Online]. Available: http://motion.me.ucsb.edu/book-lns

[32] R. Olfati-Saber, "Distributed kalman filtering for sensor networks," in *2007 46th IEEE Conference on Decision and Control*, 2007, pp. 5492–5498.

[33] F. Morbidi and G. L. Mariottini, "On active target tracking and cooperative localization for multiple aerial vehicles," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 2229–2234.

[34] W. Li, F. Yang, and G. Wei, "A novel observability gramian-based fast covariance intersection rule," *IEEE Signal Processing Letters*, vol. 25, no. 10, pp. 1570–1574, 2018.

[35] D. Shi, T. Chen, and M. Darouach, "Event-based state estimation of linear dynamic systems with unknown exogenous inputs," *Automatica*, vol. 69, pp. 275–288, 2016.

[36] D. Yu *et al.*, "Distributed covariance intersection fusion estimation with delayed measurements and unknown inputs," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–9, 2019.

[37] A. Esna Ashari, A. Y. Kibangou, and F. Garin, "Distributed input and state estimation for linear discrete-time systems," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 2012, pp. 782–787.

[38] S. Gillijns and B. D. Moor, "Unbiased minimum-variance input and state estimation for linear discrete-time systems," *Automatica*, vol. 43, no. 1, pp. 111 – 116, 2007.

[39] M. Rafiee and A. M. Bayen, "Optimal network topology design in multi-agent systems for efficient average consensus," in *49th IEEE Conference on Decision and Control*. IEEE, 2010, pp. 3877–3883.

[40] D. Grob and O. Stursberg, "Optimized distributed control and network topology design for interconnected systems," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*, 2011, pp. 8112–8117.

[41] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, 5th ed. Springer Publishing Company, Incorporated, 2012.

[42] R. Funada *et al.*, "Visual coverage control for teams of quadcopters via control barrier functions," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3010–3016.

[43] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *science*, vol. 220, no. 4598, pp. 671–680, 1983.

[44] D. Henderson, S. H. Jacobson, and A. W. Johnson, "The theory and practice of simulated annealing," in *Handbook of metaheuristics*. Springer, 2003, pp. 287–319.

[45] J. A. Preiss* *et al.*, "Crazyswarm: A large nano-quadcopter swarm," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3299–3304.

[46] D. Zhou *et al.*, "Fast, on-line collision avoidance for dynamic vehicles using buffered voronoi cells," *IEEE Robotics Autom. Lett.*, vol. 2, no. 2, pp. 1047–1054, 2017.

[47] W. Hönig *et al.*, "Trajectory planning for quadrotor swarms," *IEEE Trans. Robotics*, vol. 34, no. 4, pp. 856–869, 2018.

[48] M. Sundin *et al.*, "A connectedness constraint for learning sparse graphs," in *2017 25th European Signal Processing Conference*. IEEE, 2017, pp. 151–155.

[49] S. Boyd *et al.*, *Linear matrix inequalities in system and control theory*. Siam, 1994, vol. 15.

[50] G. Battistelli *et al.*, "Consensus-based algorithms for distributed filtering," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 794–799.