Multirobot Fully Distributed Active Joint Localization and Target Tracking

Shaoshu Su, Pengxiang Zhu, and Wei Ren, *Fellow, IEEE*

***Abstract*— In this article, we study the problem of multirobot active joint localization and target tracking (AJLATT), where a team of robots mounted with sensors of limited field of view *actively* estimate their own and the target's states cooperatively. Each robot designs its motion strategy to gain better estimation performance while avoiding collisions by using only the information from itself and its one-hop communicating neighbors. By leveraging the framework of joint localization and target tracking (JLATT) presented in our previous work, we propose two fully distributed algorithms that help each robot design motion strategies to achieve better localization and target tracking performance. These two algorithms are designed from, respectively, the control and optimization perspectives. The control-based algorithm is designed by incorporating the estimated target's and robots' states and their uncertainties as well as collision avoidance in the control policy. The optimization-based algorithm minimizes an objective function involving both the target's and robots' estimation uncertainties and a potential function that helps each robot avoid collision and maintain communication connectivity when the robot is planning its motion. Monte Carlo simulations demonstrate our algorithms' feasibility to solve the AJLATT problem, and performance comparison between these two algorithms is given.**

***Index Terms*— Distributed estimation, motion control, multi-robot systems, target tracking.**

I. Introduction

AUTONOMOUS multirobot systems equipped with sensors have attracted more and more attention in recent years due to their wide applications in search and rescue, region monitoring, area surveillance, and so on. Multirobot systems have many appealing properties. In this work, we focus on their usage for estimation. Compared with one single robot, a multirobot system is able to obtain better self-localization and target tracking performance by utilizing abundant robot-to-robot and robot-to-target measurements as well as information exchanged across the team. That property plays an essential role in the applications of the multirobot system, especially in the indoor scenario where it is hard to receive the GPS signal. Besides, compared with a static sensor network, a team of autonomous mobile sensors is more adaptive since it is much easier and more effective to be deployed in an unknown environment.

There are many results on estimating the states of robots or a target or both. Some works focus on multirobot cooperative localization [1], [2], [3], [4], [5], [6], [7], [8], [9], where a team of robots can improve their self-localization accuracy by utilizing the mutual relative measurements and interacting with their team members. Some other works focus on target tracking [10], [11], [12], [13], [14], [15], where a static or moving sensor network seeks to estimate the target's state. Here it is usually implicitly assumed that the sensors' states are known (no self-localization needed). There also appear some later works that achieve the cooperative localization and target tracking simultaneously, both in a centralized [16], [17], [18] or fully distributed manner [19], [20].[1] However, all the above works [1], [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20] have a limitation that the sensors or robots are either static or move without a properly designed motion strategy to actively localize themselves or track the target.

Some algorithms have been proposed to solve the problem of active target tracking from the control or optimization perspective. In the controls field, there are numerous results on distributed tracking or leader-follower tracking (see, e.g., [21]). However, in these results, both the robots' and target's states are usually assumed to be known, and the emphasis is on designing distributed controllers. In [22], the target's state is assumed to be an inaccurate variable, and a gradient-based decentralized motion control strategy is developed to drive a team of robots to estimate and actively track the target's state. Nevertheless, an all-to-all communication network is required in this work. In [23], an information-driven flocking algorithm is proposed to achieve the distributed target state estimation. However, the estimator in [23] relies on a restrictive assumption that the target is jointly observed by each robot and its neighbors. As for the optimization-based approaches, [24] presents an approach for a robot team to find the local optimal action to maximize the knowledge about the targets by using a greedy search. An algorithm is proposed in [25] to analytically obtain the next global optimal sensing locations for the robots. A nonmyopic search-based algorithm is proposed in [26] to drive one robot to actively track the

Manuscript received 6 March 2022; revised 4 October 2022; accepted 26 December 2022. Date of publication 6 April 2023; date of current version 22 June 2023. This work was supported by National Science Foundation under Grant CMMI-2027139. Recommended by Associate Editor J. B. Hoagg. *(Corresponding author: Wei Ren.)*

The authors are with the Department of Electrical and Computer Engineering, University of California, Riverside, CA 92521 USA (e-mail: shaoshu.su@email.ucr.edu; pzhu008@ucr.edu; ren@ee.ucr.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCST.2023.3240525.

Digital Object Identifier 10.1109/TCST.2023.3240525

[1]We use the term *fully distributed* to describe an algorithm that uses only a robot's own and one-hop neighbors' information without the requirement for global parameters, multihop information transmission, or multiple communication iterations on certain quantities for iterative consensus-type calculations per time instant.

target by minimizing the logarithm of the determinant of the target covariance obtained by the Kalman filter. An algorithm is proposed in [27] to maximize the mutual information between the robots' measurements and their current belief of the target position using an experimental time-of-flight range sensor model for measurement and a Bayesian filter for estimation. Nevertheless, all the above optimization-based works are implemented in a centralized manner. In [28], the approach in [26] is extended in a decentralized way, where each robot in the team uses the coordinate descent method to plan its motion sequentially. As a result, multihop information transmission would be required in this approach. Besides, in all of the works above, the robots' states are assumed to be accurately known, which is an unrealistic assumption in practice.

There are some works addressing the active joint localization and target tracking (AJLATT) problem, where motion strategies are designed to improve the target tracking and/or robot self-localization performance. Considering limited sensing capabilities, Hausman et al. [29] developed an algorithm which coordinates robots' motions and switches the sensing topology to minimize the uncertainty of the target state estimate. Although this work estimates the robots' states together with the target's state, it does not make improving robot self-localization performance an objective in its optimization problem. In [30], a gradient-based control policy is designed to minimize the uncertainty of both the robots' and the target's states that are estimated by the Kalman–Bucy filter. While these two works consider jointly estimating both robots' states and target's state, their approaches are centralized. Although the centralized approaches have the advantage of the capability to obtain the optimal estimation and planning performance, they also cause a heavy burden in computation and communication. In [31], a distributed method is proposed by using a Bayesian filter for estimation and a gradient-based algorithm for control design. However, the method requires multihop information transmission for estimation and multihop information transmission or multiple communication iterations for iterative consensus calculations per time instant in its control design. As a result, the method is not fully distributed. To the best of our knowledge, there is no existing fully distributed algorithm solving the AJLATT problem.

Considering all the limitations of the previous works, we aim to propose fully distributed AJLATT algorithms. Our previous work [20] introduces a framework to solve the pure joint localization and target tracking (JLATT) problem without active motion strategy design in a fully distributed manner. Leveraging that framework, in this article, we move forward by considering how to design fully distributed motion strategies for each robot so that it can not only follow the target but also achieve better self-localization and target-tracking performance.

The contributions of this article are summarized as follows. Based on our previous fully distributed JLATT framework, two motion strategies are proposed from the control and optimization perspectives to actively improve each robot's self-localization and target tracking performance. The control-based algorithm is designed by incorporating the estimated target's and robots' states and their uncertainties as well as collision avoidance in the control policy. The optimization-based algorithm minimizes an objective function involving both the target's and robots' estimation uncertainties and a potential function that helps each robot avoid collision and maintain communication connectivity when the robot is planning its motion. Compared with the existing works on active target tracking from the control or optimization perspective, the current article is the first that solves the AJLATT problem in a fully distributed manner (i.e., requiring only one-hop information transmission, no global parameter in the algorithm, and no multiple communication iterations for iterative consensus-type calculations per time instant), to the best our knowledge. Extensive Monte-Carlo simulations are used to validate and compare the proposed control- and optimization-based algorithms.

Some preliminary results of this article are presented in our conference paper [32]. The current article expands the conference version by introducing a control-based approach and providing significantly additional simulation results including comparison with the centralized counterparts, adoption of different sensor models, and demonstration of the role of the potential function in the optimization-based approach.

II. Preliminaries

A. Motion and Measurement Models

We consider the scenario where M robots track a target on a surface. We use the vectors $\mathbf{x}_i^k$ and $\mathbf{x}_T^k$ to represent, respectively, the true state of robot i and the target at time k. Their movements are driven by a nonlinear motion model as follows:

$$\mathbf{x}_i^k = f_i\left(\mathbf{x}_i^{k-1}, \mathbf{u}_i^{k-1}, \mathbf{w}_i^{k-1}\right) \tag{1}$$

$$\mathbf{x}_T^k = g\left(\mathbf{x}_T^{k-1}, \mathbf{u}_T^{k-1}, \mathbf{w}_T^{k-1}\right) \tag{2}$$

where $\mathbf{u}_i^{k-1}$ and $\mathbf{u}_T^{k-1}$ are, respectively, the control inputs for robot i and the target. $\mathbf{w}_i^{k-1} \sim \mathcal{N}(0, \mathbf{Q}_i^{k-1})$ and $\mathbf{w}_T^{k-1} \sim \mathcal{N}(0, \mathbf{Q}_T^{k-1})$ are, respectively, the zero-mean white Gaussian process noises for robot i and the target.

We define $\bar{\mathbf{x}}_i^k$ and $\hat{\mathbf{x}}_i^k$ as, respectively, robot i's prior and posterior estimates of its true state $\mathbf{x}_i^k$, $\bar{\mathbf{x}}_{T_i}^k$, and $\hat{\mathbf{x}}_{T_i}^k$ as, respectively, each robot i's prior and posterior estimates of the target's state $\mathbf{x}_{T_i}^k$. Their corresponding approximated prior and posterior covariances are defined as $\bar{\mathbf{p}}_i^k$, $\hat{\mathbf{p}}_i^k$, $\bar{\mathbf{p}}_{T_i}^k$, and $\hat{\mathbf{p}}_{T_i}^k$, respectively.

At time k, if robot j or the target is within the sensing region of robot i, robot i can obtain a robot-to-robot measurement $\mathbf{z}_{R_{ij}}^k$ or a robot-to-target measurement $\mathbf{z}_{R_iT}^k$. The measurement models are defined as follows:

$$\begin{aligned}\mathbf{z}_{R_{ij}}^k &= h_{ij}\left(\mathbf{x}_i^k, \mathbf{x}_j^k\right) + \mathbf{v}_{R_{ij}}^k \\ \mathbf{z}_{R_{iT}}^k &= h_{\mathrm{iT}}\left(\mathbf{x}_i^k, \mathbf{x}_T^k\right) + \mathbf{v}_{R_iT}^k\end{aligned} \tag{3}$$

where $\mathbf{v}_{R_{ij}}^k \sim \mathcal{N}(0, \mathbf{R}_{ij}^k)$ and $\mathbf{v}_{R_iT}^k \sim \mathcal{N}(0, \mathbf{R}_{\mathrm{iT}}^k)$ are the measurement noises assumed to be zero-mean white Gaussian. The measurement noises are assumed to be mutually uncorrelated across robots and uncorrelated with the process noises.

B. Graphs

In the team of M robots, a directed communication graph $G_c^k = (\mathcal{V}, \mathcal{E}_c^k)$ is defined, where $\mathcal{V} = \{R_1, \ldots, R_M\}$ is the robot set and $\mathcal{E}_c^k \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set representing the communication links between robots at time k. If robot i receives information from robot j at time k, then a directed edge (j, i) exists in $\mathcal{E}_c^k$. We assume that the self-edge (i, i) exists in $\mathcal{E}_c^k$, $\forall i \in \mathcal{V}$, which means robot i can also use the information from itself. At time k, the communicating neighbor set of robot i is defined as $\mathcal{N}_{c,i}^k = \{l | (l, i) \in \mathcal{E}_c^k, \forall l \neq i, l \in \mathcal{V}\}$. Then, the inclusive communicating neighbor set of robot i is $\mathcal{I}_{c,i}^k = \mathcal{N}_{c,i}^k \cup \{i\}$. Similarly, we define a directed sensing graph $G_s^k = (\mathcal{V}, \mathcal{E}_s^k)$ to describe robot-to-robot measurements, where $\mathcal{E}_s^k \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set, which represents the detection links between robots at time k. If robot i can detect robot j at time k, a directed edge (j, i) exists in $\mathcal{E}_s^k$. At time k, we define robot i's sensing neighbors (all robots detected by robot i) as $\mathcal{N}_{s,i}^k = \{i | (l, i) \in \mathcal{E}_s^k, \forall l \neq i, l \in \mathcal{V}\}$.

We assume that for each robot, the communication radius is larger than the sensing radii of all robots. Then when robot i detects robot j, robot i can receive the information from robot j.

C. Information Fusion Strategy

Consistency is a vital property for estimation. An estimate pair $(\hat{\mathbf{p}}^k, \hat{\mathbf{x}}^k)$ is *consistent* if the true error covariance is upper bounded by the estimated covariance as $\mathbb{E}\{(\mathbf{x}^k - \hat{\mathbf{x}}^k)(\mathbf{x}^k - \hat{\mathbf{x}}^k)^\mathsf{T}\} \leq \hat{\mathbf{p}}^k$ [33]. An inconsistent estimate that underestimates the actual errors might eventually diverge. At time k, given multiple consistent estimation pairs $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$, $i = 1, \ldots, n$, of $\mathbf{x}^k$, we seek to compute an improved consistent estimate $(\hat{\mathbf{p}}_c^k, \hat{\mathbf{x}}_c^k)$ by using the covariance intersection (CI) algorithm [34]

$$\begin{aligned} \hat{\mathbf{p}}_c^k &= \left(\sum_{i=1}^n \alpha_i^k \left(\hat{\mathbf{p}}_i^k\right)^{-1} \right)^{-1} \\ \hat{\mathbf{x}}_c^k &= \hat{\mathbf{p}}_c^k \left(\sum_{i=1}^n \alpha_i^k \left(\hat{\mathbf{p}}_i^k\right)^{-1} \hat{\mathbf{x}}_i^k \right) \end{aligned} \tag{4}$$

where $\alpha_i^k \in [0, 1]$ and $\sum_{i=1}^n \alpha_i^k = 1$. The parameters α_i^k are usually chosen to satisfy certain optimal criterion such as minimizing the trace of $\hat{\mathbf{p}}_c^k$. In order to reduce the computation burden, we adopt a fast and simplified approach in [35] to calculate α_i^k as follows:

$$\alpha_i^k = \frac{1/\mathrm{tr}\left(\hat{\mathbf{p}}_i^k\right)}{\sum_{j=1}^n 1/\mathrm{tr}\left(\hat{\mathbf{p}}_j^k\right)} \tag{5}$$

where $\mathrm{tr}(\cdot)$ denotes the trace of a matrix.

D. Joint Localization and Target Tracking

In this section, we briefly introduce our previous work about JLATT [20]. In this work, each robot can estimate the pose of itself (localization) and the state of a target (tracking) using only its own information and the information from its one-hop communicating neighbors while preserving estimation consistency (up to linearization due to the EKF framework).

1) Robot Propagation: The estimate of robot i's state and its corresponding covariance are propagated at time $k-1$ as follows:

$$\begin{aligned} \bar{\mathbf{x}}_i^k &= f_i\left(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0\right) \\ \bar{\mathbf{p}}_i^k &= \boldsymbol{\Phi}_i^{k-1} \hat{\mathbf{p}}_i^{k-1} \left(\boldsymbol{\Phi}_i^{k-1}\right)^\mathsf{T} + \bar{\mathbf{Q}}_i^{k-1} \\ &= \rho_{Rp}\left(\hat{\mathbf{p}}_i^{k-1}, \hat{\mathbf{x}}_i^{k-1}, \mathbf{Q}_i^{k-1}\right) \end{aligned} \tag{6}$$

where $\boldsymbol{\Phi}_i^{k-1} = (\partial f_i / \partial \mathbf{x}_i)(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0)$, $\mathbf{G}_i^{k-1} = (\partial f_i / \partial \mathbf{w}_i)(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0)$, and $\bar{\mathbf{Q}}_i^{k-1} = \mathbf{G}_i^{k-1} \mathbf{Q}_i^{k-1} (\mathbf{G}_i^{k-1})^\mathsf{T}$.

2) Target Propagation: Robot i propagates its estimate of the target's state and its corresponding covariance at time $k-1$ as follows:

$$\begin{aligned} \bar{\mathbf{x}}_{T_i}^k &= g\left(\hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{u}_T^{k-1}, 0\right) \\ \bar{\mathbf{p}}_{T_i}^k &= \boldsymbol{\Phi}_{T_i}^k \hat{\mathbf{p}}_{T_i}^{k-1} \left(\boldsymbol{\Phi}_{T_i}^{k-1}\right)^\mathsf{T} + \bar{\mathbf{Q}}_{T_i}^{k-1} \\ &= \rho_{Tp}\left(\hat{\mathbf{p}}_{T_i}^{k-1}, \hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{Q}_{T_i}^{k-1}\right) \end{aligned} \tag{7}$$

where $\boldsymbol{\Phi}_{T_i}^{k-1} = (\partial g / \partial \mathbf{x}_T)(\hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{u}_T^{k-1}, 0)$, $\mathbf{G}_{T_i}^{k-1} = (\partial g / \partial \mathbf{w}_T)(\hat{\mathbf{x}}_{T_i}^{k-1}, \mathbf{u}_T^{k-1}, 0)$, and $\bar{\mathbf{Q}}_{T_i}^{k-1} = \mathbf{G}_{T_i}^{k-1} \mathbf{Q}_{T_i}^{k-1} (\mathbf{G}_{T_i}^{k-1})^\mathsf{T}$.

3) Robot State Update: To update the estimate of robot i's state, we first obtain the correction pairs $(\mathbf{s}_{R_{il}}^k, \mathbf{y}_{R_{il}}^k)$ and $(\mathbf{s}_{R_iT}^k, \mathbf{y}_{R_iT}^k)$ using robot i's robot-to-robot measurements $\mathbf{z}_{R_{il}}^k$, $l \in \mathcal{N}_{s,i}^k$, and robot-to-target measurement $\mathbf{z}_{R_iT}^k$, respectively. These correction pairs are calculated as follows:

$$\mathbf{s}_{R_{il}}^k = \left(\mathbf{H}_{il}^k\right)^\mathsf{T} \left(\bar{\mathbf{R}}_{il}^k\right)^{-1} \mathbf{H}_{il}^k \tag{8a}$$

$$\mathbf{y}_{R_{il}}^k = \left(\mathbf{H}_{il}^k\right)^\mathsf{T} \left(\bar{\mathbf{R}}_{il}^k\right)^{-1} \left(\bar{\mathbf{z}}_{R_{il}}^k + \mathbf{H}_{il}^k \bar{\mathbf{x}}_i^k\right) \tag{8b}$$

where $\mathbf{H}_{il}^k = (\partial \mathbf{h}_{il} / \partial \mathbf{x}_i^k)(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_l^k)$, $\bar{\mathbf{R}}_{il}^k = \mathbf{R}_{il}^k + \widetilde{\mathbf{H}}_{il}^k \bar{\mathbf{p}}_l^k (\widetilde{\mathbf{H}}_{il}^k)^\mathsf{T}$, $\widetilde{\mathbf{H}}_{il}^k = (\partial \mathbf{h}_{il} / \partial \mathbf{x}_l^k)(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_l^k)$, and $\bar{\mathbf{z}}_{R_{il}}^k = \mathbf{z}_{R_{il}}^k - \mathbf{h}_{il}(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_l^k)$, and

$$\mathbf{s}_{R_iT}^k = \left(\mathbf{H}_{\mathrm{iT}}^k\right)^\mathsf{T} \left(\bar{\mathbf{R}}_{\mathrm{iT}}^k\right)^{-1} \mathbf{H}_{\mathrm{iT}}^k \tag{9a}$$

$$\mathbf{y}_{R_iT}^k = \left(\mathbf{H}_{\mathrm{iT}}^k\right)^\mathsf{T} \left(\bar{\mathbf{R}}_{\mathrm{iT}}^k\right)^{-1} \left(\bar{\mathbf{z}}_{R_iT}^k + \mathbf{H}_{\mathrm{iT}}^k \bar{\mathbf{x}}_i^k\right) \tag{9b}$$

where $\mathbf{H}_{\mathrm{iT}}^k = (\partial \mathbf{h}_{\mathrm{iT}} / \partial \mathbf{x}_i^k)(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_{T_i}^k)$, $\bar{\mathbf{R}}_{\mathrm{iT}}^k = \mathbf{R}_{\mathrm{iT}}^k + \widetilde{\mathbf{H}}_{\mathrm{iT}}^k \bar{\mathbf{p}}_{T_i}^k (\widetilde{\mathbf{H}}_{\mathrm{iT}}^k)^\mathsf{T}$, $\widetilde{\mathbf{H}}_{\mathrm{iT}}^k = (\partial \mathbf{h}_{\mathrm{iT}} / \partial \mathbf{x}_{T_i}^k)(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_{T_i}^k)$, and $\bar{\mathbf{z}}_{R_iT}^k = \mathbf{z}_{R_iT}^k - \mathbf{h}_{\mathrm{iT}}(\bar{\mathbf{x}}_i^k, \bar{\mathbf{x}}_{T_i}^k)$.

Then, we apply the CI algorithm (4) on these correction pairs to compute a consistent (up to linearization) estimate $\breve{\mathbf{x}}_i$ and its corresponding covariance $\breve{\mathbf{p}}_i$ as follows:

$$\breve{\mathbf{p}}_i^k = \left(\sum_{l \in \mathcal{N}_{s,i}^k} \eta_{il}^k \mathbf{s}_{R_{il}}^k + \eta_{\mathrm{iT}}^k \mathbf{s}_{R_iT}^k \right)^{-1} \tag{10a}$$

$$\breve{\mathbf{x}}_i^k = \breve{\mathbf{p}}_i^k \left(\sum_{l \in \mathcal{N}_{s,i}^k} \eta_{il}^k \mathbf{y}_{R_{il}}^k + \eta_{\mathrm{iT}}^k \mathbf{y}_{R_iT}^k \right) \tag{10b}$$

where $\eta_{il}^k \in [0, 1]$, and $\eta_{\mathrm{iT}}^k = 0$ if robot i cannot detect the target and otherwise $\eta_{\mathrm{iT}}^k \in [0, 1]$ subject to $\sum_{l \in \mathcal{N}_{s,i}^k} \eta_{il}^k + \eta_{\mathrm{iT}}^k = 1$. The calculation of these η_{il} and η_{iT} follows the fast and simplified approach (5) with $(\mathbf{s}_{R_{il}}^k)^{-1}$ and $(\mathbf{s}_{R_iT}^k)^{-1}$ playing the role of the covariances.

After we obtain the estimation pair $(\check{\mathbf{p}}_i^k, \check{\mathbf{x}}_i^k)$ using relative measurements and also the prior estimation pair $(\bar{\mathbf{p}}_i^k, \bar{\mathbf{x}}_i^k)$ from the robot propagation step, we can use the CI algorithm (4) to fuse these two estimation pairs to obtain the posterior estimation pair $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$ as follows:

$$\hat{\mathbf{p}}_i^k = \left(\zeta_{i1}^k \left(\check{\mathbf{p}}_i^k\right)^{-1} + \zeta_{i2}^k \left(\bar{\mathbf{p}}_i^k\right)^{-1}\right)^{-1} \tag{11a}$$

$$\hat{\mathbf{x}}_i^k = \hat{\mathbf{p}}_i^k \left(\zeta_{i1}^k \left(\check{\mathbf{p}}_i^k\right)^{-1} \check{\mathbf{x}}_i^k + \zeta_{i2}^k \left(\bar{\mathbf{p}}_i^k\right)^{-1} \bar{\mathbf{x}}_i^k\right) \tag{11b}$$

where ζ_{i1}^k and $\zeta_{i2}^k \in [0, 1]$, subject to $\zeta_{i1}^k + \zeta_{i2}^k = 1$, are calculated according to (5).

4) Target State Update: Similarly, for the target state estimation, each robot i first collects the target correction pairs $(\tilde{\mathbf{s}}_{R_jT}^k, \tilde{\mathbf{y}}_{R_jT}^k)$ from available robot-to-target measurements calculated by its inclusive communicating neighbors j, $j \in \mathcal{I}_{c,i}^k$

$$\tilde{\mathbf{s}}_{R_jT}^k = \left(\widetilde{\mathbf{H}}_{jT}^k\right)^{\mathsf{T}} \left(\widetilde{\mathbf{R}}_{jT}^k\right)^{-1} \widetilde{\mathbf{H}}_{jT}^k \tag{12a}$$

$$\tilde{\mathbf{y}}_{R_jT}^k = \left(\widetilde{\mathbf{H}}_{jT}^k\right)^{\mathsf{T}} \left(\widetilde{\mathbf{R}}_{jT}^k\right)^{-1} \left(\bar{\mathbf{z}}_{R_jT}^k + \widetilde{\mathbf{H}}_{jT}^k \bar{\mathbf{x}}_{T_j}^k\right) \tag{12b}$$

where $\widetilde{\mathbf{R}}_{jT}^k = \mathbf{R}_{jT}^k + \mathbf{H}_{jT}^k \bar{\mathbf{p}}_j^k (\mathbf{H}_{jT}^k)^{\mathsf{T}}$.

Then, at time k, by combining all available target correction pairs, we obtain an intermediate estimation pair as follows:

$$\check{\mathbf{p}}_{T_i}^k = \left(\sum_{j \in \mathcal{I}_{c,i}^k} \tilde{\eta}_j^k \tilde{\mathbf{s}}_{R_jT}^k\right)^{-1} \tag{13a}$$

$$\check{\mathbf{x}}_{T_i}^k = \check{\mathbf{p}}_{T_i}^k \left(\sum_{j \in \mathcal{I}_{c,i}^k} \tilde{\eta}_j^k \tilde{\mathbf{y}}_{R_jT}^k\right) \tag{13b}$$

where $\tilde{\eta}_j^k = 0$ if robot j cannot directly detect the target, and otherwise $\tilde{\eta}_j^k \in [0, 1]$ subject to $\sum_{j \in \mathcal{I}_{c,i}^k} \tilde{\eta}_j^k = 1$.

Then we fuse all available prior estimation pairs $(\bar{\mathbf{p}}_{T_j}^k, \bar{\mathbf{x}}_{T_j}^k)$, $\forall j \in \mathcal{I}_{c,i}^k$ with the CI algorithm (4) as follows:

$$\check{\mathbf{p}}_{T_i}^k = \left(\sum_{j \in \mathcal{I}_{c,i}^k} \pi_j^k \left(\bar{\mathbf{p}}_{T_j}^k\right)^{-1}\right)^{-1} \tag{14a}$$

$$\check{\mathbf{x}}_{T_i}^k = \check{\mathbf{p}}_{T_i}^k \left(\sum_{j \in \mathcal{I}_{c,i}^k} \pi_j^k \left(\bar{\mathbf{p}}_{T_j}^k\right)^{-1} \bar{\mathbf{x}}_{T_j}^k\right) \tag{14b}$$

where $\pi_j^k \in [0, 1]$, subject to $\sum_{j \in \mathcal{I}_{c,i}^k} \pi_j^k = 1$, is calculated according to (5).

Eventually, after obtaining the pairs $(\check{\mathbf{p}}_{T_i}^k, \check{\mathbf{x}}_{T_i}^k)$ and $(\check{\mathbf{p}}_{T_i}^k, \check{\mathbf{x}}_{T_i}^k)$, we calculate the posterior estimation pair $(\hat{\mathbf{p}}_{T_i}^k, \hat{\mathbf{x}}_{T_i}^k)$ of the target as follows:

$$\hat{\mathbf{p}}_{T_i}^k = \left(\zeta_{iT_1}^k \left(\check{\mathbf{p}}_{T_i}^k\right)^{-1} + \zeta_{iT_2}^k \left(\check{\mathbf{p}}_{T_i}^k\right)^{-1}\right)^{-1} \tag{15a}$$

$$\hat{\mathbf{x}}_{T_i}^k = \hat{\mathbf{p}}_{T_i}^k \left(\zeta_{iT_1}^k \left(\check{\mathbf{p}}_{T_i}^k\right)^{-1} \check{\mathbf{x}}_{T_i}^k + \zeta_{iT_2}^k \left(\check{\mathbf{p}}_{T_i}^k\right)^{-1} \check{\mathbf{x}}_{T_i}^k\right) \tag{15b}$$

where $\zeta_{iT_1}^k$ and $\zeta_{iT_2}^k \in [0, 1]$, subject to $\zeta_{iT_1}^k + \zeta_{iT_2}^k = 1$, are calculated according to (5).

TABLE I
AJLATT BY ROBOT i

Initialization:
1 Initialize $\hat{\mathbf{x}}_i^0, \hat{\mathbf{p}}_i^0, \hat{\mathbf{x}}_{T_i}^0, \hat{\mathbf{p}}_{T_i}^0$, and set $u_i^{-1} = 0$.

At time $k-1$
Information Exchange for AJLATT:
2.1 Send $\hat{\mathbf{x}}_i^{k-1}, \hat{\mathbf{p}}_i^{k-1}, \mathbf{u}_i^{k-2}, f_i(\cdot), \hat{\mathbf{x}}_{T_i}^{k-1}, \hat{\mathbf{p}}_{T_i}^{k-1}$ to robot j, $i \in \mathcal{N}_{c,j}^{k-1}$.
2.2 Receive $\hat{\mathbf{x}}_j^{k-1}, \hat{\mathbf{p}}_j^{k-1}, \mathbf{u}_j^{k-2}, f_j(\cdot), \hat{\mathbf{x}}_{T_j}^{k-1}, \hat{\mathbf{p}}_{T_j}^{k-1}$ from robot j, $j \in \mathcal{N}_{c,i}^{k-1}$.

AJLATT Motion Planning:
3 (1) Calculate $\mathbf{u}_i^{k-1}$ using the control-based AJLATT algorithm (19).
3 (2) Calculate $\mathbf{u}_i^{k-1}$ using the optimization-based AJLATT algorithm (24).
Propagation:
4 Propagate robot i's estimate of its own state $\hat{\mathbf{x}}_i^{k-1}$ and covariance $\hat{\mathbf{p}}_i^{k-1}$ with the obtained $\mathbf{u}_i^{k-1}$ using (6) and the target's state $\hat{\mathbf{x}}_{T_i}^{k-1}$ and covariance $\hat{\mathbf{p}}_{T_i}^{k-1}$ using (7) to obtain $\bar{\mathbf{x}}_i^k$, $\bar{\mathbf{p}}_i^k$, $\bar{\mathbf{x}}_{T_i}^k$ and $\bar{\mathbf{p}}_{T_i}^k$.

At time k
Update:
5.1 Obtain the robot-to-robot measurements $\mathbf{z}_{R_{il}}^k$, $l \in \mathcal{N}_{s,i}^k$, and robot-to-target measurement $\mathbf{z}_{R_iT}^k$ (if the target is detected by robot i) and generate the corresponding correction pairs $(\mathbf{s}_{R_{il}}^k, \mathbf{y}_{R_{il}}^k)$, $(\mathbf{s}_{R_iT}^k, \mathbf{y}_{R_iT}^k)$, $(\tilde{\mathbf{s}}_{R_iT}^k, \tilde{\mathbf{y}}_{R_iT}^k)$ using (8), (9), (12).
5.2 Send $(\tilde{\mathbf{s}}_{R_iT}^k, \tilde{\mathbf{y}}_{R_iT}^k), (\bar{\mathbf{p}}_{T_i}^k, \bar{\mathbf{x}}_{T_i}^k)$ to robot j, $i \in \mathcal{N}_{c,j}^k$.
5.3 Receive $(\tilde{\mathbf{s}}_{R_jT}^k, \tilde{\mathbf{y}}_{R_jT}^k), (\bar{\mathbf{p}}_{T_j}^k, \bar{\mathbf{x}}_{T_j}^k)$ from robot j, $j \in \mathcal{N}_{c,i}^k$.
5.4 Calculate the posterior robot estimate pair $(\hat{\mathbf{p}}_i^k, \hat{\mathbf{x}}_i^k)$ using (10), (11), and the posterior target estimate pair $(\hat{\mathbf{p}}_{T_i}^k, \hat{\mathbf{x}}_{T_i}^k)$ using (13), (14), (15).

III. Active Joint Localization and Target Tracking

While our previous work [20] achieves fully distributed JLATT, the robots' motions are not actively controlled or planned and they simply move around randomly. As a result, the full potential to improve the localization and target tracking performance is not exploited. In this article, we aim to actively control the motions of the robots to achieve better localization and target tracking performance than random movements. We use the term AJLATT to emphasize the fact that we try to not only estimate the states of the robots and the target but also design proper control actions to improve the estimation performance. Our goal is to design the control input $\mathbf{u}_i^k$ for each robot i modeled by (1) at each time k in a fully distributed manner by using only its own information and the information from its one-hop communicating neighbors. Next, we present two approaches, one control based and the other optimization based.

An overview of the proposed AJLATT algorithm is summarized in Table I. In *Step 3*, we either implement our control-based AJLATT algorithm in *Step 3 (1)* using (19) or optimization-based AJLATT algorithm in *Step 3 (2)* using (24) to solve the AJLATT problem. Detailed explanations for these two algorithms are given in the following Sections III-A and III-B.

A. Control-Based Approach

In order to obtain a more accurate estimate of the target's state, one way is to approach the target. The first reason is that the detection range of a sensor is limited in general. Besides, the measurement noise usually has a positive relationship with the measurement distance of the detected object during a certain range interval. Second, when the robots move closer to the target, they are closer to other robots, which will also intuitively help improve self-localization. While the distributed tracking problem has been studied in the controls community extensively (see, e.g., [21]), most of the works assume the poses of the robots and even the target are known (no estimation needed), which is not realistic. Therefore, in this section, we propose a fully distributed algorithm to solve the AJLATT problem from the perspective of control.

Consider robot i with the model

$$\dot{\mathbf{r}}_i(t) = \mathbf{u}_i(t) \tag{16}$$

where $\mathbf{r}_i(t) = [x_i(t), y_i(t)]^T$ is the position in 2-D, and $\mathbf{u}_i(t) = [u_{xi}(t), u_{yi}(t)]^T$ is the control input. The control input $u_i(t)$ is designed for (16) as follows:

$$\mathbf{u}_i(t) = \mathbf{v}_T(t) - \alpha\left[\sum_{j\in\mathcal{N}_{c,i}(t)}\left(\frac{\partial V_{ij}}{\partial \mathbf{r}_i(t)}\right)^T + \left(\frac{\partial V_{\mathrm{iT}}}{\partial \mathbf{r}_i(t)}\right)^T\right] - \gamma[\mathbf{r}_i(t) - \mathbf{r}_T(t)] \tag{17}$$

where $\mathbf{r}_T(t)$ and $\mathbf{v}_T(t)$ are the target's position and velocity in 2-D at time t, α, and γ are two positive constants which are used to adjust, respectively, the influence of the collision avoidance term and active target tracking term, and the differentiable, nonnegative functions V_{iT} and V_{ij} are, respectively, and the potential function defined on, respectively, $\|\mathbf{r}_i(t) - \mathbf{r}_T(t)\|$ and $\|\mathbf{r}_i(t) - \mathbf{r}_j(t)\|$ for collision avoidance as detailed later. In the following, we omit the time dependence (t) for notation simplicity.

The potential function V_{ij} has the following properties.

1) V_{ij} achieves its unique minimum when $\|\mathbf{r}_i - \mathbf{r}_j\|$ is equal to its desired value d_{o_i}.
2) $V_{ij} \to \infty$ if $\|\mathbf{r}_i - \mathbf{r}_j\| \to d_{s_i}$, where d_{s_i} $(0 < d_{s_i} < d_{o_i})$ is the safe distance for collision avoidance.
3) $\partial V_{ij}/\partial\|\mathbf{r}_i - \mathbf{r}_j\| = \mathbf{0}$, if $\|\mathbf{r}_i - \mathbf{r}_j\| \geqslant R_i$, where $R_i > d_{o_i}$ denotes the communication radius of robot i.

The potential function V_{iT} is defined analogously.

Consider the Lyapunov function candidate

$$V = \frac{\alpha}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} V_{ij} + \alpha\sum_{i=1}^{n} V_{\mathrm{iT}} + \frac{1}{2}\gamma\sum_{i=1}^{n}||\mathbf{r}_i - \mathbf{r}_T||^2.$$

Define $\mathbf{a}_i = -\alpha\sum_{j=1}^{n}(\partial V_{ij}/\partial \mathbf{r}_i)^T$ and $\mathbf{b}_i = -\alpha(\partial V_{\mathrm{iT}}/\partial \mathbf{r}_i)^T - \gamma(\mathbf{r}_i - \mathbf{r}_T)$. Due to property (3) of V_{ij}, we have $\sum_{j=1}^{n}(\partial V_{ij}/\partial \mathbf{r}_i)^T = \sum_{j\in\mathcal{N}_{c,i}}(\partial V_{ij}/\partial \mathbf{r}_i)^T$. It thus follows from (16) to (17) that $\dot{\mathbf{r}}_i - \mathbf{v}_T = \mathbf{a}_i + \mathbf{b}_i$. Note that $(\partial V_{ij}/\partial \mathbf{r}_i) = -(\partial V_{ij}/\partial \mathbf{r}_j)$ and $(\partial V_{\mathrm{iT}}/\partial \mathbf{r}_i) = -(\partial V_{\mathrm{iT}}/\partial \mathbf{r}_T)$. The derivative of V is given by the following equation:

$$\begin{aligned}
\dot{V} &= \alpha\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial V_{ij}}{\partial \mathbf{r}_i}\dot{\mathbf{r}}_i + \alpha\sum_{i=1}^{n}\frac{\partial V_{\mathrm{iT}}}{\partial \mathbf{r}_i}(\dot{\mathbf{r}}_i - \mathbf{v}_T) \\
&\quad + \gamma\sum_{i=1}^{n}(\mathbf{r}_i - \mathbf{r}_T)^T(\dot{\mathbf{r}}_i - \mathbf{v}_T) \\
&= \alpha\sum_{i=1}^{n}\sum_{j=1}^{n}\frac{\partial V_{ij}}{\partial \mathbf{r}_i}(\mathbf{v}_T + \mathbf{a}_i + \mathbf{b}_i) \\
&\quad + \sum_{i=1}^{n}(\dot{\mathbf{r}}_i - \mathbf{v}_T)^T\left[\alpha\left(\frac{\partial V_{\mathrm{iT}}}{\partial \mathbf{r}_i}\right)^T + \gamma(\mathbf{r}_i - \mathbf{r}_T)\right] \\
&= -\sum_{i=1}^{n}\mathbf{a}_i^T(\mathbf{v}_T + \mathbf{a}_i + \mathbf{b}_i) - \sum_{i=1}^{n}(\mathbf{a}_i + \mathbf{b}_i)^T\mathbf{b}_i \\
&= -\mathbf{v}_T^T\sum_{i=1}^{n}\mathbf{a}_i - \sum_{i=1}^{n}||\mathbf{a}_i + \mathbf{b}_i||^2
\end{aligned}$$

where we have used [21, Lemma 3.1] to derive the first equality. Note that $\sum_{i=1}^{n}\mathbf{a}_i = 0$. It follows that $\dot{V} \leq 0$. As a result, the robots chase the target while avoiding collisions.

After discretization, the model (16) becomes

$$\mathbf{r}_i^k = \mathbf{r}_i^{k-1} + \mathbf{u}_i^{k-1}\delta t \tag{18}$$

where $\mathbf{r}_i^{k-1} = [x_i^{k-1}, y_i^{k-1}]^T$ is the position in 2-D at time $k-1$, $\mathbf{u}_i^{k-1} = [u_{xi}^{k-1}, u_{yi}^{k-1}]^T$ is the control input in 2-D at time $k-1$, and δt is the sampling interval. We will later adapt the model and design to address more realistic robot models (e.g., nonholonomic differential drive robots). Let $\mathbf{r}_T^{k-1}$ and $\mathbf{v}_T^{k-1}$ denote, respectively, the target's position and velocity at time $k-1$. Also let $\hat{\mathbf{r}}_i^{k-1}$, $\hat{\mathbf{r}}_{T_i}^{k-1}$, and $\hat{\mathbf{v}}_{T_i}^{k-1}$ denote, respectively, robot i's estimate of its own position, the target's position, and the target's velocity. Motivated by (17), we design the control $\mathbf{u}_i^{k-1}$ for (18) as follows:

$$\mathbf{u}_i^{k-1} = \sum_{j\in\mathcal{I}_{c,i}^{k-1}}\eta_j^{k-1}\hat{\mathbf{v}}_{T_j}^{k-1} - \alpha\left[\sum_{m\in\mathcal{N}_{c,i}^{k-1}}\left(\frac{\partial V_{im}}{\partial \hat{\mathbf{r}}_i^{k-1}}\right)^T + \left(\frac{\partial V_{\mathrm{iT}}}{\partial \hat{\mathbf{r}}_i^{k-1}}\right)^T\right] - \gamma\left(\hat{\mathbf{r}}_i^{k-1} - \sum_{j\in\mathcal{I}_{c,i}^{k-1}}\eta_j^{k-1}\hat{\mathbf{r}}_{T_j}^{k-1}\right) \tag{19}$$

where α and γ are defined as in (17)

$$\eta_j^{k-1} = \frac{1/\mathrm{tr}\left(\hat{\mathbf{p}}_{T_j}^{k-1}\right)}{\sum_{l\in\mathcal{I}_{c,i}^{k-1}}1/\mathrm{tr}\left(\hat{\mathbf{p}}_{T_l}^{k-1}\right)} \tag{20}$$

is the weight denoting the certainty of neighbor j's estimate of the target's state from the inclusive communicating neighbor set of robot i, and the potential functions V_{im} and V_{iT} are as in (17) but defined on $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_m\|$ and $\|\hat{\mathbf{r}}_i - \sum_{j\in\mathcal{I}_{c,i}^{k-1}}\eta_j^{k-1}\hat{\mathbf{r}}_{T_j}\|$ for collision avoidance. Note that (19) is fully distributed, using only the information from each robot itself and its one-hop communicating neighbors.

The motivation behind (19) is to push each robot toward the weighted average of its own and communicating neighbors' estimates of the target's position while avoiding collisions with the estimated positions of its communicating neighbors

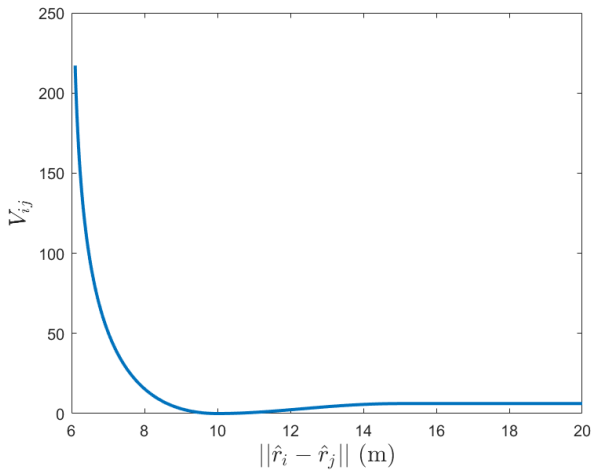


Fig. 1. Potential function V_{ij} with $d_{s_i} = 6$ m, $d_{o_i} = 10$ m, and $R_i = 15$ m.

as well as the estimated positions of the target on itself and its communicating neighbors. By doing so, the robots are actively driven to move closer to the target while avoiding collisions. As a result, the target (respectively, each robot) will more likely or frequently be in the field of view of the robots (respectively, some other robots), which will generate more robot-to-target and robot-to-robot measurements. In addition, the robots will be able to communicate with other robots and maintain communication connectivity for the entire team. Generally, more measurements and communication would bring better estimation performance, which makes our control-based approach achieve good self-localization and target-tracking performance.

Motivated by [21], we choose V_{ij} such that

$$\frac{\partial V_{ij}}{\partial \hat{\mathbf{r}}_i} = \begin{cases} -\infty \mathrm{sgn}\left(\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\right) & \|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| < d_{s_i} \\ 20\frac{(\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j)}{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\|}\frac{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| - d_{o_i}}{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| - d_{s_i}}, & d_{s_i} \leqslant \|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| < d_{o_i} \\ 0.5\frac{(\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j)\sin\left[\frac{\pi}{R_i - d_{o_i}}\left(\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| - d_{o_i}\right)\right]}{\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\|}, & d_{o_i} \leqslant \|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_j\| < R_i \\ \mathbf{0}, & \text{otherwise} \end{cases} \tag{21}$$

where $\mathrm{sgn}(\cdot)$ denotes the sign function defined component-wise. The potential function V_{iT} is defined analogously. An example of V_{ij} is shown in Fig. 1. Note that unlike [21], we do not have the actual positions of the robots and the target, $\mathbf{r}_i$ and $\mathbf{r}_T$, and hence the actual distances between robots and between robot and target. Therefore, the potential functions are defined on the estimated relative distance between robots $\|\hat{\mathbf{r}}_i - \hat{\mathbf{r}}_m\|$ and that between robot and target $\|\hat{\mathbf{r}}_i - \sum_{j\in\mathcal{I}_{c,i}^{k-1}} \eta_j^{k-1}\hat{\mathbf{r}}_{T_j}\|$ instead.

The control input (19) is defined according to a simplified model (18). The approach can be applied to a system that can be feedback linearized as an integrator-type system. Many fully actuated systems and under-actuated systems (the portion corresponding to the actuated degrees of freedom) can be feedback linearized as integrator systems. In fact, some quadrotor kinematics can be feedback linearized as an integrator model. Next, we use the commonly used unicycle model as an example to show how we extend the calculated control input $\mathbf{u}_i^{k-1} = [u_{xi}^{k-1}, u_{yi}^{k-1}]^T$ in (19) to design the linear and angular velocity inputs in the unicycle model.

Consider a unicycle model described as follows:

$$\begin{aligned} x_i^k &= x_i^{k-1} + v_i^{k-1}\delta t\cos\left(\theta_i^{k-1}\right) \\ y_i^k &= y_i^{k-1} + v_i^{k-1}\delta t\sin\left(\theta_i^{k-1}\right) \\ \theta_i^k &= \theta_i^{k-1} + \omega_i^{k-1}\delta t \end{aligned} \tag{22}$$

where (x_i^{k-1}, y_i^{k-1}) is the position, θ_i^{k-1} is the orientation, v_i^{k-1} is the linear velocity, and ω_i^{k-1} is the angular velocity associated with robot i at time $k-1$. With $\mathbf{u}_i^{k-1} = [u_{xi}^{k-1}, u_{yi}^{k-1}]^T$ given by (19), we can calculate the desired linear velocity as follows:

$$v_{i\,d}^{k-1} = \sqrt{\left(u_{xi}^{k-1}\right)^2 + \left(u_{yi}^{k-1}\right)^2}$$

and the desired orientation is as follows:

$$\theta_{i\,d}^{k-1} = \mathrm{atan2}\left(u_{yi}^{k-1}, u_{xi}^{k-1}\right).$$

Then we can design the linear and angular velocity control input as follows:

$$\begin{aligned} v_i^{k-1} &= v_{i\,d}^{k-1} \\ \omega_i^{k-1} &= -\lambda_i\left(\theta_i^{k-1} - \theta_{i\,d}^{k-1}\right) \end{aligned} \tag{23}$$

where λ_i is a positive constant.

B. Optimization Based Approach

In this section, we propose an optimization-based approach to solve the AJLATT problem. Our goal is to find an optimal control input $\mathbf{u}_i^{k-1}$ for each robot i modeled by (1), to optimize certain functions.

There exist some previous works solving a related problem from the optimization perspective. However, they either ignore the robot localization [24], [25], [26], [27], [28] or are implemented in a centralized manner [29], [30]. In contrast, here robot localization is explicitly considered and the problem is solved in a fully distributed manner by using each robot's own and its one-hop neighbors' information. There is no center node required for computation, global parameter shared among the entire team or information transmitted via multiple hops.

Our optimization-based AJLATT algorithm can be summarized as follows:

$$\begin{aligned} \mathbf{u}_i^{k-1} = \underset{\mathbf{u}_i^{k-1}\in\mathcal{U}_i}{\arg\min}\ & \alpha_{R_i}\mathrm{tr}\left(\hat{\mathbf{p}}_i^{k+t_2}\right) + \alpha_{T_i}\mathrm{tr}\left(\hat{\mathbf{p}}_{T_i}^{k+t_2}\right) \\ & + \beta_i \sum_{j\in\mathcal{N}_{c,i}^{k-1}\cup\{T_i\}} J_{R_{ij}}^{k+t_1}, \quad i = 1, \ldots, M \\ \text{s.t. } & \left(\bar{\mathbf{x}}_j^{k+t_1}, \bar{\mathbf{x}}_{T_j}^{k+t_1}, \hat{\mathbf{p}}_i^{k+t_2}, \hat{\mathbf{p}}_{T_i}^{k+t_2}\right) \\ = & \ \mathrm{SACP}\left(\hat{\mathbf{x}}_j^{k-1}, \hat{\mathbf{x}}_{T_j}^{k-1}, \hat{\mathbf{p}}_j^{k-1}, \hat{\mathbf{p}}_{T_j}^{k-1}, \mathbf{u}_j^{k-2}, \mathbf{u}_T^{k-1}\right) \\ & j \in \mathcal{I}_{c,i}^{k-1} \end{aligned} \tag{24}$$

where $\mathcal{U}_i$ is the control space of robot i. α_{R_i}, α_{T_i}, and β_i are three positive constant parameters that are used to adjust the influence of each term on the objective function. $\hat{\mathbf{p}}_i^{k+t_2}$ and $\hat{\mathbf{p}}_{T_i}^{k+t_2}$ are, respectively, the predicted robot and target covariances at time $k + t_2$ representing, respectively, robot i's predicted self-localization and target tracking uncertainty.

Here a grid-search-based method is adopted to find the optimal solution. We will explore more efficient methods in future work. As the traces of the robot and target covariances are good measures of the robot self-localization and target tracking performance, we incorporate them in the objective function in (24). By optimizing the traces of the predicted robot and target covariances, each robot plans its motion to improve its self-localization and target tracking performance. Of course, depending on specific applications or corresponding preferences, other criteria (e.g., observability and information) other than traces could be used to construct the objective function. $J_{R_{ij}}^{k+t_1}$, $j \in \mathcal{N}_{c,i}^{k-1} \cup \{T_i\}$, is the potential function term at time $k+t_1$, which is incorporated to the objective function to maintain communication connectivity, avoid collisions, and keep sight of the target. t_1 and t_2 are, respectively, the planning horizons for the potential function term and covariance terms. In general, we have $t_1 < t_2$, since the difference of uncertainty between different control inputs needs more time to show up, but the potential function term that helps robots to quickly avoid collision and frequently keep communication connectivity needs to be computed in a shorter time period. SACP is a function that is used to predict the estimate of the states $\bar{\mathbf{x}}_j^{k+t_1}$, $\bar{\mathbf{x}}_{T_j}^{k+t_1}$ and covariances $\hat{\mathbf{p}}_i^{k+t_2}$, $\hat{\mathbf{p}}_{T_i}^{k+t_2}$ at time $k+t_1$ and $k+t_2$, respectively, and will be shown in detail later. Note that the optimization-based approach is fully distributed, using only the information from each robot itself and its one-hop communicating neighbors.

Let $\mathbf{r}_i = [x_i, y_i]^T$ be the position part of the state $\hat{\mathbf{x}}_i$ in (1). For each robot i, the potential function $J_{R_{ij}}$ is defined as follows:

$$J_{R_{ij}} = \begin{cases} \infty, & \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \leqslant \underline{d}_i \\ -10\log\left(\frac{\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| - \underline{d}_i}{a_i}\right), & \underline{d}_i < \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \\ & \leqslant \underline{d}_i + a_i \\ 0, & \underline{d}_i + a_i < \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \\ & \leqslant \bar{d}_i - a_i \\ 10\left(\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| - (\bar{d}_i - a_i)\right)^2, & \|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| > \bar{d}_i - a_i \end{cases} \tag{25}$$

where $\bar{\mathbf{r}}_i = [\bar{x}_i, \bar{y}_i]^T$ is the prior estimate of $\mathbf{r}_i$, $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\|$, $j \in \mathcal{N}_{c,i} \cup \{T_i\}$, is the estimated robot-to-robot or robot-to-target distance, $\underline{d}_i$ and $\bar{d}_i$ are, respectively, the minimum and maximum acceptable distances that are used to avoid collisions and maintain communication connectivity, and a_i is the length of the nonzero interval. Note that the definition of $J_{R_{ij}}$ accommodates both robot-to-robot potential and robot-to-target potential. An example of the potential function is shown in Fig. 2.

According to the definition of the potential function $J_{R_{ij}}$, there are several properties that we would like to point out.

1) $J_{R_{ij}} = 0$, when $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \in [\underline{d}_i + a_i, \bar{d}_i - a_i]$, which means that only when other robots or the target moves too close to the minimum range or almost moves out of the maximum acceptable range, this potential function will play a role. Otherwise, only the covariance terms take effect in the objective function in (24).
2) $J_{R_{ij}} \to \infty$, if $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \to \underline{d}_i$ or $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| \to \infty$. As shown in Fig. 2, the potential function will immediately give a large penalty (close to infinity) when the robot comes too close to its neighbors or the target to avoid collisions. In contrast, it gives a relatively soft and weak penalty when the robot moves far away from its neighbors or the target so as to maintain communication connectivity or not lose sight of the target, respectively.

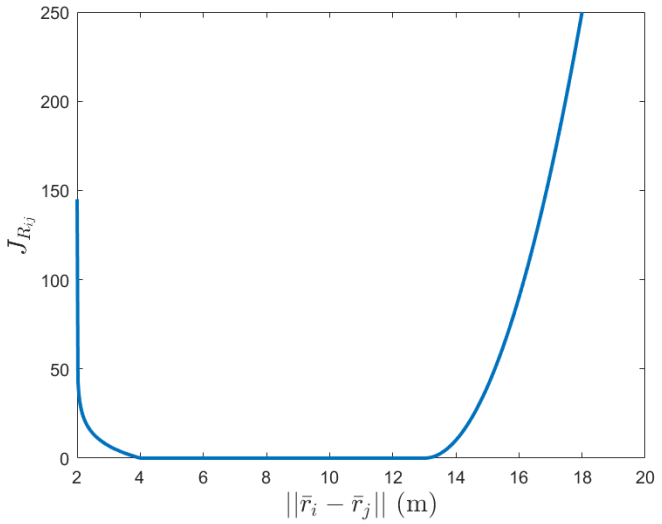


Fig. 2. Potential function $J_{R_{ij}}$ with $\underline{d}_i = 2$ m, $\bar{d}_i = 15$ m, and $a_i = 2$ m. Notice that the right-hand side increases more moderately than the left-hand side, and is still defined when $\|\bar{\mathbf{r}}_i - \bar{\mathbf{r}}_j\| > \bar{d}_i$.

The state and covariance prediction (SACP) process is first used to predict robot i's estimates of its own and the target's states, and robot i's estimates of each neighbor j's state at time $k+t_1$ so as to calculate the potential function term $J_{R_{ij}}^{k+t_1}$. The process is consecutively used to predict robot i's covariance $\hat{\mathbf{p}}_i^{k+t_2}$ and the target covariance $\hat{\mathbf{p}}_{T_i}^{k+t_2}$ at time $k+t_2$ which constitute the uncertainty minimization terms in the objective function in (24). The SACP process is shown in Table II.

Since $\mathbf{u}_j^{k-1}$ is the optimization variable for each robot j, robot i propagates its neighbor j's state estimate with the neighbor's latest available control input $\mathbf{u}_j^{k-2}$ at time $k-2$. Robot i's own state is propagated with $\mathbf{u}_i^{k-1}$. The target's state is propagated with $\mathbf{u}_T^{k-1}$, which is assumed to be known for each robot at time $k-1$.

In Step 1.1 in Table II, robot i propagates its inclusive communicating neighbor j's, $j \in \mathcal{I}_{c,i}^{k-1}$, estimates of its own and the target's states and covariances from time $k-1$ to k. In Step 1.2.1, ${}^*\mathcal{N}_{s,i}^k$ and ${}^*\mathcal{I}_{c,i}^k$ are two predicted neighbor sets at time k in the prediction process. Since the real movement does not happen, robot i can only utilize the information from its communicating neighbors at time $k-1$. Thus ${}^*\mathcal{N}_{s,i}^k$ and ${}^*\mathcal{I}_{c,i}^k$ are the subsets of the known set $\mathcal{I}_{c,i}^{k-1}$, where ${}^*\mathcal{N}_{s,i}^k = \{i|(l,i) \in ({}^*\mathcal{E}_s^k \cap \mathcal{E}_c^{k-1}), \forall l \neq i, l \in \mathcal{V}\}$ and ${}^*\mathcal{I}_{c,i}^k = \{i|(l,i) \in ({}^*\mathcal{E}_c^k \cap \mathcal{E}_c^{k-1}), l \in \mathcal{V}\}$. Here ${}^*\mathcal{E}_s^k$ and ${}^*\mathcal{E}_c^k$ are, respectively, the predicted sensing and communication edge set at time k which are determined by the predicted estimates of the robots' states $\bar{\mathbf{x}}_j^k$ together with, respectively, the robots' sensing fields of view and communication radii. In addition, η_{iT}^k in (10a) is determined by $\bar{\mathbf{x}}_i^k$ and the estimate of the target's state $\bar{\mathbf{x}}_{T_i}^k$ together with the robots' sensing fields of view. Although real future robot-to-robot or robot-to-target measurements are not available, the noise characteristics of the sensors can be obtained and the Jacobians associated with the measurement models can be calculated according to the

TABLE II
SACP BY ROBOT i AT $k-1$

1. One-step Propagation and Predicted Update:

1.1 Propagate each inclusive communicating neighbor j's, $j \in \mathcal{I}_{c,i}^{k-1}$, estimates of its own and the target's states and covariances from time $k-1$ to k:

$$\begin{aligned}
\bar{\mathbf{x}}_i^k &= f_i(\hat{\mathbf{x}}_i^{k-1}, \mathbf{u}_i^{k-1}, 0), \\
\bar{\mathbf{p}}_i^k &= \rho_{Rp}(\hat{\mathbf{p}}_i^{k-1}, \hat{\mathbf{x}}_i^{k-1}, \mathbf{Q}_i^{k-1}), \\
\bar{\mathbf{x}}_j^k &= f_j(\hat{\mathbf{x}}_j^{k-1}, \mathbf{u}_j^{k-2}, 0), & j \in \mathcal{N}_{c,i}^{k-1} \\
\bar{\mathbf{p}}_j^k &= \rho_{Rp}(\hat{\mathbf{p}}_j^{k-1}, \hat{\mathbf{x}}_j^{k-1}, \mathbf{Q}_j^{k-1}), & j \in \mathcal{N}_{c,i}^{k-1} \\
\bar{\mathbf{x}}_{T_j}^k &= g(\hat{\mathbf{x}}_{T_j}^{k-1}, \mathbf{u}_T^{k-1}, 0), & j \in \mathcal{I}_{c,i}^{k-1} \\
\bar{\mathbf{p}}_{T_j}^k &= \rho_{Tp}(\hat{\mathbf{p}}_{T_j}^{k-1}, \hat{\mathbf{x}}_{T_j}^{k-1}, \mathbf{Q}_{T_j}^{k-1}), & j \in \mathcal{I}_{c,i}^{k-1}
\end{aligned}$$

1.2 Calculate one-step predicted robot posterior covariance $\hat{\mathbf{p}}_i^k$ and target posterior covariance $\hat{\mathbf{p}}_{T_i}^k$ at time k:

1.2.1 Generate one-step predicted correction terms $\mathbf{s}_{R_{il}}^k$, $l \in {}^*\mathcal{N}_{s,i}^k$, $\mathbf{s}_{R_iT}^k$, $\tilde{\mathbf{s}}_{R_jT}^k$, $j \in {}^*\mathcal{I}_{c,i}^k$, using (8a), (9a), (12a).

1.2.2 Calculate $\hat{\mathbf{p}}_i^k$ using (10a), (11a), and $\hat{\mathbf{p}}_{T_i}^k$ using (13a), (14a), (15a).

2. Propagation and Optional Predicted Update in t_1 and t_2 Planning Horizons:

For $t = 0, \ldots, t_2 - 1$, execute the following Steps 2.1-2.4:

2.1 Propagate each inclusive communicating neighbor j's, $j \in \mathcal{I}_{c,i}^{k-1}$, estimates of its own and the target's states and covariances from time $k+t$ to $k+t+1$:

$$\begin{aligned}
\bar{\mathbf{x}}_i^{k+t+1} &= f_i(\bar{\mathbf{x}}_i^{k+t}, \mathbf{u}_i^{k-1}, 0), \\
\bar{\mathbf{p}}_i^{k+t+1} &= \rho_{Rp}(\hat{\mathbf{p}}_i^{k+t}, \bar{\mathbf{x}}_i^{k+t}, \mathbf{Q}_i^{k+t}), \\
\bar{\mathbf{x}}_j^{k+t+1} &= f_j(\bar{\mathbf{x}}_j^{k+t}, \mathbf{u}_j^{k-2}, 0), & j \in \mathcal{N}_{c,i}^{k-1} \\
\bar{\mathbf{p}}_j^{k+t+1} &= \rho_{Rp}(\bar{\mathbf{p}}_j^{k+t}, \bar{\mathbf{x}}_j^{k+t}, \mathbf{Q}_j^{k+t}), & j \in \mathcal{N}_{c,i}^{k-1} \\
\bar{\mathbf{x}}_{T_j}^{k+t+1} &= g(\bar{\mathbf{x}}_{T_j}^{k+t}, \mathbf{u}_T^{k-1}, 0), & j \in \mathcal{I}_{c,i}^{k-1} \\
\bar{\mathbf{p}}_{T_i}^{k+t+1} &= \rho_{Tp}(\hat{\mathbf{p}}_{T_i}^{k+t}, \bar{\mathbf{x}}_{T_i}^{k+t}, \mathbf{Q}_{T_i}^{k+t}), \\
\bar{\mathbf{p}}_{T_j}^{k+t+1} &= \rho_{Tp}(\bar{\mathbf{p}}_{T_j}^{k+t}, \bar{\mathbf{x}}_{T_j}^{k+t}, \mathbf{Q}_{T_j}^{k+t}), & j \in \mathcal{N}_{c,i}^{k-1}
\end{aligned}$$

2.2 **Option 1:** Calculate one-step predicted robot posterior covariance $\hat{\mathbf{p}}_i^{k+t+1}$ and target posterior covariance $\hat{\mathbf{p}}_{T_i}^{k+t+1}$ at time $k+t+1$ as in Step 1.2.
Option 2 (for computation saving purpose): Set $\hat{\mathbf{p}}_i^{k+t+1} = \bar{\mathbf{p}}_i^{k+t+1}$ and $\hat{\mathbf{p}}_{T_i}^{k+t+1} = \bar{\mathbf{p}}_{T_i}^{k+t+1}$.

2.3 If $t = t_1 - 1$, compute $J_{R_{ij}}^{k+t_1}$ using $\|\bar{\mathbf{r}}_i^{k+t_1} - \bar{\mathbf{r}}_j^{k+t_1}\|$, $j \in \mathcal{N}_{c,i}^{k-1} \cup \{T_i\}$.

predicted state estimates. With that information, the one-step predicted correction terms can be generated in Step 1.2.1. The one-step predicted posterior covariances can then be calculated in Step 1.2.2.

In Step 2.1, robot i propagates from time $k+t$ to $k+t+1$ similar to Step 1.1. In contrast, as the posterior estimates $\hat{\mathbf{x}}_j^{k+t}$ and $\hat{\mathbf{x}}_{T_j}^{k+t}$, $j \in \mathcal{I}_{c,i}^{k-1}$, are no longer available due to the lack of real future robot-to-robot or robot-to-target measurements, the prior estimates are used for propagation. In Step 2.2, there are two options. Options 1 and 2 correspond to, respectively, multistep and one-step predicted posterior covariance calculation. Option 1 calculates robot i's predicted posterior covariances for itself and the target at time $k+t+1$ while Option 2 omits this procedure to save computation costs. In Step 2.3, $J_{R_{ij}}^{k+t_1}$ is computed. The potential function term in (24) would result in a larger penalty when robot i loses communication connectivity with its neighbor j, $j \in \mathcal{N}_{c,i}^{k-1}$, at time $k+t_1$. Instead, if the potential function term were given by $\beta_i \sum_{j \in {}^*\mathcal{N}_{c,i}^{k+t_1} \cup \{T_i\}} J_{R_{ij}}^{k+t_1}$ in (24), then a smaller value would be obtained due to the removal of some robots in ${}^*\mathcal{N}_{c,i}^{k+t_1}$. If Option 2 is chosen, because predicted posterior covariances are not computed, there is no need to calculate $\bar{\mathbf{p}}_j^{k+t+1}$ and $\bar{\mathbf{p}}_{T_j}^{k+t+1}$ for $t = 0, \ldots, t_2-1$ and $\bar{\mathbf{x}}_j^{k+t+1}$ for $t = t_1, \ldots, t_2-1$ in Step 2.1.

In the fully distributed setting, the motion of neighboring robots is predicted using old control inputs while the control inputs of these agents will be optimized in the SACP. There would be differences between the old control inputs and the optimized ones. In this setting, it is not clear how the robots can coordinate their control decisions using updated optimized control inputs because such coordination might require certain information propagation through multiple hops or multiple communication iterations per time instant, which does not exist in the current fully distributed setting. Hence there is a tradeoff between how well the algorithm can perform and how distributed the entire system can be. As the robots have a similar objective of tracking the target, the target covariance as well as the connectivity maintenance potential function in each robot's cost function provide certain coordination among robots in the sense that the robots would try to chase the target and stay close to the target and their neighbors to make sure that the team communication graph is not disconnected. In future work, we will explore how to coordinate the robots' control decisions in a distributed setting.

There is a tradeoff between the control- and optimization-based approaches. The control-based approach is computationally simple and time efficient in real-world implementations. However, a simplified model is adopted. It is worth noticing that the control policy here is not designed to explicitly optimize a certain criterion on the localization and target tracking performance. Instead, the hope is to bring the robots closer so as to improve the localization and target tracking performance. In contrast, the optimization-based approach is more performance-improvement-oriented and applicable to more general robot models. However, the computation load of the optimization-based approach is high. In particular, with the control-based approach, each robot incorporates information from its communicating neighbors. Each robot's computation complexity increases linearly with the number of its communicating neighbors, denoted by $\mathcal{O}(|\mathcal{N}_{c,i}|)$. In comparison, with the optimization-based approach, each robot uses a grid-search-based method to find its control input. Let k_d denote the dimension of each robot's control input (for instance, if a robot has two inputs, linear and angular velocities, then $k_d = 2$), and n_i denote the grid size of the ith dimension. The grid size of the control input is $\prod_{i=1}^{k_d} n_i$. For each combination in the grid search, the computation load of the SACP process is linear in $|\mathcal{N}_{c,i}|$. As a result, the computation load of each robot becomes $\mathcal{O}(|\mathcal{N}_{c,i}| \times \prod_{i=1}^{k_d} n_i)$.

IV. SIMULATION

In this section, we will use Monte-Carlo simulations to demonstrate the performance of our algorithms.

A. Simulation Setup

Consider the scenario where $M = 6$ robots and a target move on a surface. Here we adopt the widely used unicycle model for both robots and the target in the simulation. The robot pose $\mathbf{x}_i^k$ consists of the position (x_i^k, y_i^k) and the orientation θ_i^k in the global frame. The motion models (1) and (2) can be expressed as follows:

$$\begin{aligned} x_i^k &= x_i^{k-1} + \left(v_i^{k-1} + w_{v_i}^{k-1}\right)\delta t \cos\left(\theta_i^{k-1}\right) \\ y_i^k &= y_i^{k-1} + \left(v_i^{k-1} + w_{v_i}^{k-1}\right)\delta t \sin\left(\theta_i^{k-1}\right) \\ \theta_i^k &= \theta_i^{k-1} + \left(\omega_i^{k-1} + w_{\omega_i}^{k-1}\right)\delta t \end{aligned} \tag{26}$$

where $i \in \{1, \ldots, M\} \cup \{T\}$, $\delta t = 1$ s is the sampling interval, $\mathbf{u}_i^{k-1} = [v_i^{k-1}, \omega_i^{k-1}]^T$ represents the linear and angular velocities as the input for robot i, and $\mathbf{w}_i = [w_{v_i}^{k-1}, w_{\omega_i}^{k-1}]^T$ represents process noises for the linear and angular velocities. For robot i, the input $\mathbf{u}_i^{k-1}$ is calculated by our AJLATT algorithms. The target's linear velocity input is assumed to be constant with $v_T = 0.25$ m/s while the angular velocity input ω_T is uniformly generated from the interval $[-(\pi/6), (\pi/6)]$ rad/s. The target's input $\mathbf{u}_T = [v_T, \omega_T]^T$ is known by every robot i. The corresponding process noise $\mathbf{w}_i$, $i \in \{1, \ldots, M\} \cup \{T\}$, is assumed to be white Gaussian, with the standard deviations for $w_{v_i}^{k-1}$ and $w_{\omega_i}^{k-1}$ as, respectively, $\sigma_{v_i}^{k-1} = (\sqrt{2}/2)\sigma_i^{k-1}$ and $\sigma_{\omega_i}^{k-1} = 2\sqrt{2}\sigma_i^{k-1}$, where σ_i^{k-1} is proportional to the linear velocity as $\sigma_i^{k-1} = 1\% v_i^{k-1}$ for each robot i, $i \in \{1, \ldots, M\}$, and $\sigma_T^{k-1} = 3\% v_T^{k-1}$ for the target. Hence $\mathbf{Q}_i^{k-1}$, $i \in \{1, \ldots, M\} \cup \{T\}$ defined after (1) and (2) is given as follows:

$$\mathbf{Q}_i^{k-1} = \begin{bmatrix} \left(\sigma_{v_i}^{k-1}\right)^2 & 0 \\ 0 & \left(\sigma_{\omega_i}^{k-1}\right)^2 \end{bmatrix}. \tag{27}$$

Given the model (26), it follows from (6) that

$$\mathbf{\Phi}_i^{k-1} = \begin{bmatrix} 1 & 0 & -v_i^{k-1}\delta t \sin\left(\theta_i^{k-1}\right) \\ 0 & 1 & v_i^{k-1}\delta t \cos\left(\theta_i^{k-1}\right) \\ 0 & 0 & 1 \end{bmatrix}. \tag{28}$$

According to (27) and (28), when robot i stops moving (i.e., $v_i = 0$), $\mathbf{Q}_i^{k-1}$ and $\mathbf{\Phi}_i^{k-1}$ will become, respectively, the zero matrix $\mathbf{0}_{2\times 2}$ and the identity matrix I_3. As a result, it follows from (6) that $\bar{\mathbf{p}}_i^k = \hat{\mathbf{p}}_i^{k-1}$, which means that the robot covariance will not increase during propagation.

We assume that each robot has a limited communication range with a radius of $R_i = 30$ m and a limited field of view with $R_{\min} = 2$ m and $R_{\max} = 15$ m for the range and $\phi = 60^\circ$ for the angle of view. As for the measurement model, we consider an indoor application scenario in this work and assume that these robots do not have access to absolute position measurement. The relative distance-bearing measurement model is mainly adopted in simulation. If robot i detects robot j at time instant k, then the relative measurement can be expressed as follows:

$$\mathbf{z}_{R_{ij}}^k = \begin{bmatrix} \sqrt{\left(x_j^k - x_i^k\right)^2 + \left(y_j^k - y_i^k\right)^2} \\ \text{atan2}\left(\left(y_j^k - y_i^k\right), \left(x_j^k - x_i^k\right)\right) - \theta_i^k \end{bmatrix} + \mathbf{v}_{R_{ij}}^k$$

where $\mathbf{v}_{R_{ij}}$ is a zero-mean white Gaussian noise. The standard deviation of the distance noise is set to be 3% of the actual distance, and the standard deviation of the bearing noise equals 1°. The same measurement model is used for the robot-to-target measurement $\mathbf{z}_{R_iT}$. To show the applicability of our algorithms in different scenarios, we will also include simulation results using distance-only and bearing-only measurement models.

Since the absolute measurement is not available, we assume that each robot initializes its estimated pose $\hat{\mathbf{x}}_i^0$ with its true pose $\mathbf{x}_i^0$, and the initial pose covariance $\hat{\mathbf{p}}_i^0$ is set to $\hat{\mathbf{p}}_i^0 = 10^{-3}\mathbf{I}_3$. The initial estimate of the target's state obtained by each robot i, $\hat{\mathbf{x}}_{T_i}^0$, does not necessarily equal the true initial state of the target $\mathbf{x}_T^0$. In our simulation, we set $\hat{\mathbf{x}}_T^0 = [10, 10, 0]$ while the true initial target state is $\mathbf{x}_T^0 = [10, 5, 0]$. Since we assume that an accurate initial target state is not available, we initialize the target covariance with relatively large uncertainty as $\hat{\mathbf{p}}_{T_i}^0 = 4\mathbf{I}_3$.

We compare the performance of the following five algorithms.

1) *Random Motion (RM):* In this case, each robot moves with a constant linear velocity of $v_i = 0.5$ m/s. Its angular velocity ω_i is uniformly chosen from an interval of $[-(\pi/5), (\pi/5)]$ rad/s. These robots behave as in our previous work [20] except that there is no moving field boundary for them. By adopting the RM strategy, robots do not tend to pursue the target or maintain communication connectivity with other robots, which eventually results in worse localization and tracking performance than other AJLATT algorithms as shown later.
2) *Control-Based AJLATT (AJLATT-C):* The control-based AJLATT algorithm uses the control policy in Section III-A. The parameters as set as $\alpha = 0.02$, $\gamma = 1$ in (19), $d_{s_i} = 6$ m, $d_{o_i} = 10$ m in (21), and $\lambda_i = 1$ in (23). The estimated target velocity $\hat{\mathbf{v}}_{T_j}$ in (19) is calculated by $\hat{\mathbf{v}}_{T_j} = v_T\left[\cos(\hat{\theta}_{T_j})\ \sin(\hat{\theta}_{T_j})\right]^T$, where v_T is the known target's velocity, and $\hat{\theta}_{T_j}$ is the target's orientation estimated by robot j. This algorithm drives all of the robots to track the target while avoiding collisions. As a result, the robots tend to maintain communication connectivity with others and observe the target and other robots more often than RM. As shown later, it has better performance than RM.
3) *Optimization-Based AJLATT (AJLATT-O):* In this setting, the robots' linear and angular velocities are calculated by the optimization-based AJLATT algorithm in Section III-B. The parameters of each term in the objective function (24) are set as $\alpha_{R_i} = 3$, $\alpha_{T_i} = 2$, and $\beta_i = 1$. As for the parameters on each robot i's robot-to-robot and robot-to-target potential functions (25), we set the length of the nonzero interval as $a_i = 2$ m and $\underline{d}_i = 2$ m. We also set $\bar{d}_i = 30$ m in (25) for the robot-to-robot potential function to make the robots maintain the communication with their neighbors, and $\bar{d}_i = 20$ m for the robot-to-target potential function to help keep sight of the target. For the SACP process, the planning horizons are $t_1 = 4$ and $t_2 = 11$ and Option 2 is adopted. Based on our tests, Option 2 balances good performance and computation costs while Option 1 generally achieves better performance for target position estimation and

comparable performance for other parts but with significant computation costs.

4) *Centralized Version of Control-Based AJLATT (ACEKF-C):* ACEKF-C is the centralized version of AJLATT-C, which consists of a centralized Kalman filter estimator and a centralized controller that incorporates information from all robots instead of only the communicating neighbors.
5) *Centralized Version of Optimization-Based AJLATT (ACEKF-O):* Similarly, ACEKF-O adopts a centralized Kalman filter for estimation and a centralized motion planner, which plans the motions for all robots simultaneously. ACEKF-O is summarized as follows:

$$\begin{aligned}\mathbf{u}^{k-1} &= \arg\min_{\mathbf{u}^{k-1}\in\mathcal{U}} \frac{\alpha_R}{M}\mathrm{tr}\left(\hat{\mathbf{p}}_R^{k+t_2}\right) + \alpha_T \mathrm{tr}\left(\hat{\mathbf{p}}_T^{k+t_2}\right) \\ &\quad + \frac{\beta}{M}\sum_{i\in\{1,\ldots,M\}}\sum_{j\in\{1,\ldots,M\}\setminus\{i\}} J_{R_{ij}}^{k+t_1} \\ &\quad \text{s.t. } \left(\bar{\mathbf{x}}^{k+t_1}, \hat{\mathbf{p}}^{k+t_2}\right) \\ &= \mathrm{SACP}_{\mathrm{CEKF}}\left(\hat{\mathbf{x}}^{k-1}, \hat{\mathbf{p}}^{k-1}\right)\end{aligned}$$

where $\mathbf{x}^\ell = [\mathbf{x}_1^{\ell\,T}, \ldots, \mathbf{x}_M^{\ell\,T}, \mathbf{x}_T^{\ell\,T}]^T$ is a vector comprised of the true states of all robots and the target at time ℓ. $\bar{\mathbf{x}}^\ell$ and $\hat{\mathbf{x}}^\ell$ are, respectively, the prior and posterior estimates of $\mathbf{x}^\ell$ with corresponding covariances $\bar{\mathbf{p}}^\ell$ and $\hat{\mathbf{p}}^\ell$. $\hat{\mathbf{p}}_R$ and $\hat{\mathbf{p}}_T$ are, respectively, the blocks in $\hat{\mathbf{p}}^\ell$ that are associated with, respectively, all the robots and the target. $\mathbf{u}^{\ell-1} = [u_1^{\ell-1\,T}, \ldots, u_M^{\ell-1\,T}]^T$ is a vector comprised of all robots' control inputs. $\mathcal{U}$ is the control space associated with $\mathbf{u}$. α_R, α_T, and β are three positive constant parameters. $\mathrm{SACP}_{\mathrm{CEKF}}$ denotes the centralized version of the SACP process, which implements the propagation and update of the centralized Kalman filter.

The control space for each robot i is set as $\mathcal{U} = \{(v_i, \omega_i) | v_i \in [0, 0.5]$ m/s, $\omega_i \in [-(\pi/5), (\pi/5)]$ rad/s$\}$. For the optimization-based algorithms, the grid size for v_i (respectively, ω_i) is set as 11 with an increment of 0.05 m/s (respectively, $(\pi/25)$ rad/s).

We run 50 Monte Carlo simulations and use the root mean square error (RMSE) as the metric for accuracy to test the performance of these algorithms with the distance-bearing model. Fig. 3 shows each robot's average target position and orientation estimate RMSE for target tracking, and Fig. 4 shows each robot's own average position and orientation estimate RMSE for localization. As shown in Figs. 3 and 4, AJLATT-C and AJLATT-O achieve good performance even if, as expected, their centralized counterparts ACEKF-C and ACEKF-O have better performance. However, the centralized approaches require the existence of a central station or node that collects all data from every robot, conducts extensive computation, and then broadcasts the control commands back to each robot. Such approaches are not practical in reality due to the bottleneck in communication and the computation load as well as the single point of failure of the central station. AJLATT-C and AJLATT-O achieve significantly better performance than RM. The result confirms our expectation. The RM approach does not actively drive one robot to observe the target and other robots or maintain communication connectivity with its neighbors. As a result, the robot obtains fewer measurements and has fewer neighbors to exchange

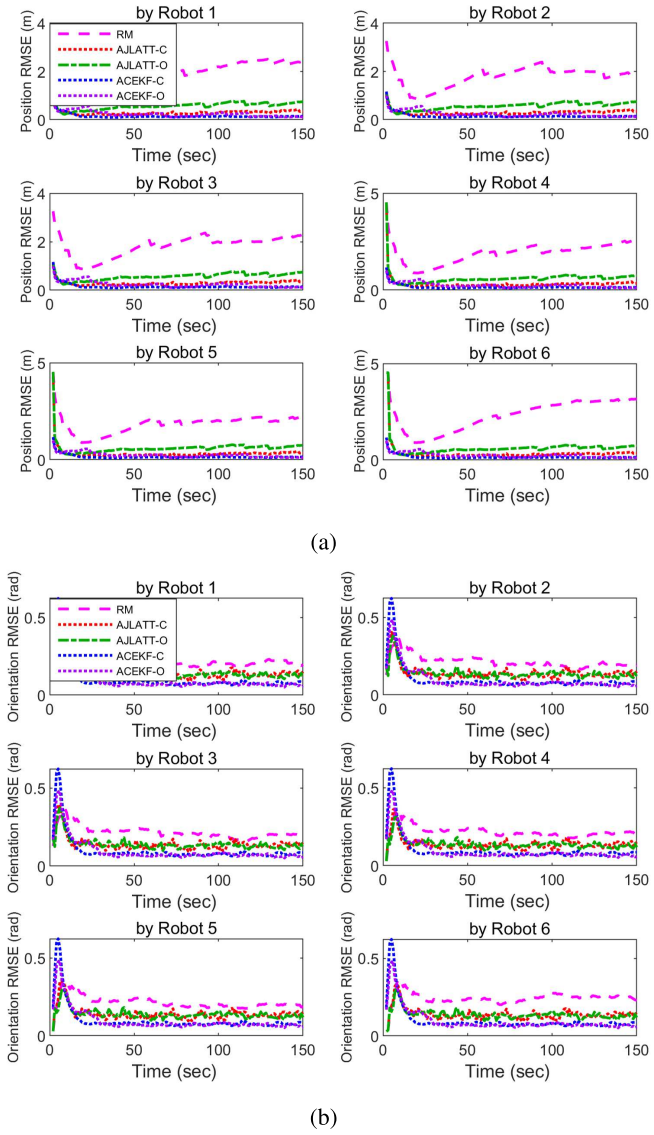
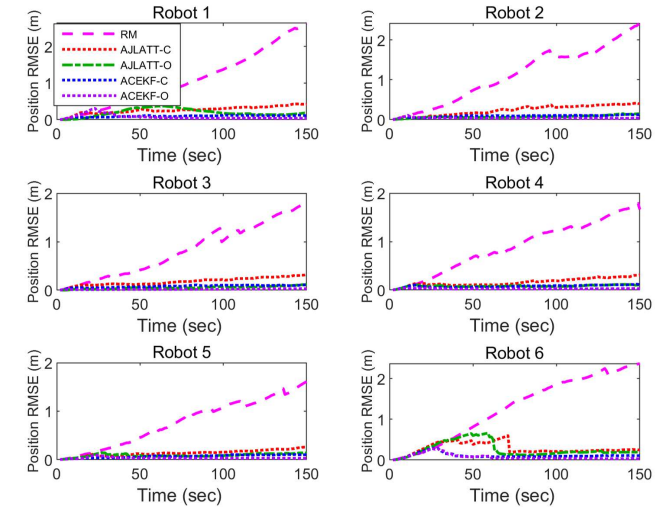


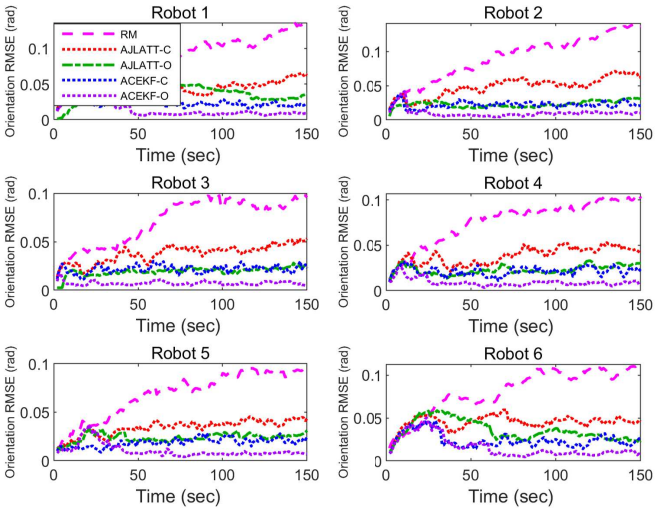
Fig. 3. Position and orientation estimate RMSE for the target on six robots (tracking) with distance-bearing model. (a) Position estimate RMSE. (b) Orientation estimate RMSE.

information, thus obtaining less accurate estimates of its own and the target's states compared with the AJLATT algorithms. Hence the RM approach has the worst performance. We can also notice that AJLATT-C and AJLATT-O have comparable performance with AJLATT-O being slightly better largely while each of these two algorithms has its own benefit. The optimization-based approach is not limited to specific models while the control-based one is computationally simple.

We also run 50 Monte Carlo simulations with, respectively, the distance-only and bearing-only models. Due to space limitation, we summarize the results of the average RMSE of six robots and 50 Monte Carlo simulations for all three measurement models at $k = 150$ in Table III. As can be seen, our proposed AJLATT algorithms still achieve good performance with distance-only and bearing-only measurement models. As expected, the performance with the distance-bearing model is slightly better than the distance-only or bearing-only ones. For each measurement



(a)



(b)

Fig. 4. Position and orientation estimate RMSE for six robots (localization) with distance-bearing model. (a) Position estimate RMSE comparison. (b) Orientation estimate RMSE Comparison.

model, the AJLATT algorithms significantly outperform the RM approach.

One thing worth noticing is the necessity of the potential function term in the objective function in (24). On the one hand, as stated in Section III-B, the potential function term is used to avoid collisions. On the other hand, it also helps each robot to maintain a certain distance between itself and its teammates and between itself and the target. If a robot moves too close or too far away from its communicating neighbors or the target, the robot will receive a penalty. Although purely minimizing the target covariance term to some extent pushes these robots to chase the target, it will not always force all of the robots to chase the target at the same time. There are two reasons. First, a robot with a larger trace of the self-localization covariance may move its field of view away from observing the target and let its neighbors observe the target instead. A robot might have a larger trace of the self-localization covariance than its neighbors due to its movement. The previous movement of the robot such as

TABLE III

AVERAGE RMSE OF SIX ROBOTS AND 50 MONTE CARLO SIMULATIONS FOR THREE MEASUREMENT MODELS AT $k = 150$

		RM	AJLATT-C	AJLATT-O	ACEKF-C	ACEKF-O
Distance-Bearing	Robot Position	2.0868	0.3272	0.1494	0.1100	0.0309
	Robot Orientation	0.1120	0.0513	0.0299	0.0227	0.0074
	Target Position	2.4323	0.3434	0.7343	0.1425	0.1162
	Target Orientation	0.1996	0.1510	0.1177	0.0857	0.0589
Distance-only	Robot Position	3.4778	1.0051	0.3198	0.3220	0.1828
	Robot Orientation	0.1573	0.1152	0.0713	0.0505	0.0328
	Target Position	3.8036	0.9107	0.7682	0.3075	0.6442
	Target Orientation	0.2715	0.1647	0.1240	0.0863	0.0963
Bearing-only	Robot Position	2.6287	0.4395	0.3861	0.1312	0.0536
	Robot Orientation	0.1461	0.0654	0.0335	0.0235	0.0086
	Target Position	3.9788	0.6362	0.8273	0.1608	0.2200
	Target Orientation	0.4040	0.1583	0.1068	0.0946	0.0761

continuously chasing the target would result in a large and consecutive nonzero v_i and would hence cause large $\mathbf{Q}_i^{k-1}$ and $\mathbf{\Phi}_i^{k-1}\hat{\mathbf{p}}_i^{k-1}(\mathbf{\Phi}_i^{k-1})^{\mathsf{T}}$ according to (27) and (28). From (6), these two terms will together induce the increase of the robot covariance during propagation. If the robot keeps sight of the target, its large trace of the self-localization covariance might make the trace of the fused target covariance $\hat{\mathbf{p}}_{T_i}^k$ calculated by (15a) (see Step 1.2.2 in Table II) larger than the case that it does not observe the target, which will eventually cause the increment of the cost of the target covariance term in the objective function (24). Therefore, in that case, the robot with larger self-localization covariance will move its field of view away from observing the target. Second, if we add the robot covariance term in addition to the target covariance term without using the potential function term, robots might slow down or even stop moving to slow down the increase of the self-localization uncertainty caused by its movement. These two factors will gradually make only a few robots keep sight of the target. That brings benefits in the short term for the localization performance of the robots that stop moving. However, in the long term, these robots that stop moving could lose communication connectivity with the robots that keep sight of the target. As a result, these robots that stop moving might not be able to observe the target and eventually lose their target state update. Besides, for the few robots that keep sight of the target, due to the reduction of the robot-to-robot measurements obtained among the entire team, they will gradually calculate inaccurate estimates of their own states at first and then the target's state, which might even eventually result in totally losing sight of the target. By adding the potential function term in the objective function (24), some robots might still move their fields of view away from the target, and only a few robots will keep sight of the target. However, since the robots are close enough to each other,

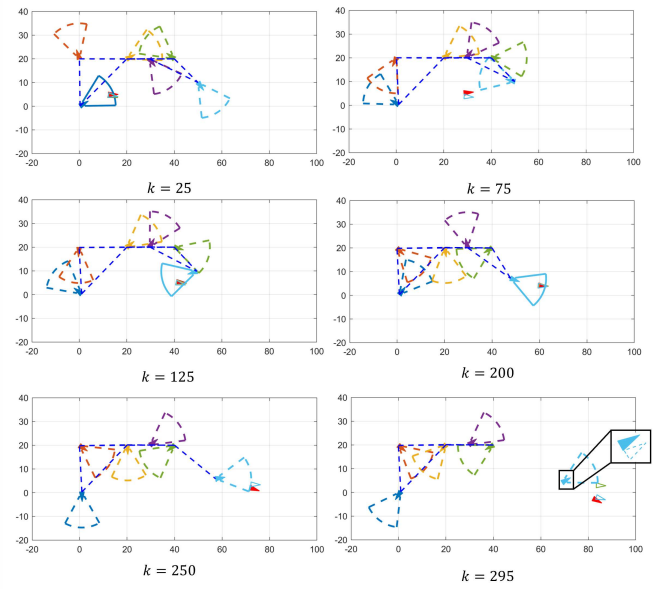


Fig. 5. Snapshots of the robots without the potential function term in the objective function (24). The red solid triangle denotes the true target, and the hollow triangles with solid lines (overlapped with the true target at the beginning) in different colors denote the estimated targets on different robots. Solid triangles and their attached sectors in different colors represent, respectively, different true robots and their corresponding fields of view. The lines of a robot's sector are solid when the target is in the robot's field of view; otherwise, they are dashed. The hollow triangles with dash lines in different colors represent different robots' self-estimates.

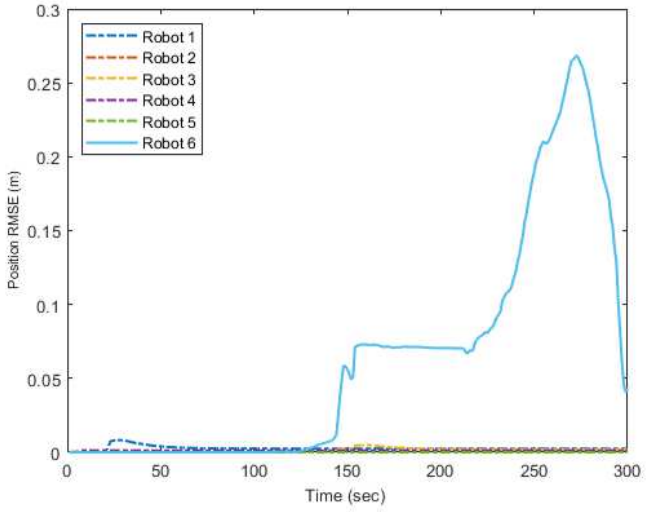


Fig. 6. Robot position RMSE for the example illustrated by Fig. 5. The cyan solid line corresponds to the cyan robot (robot 6 here), and the dashed lines with different colors correspond to other robots in the same color in the snapshots in Fig. 5.

a robot that has a smaller self-localization covariance but does not observe the target is able to quickly replace the role of the robots that are currently keeping sight of the target but having large self-localization covariances to observe the target.

The phenomenon mentioned above can be shown in Fig. 5 in snapshots, where we use $\alpha_{R_i} = 3$, $\alpha_{T_i} = 2$, and $\beta_i = 0$ [i.e., no potential function term in the objective function in (24)], and the target's angular velocity ω_T is set to 0 to show this phenomenon more clearly. As we can see, at the beginning ($k = 10$), one robot moves its field of view to observe the target. However, without the penalty introduced by the potential function for keeping a distance with the target, the robots tend to not chase and keep sight of the target as can be seen at $k = 75$. After a certain time ($k = 125$), only the cyan robot keeps sight of the target driven by the goal to minimize the large cost associated with the target covariance at that time, while the other robots stop moving. As shown in Fig. 6, the continuous movement through keeping sight of the target and the lack of robot-to-robot measurements by the cyan robot (robot 6) gradually induce a dramatic increase of the self-localization error compared with other robots that stop moving and also induce a large target estimation error as shown in Fig. 5 at $k = 295$. Besides, there also comes communication disconnection between robots. As can be seen, the cyan robot loses direct communication connectivity with the green one (and hence the rest of the team) at $k = 295$. In a word, if there lacks the potential function term, AJLATT-O might fail to work.

V. Conclusion and Discussions

In this article, we have proposed two algorithms to solve the AJLATT problem in a fully distributed (communication, estimation, planning) manner to drive a team of robots to actively track a target and localize itself so as to achieve better self-localization and target tracking performance.

The first control-based algorithm explicitly incorporates the estimates of their own states and the target's state and collision avoidance in algorithm design. The other algorithm based on the optimization framework tries to find the optimal motion so that optimal robot localization and target tracking performance can be achieved while collision avoidance and communication maintenance are considered at the same time. Monte Carlo simulations are performed to illustrate the effectiveness of our approaches. Factors that influence the performance of the optimization-based approach are discussed. The simulation result shows that both approaches work well, and their performance is comparable. Each of these two approaches has its benefits. The control-based approach is computationally simple and time efficient, while the optimization-based approach can be applied to a wide range of realistic models. Our work also has limitations. One limitation is that we have assumed that the target model is known in the sense that there is some rough idea about the target's input (not necessarily accurate) with the uncertain part modeled as noise. If the target model is unknown, one idea might be to assume that the target follows a generic model (e.g., unknown constant linear and angular velocities with velocities as part of the state to be estimated) for estimation purposes. Another idea might be the multiple model approach. While how to identify the target model is not the focus of this article, the motion planning algorithms proposed in this article can be applied and adapted to the case where any good estimation scheme can be adopted to estimate the target's state (whether the target model is known or unknown). In addition, the proposed control algorithm uses estimates to replace the true values. When dealing with nonlinear systems, it is worth rigorous analysis in terms of the effectiveness and stability of the resulting system.

REFERENCES

[1] S. I. Roumeliotis and G. A. Bekey, "Distributed multirobot localization," *IEEE Trans. Robot. Automat.*, vol. 18, no. 5, pp. 781–795, Oct. 2002.

[2] S. S. Kia, S. F. Rounds, and S. Martinez, "A centralized-equivalent decentralized implementation of extended Kalman filters for cooperative localization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 3761–3766.

[3] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman, and H. Durrant-Whyte, "Decentralised cooperative localisation for heterogeneous teams of mobile robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 2859–2865.

[4] T. R. Wanasinghe, G. K. I. Mann, and R. G. Gosine, "Distributed leader-assistive localization method for a heterogeneous multirobotic system," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 795–809, Jul. 2015.

[5] A. Martinelli, "Improving the precision on multi robot localization by using a series of filters hierarchically distributed," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2007, pp. 1053–1058.

[6] S. Panzieri, F. Pascucci, and R. Setola, "Multirobot localisation using interlaced extended Kalman filter," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2006, pp. 2816–2821.

[7] L. Luft, T. Schubert, S. I. Roumeliotis, and W. Burgard, "Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication," *Int. J. Robot. Res.*, vol. 37, no. 10, pp. 1152–1167, 2018.

[8] L. C. Carrillo-Arce, E. D. Nerurkar, J. L. Gordillo, and S. I. Roumeliotis, "Decentralized multi-robot cooperative localization using covariance intersection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 1412–1417.

[9] A. Bahr, M. R. Walter, and J. J. Leonard, "Consistent cooperative localization," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2009, pp. 3415–3422.

[10] A. T. Kamal, J. A. Farrell, and A. K. Roy-Chowdhury, "Information weighted consensus filters and their application in distributed camera networks," *IEEE Trans. Autom. Control*, vol. 58, no. 12, pp. 3112–3125, Dec. 2013.

[11] G. Battistelli and L. Chisci, "Kullback–Leibler average, consensus on probability densities, and distributed state estimation with guaranteed stability," *Automatica*, vol. 50, no. 3, pp. 707–718, 2014.

[12] R. Olfati-Saber, "Kalman-consensus filter: Optimality, stability, and performance," in *Proc. 48th IEEE Conf. Decis. Control (CDC) Held Jointly 28th Chin. Control Conf.*, Dec. 2009, pp. 7036–7042.

[13] S. Wang and W. Ren, "On the convergence conditions of distributed dynamic state estimation using sensor networks: A unified framework," *IEEE Trans. Control Syst. Technol.*, vol. 26, no. 4, pp. 1300–1316, Jul. 2018.

[14] F. S. Cattivelli and A. H. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Trans. Autom. Control*, vol. 55, no. 9, pp. 2069–2084, Sep. 2010.

[15] J. Hu, L. Xie, and C. Zhang, "Diffusion Kalman filtering based on covariance intersection," *IEEE Trans. Signal Process.*, vol. 60, no. 2, pp. 891–902, Feb. 2012.

[16] G. Huang, M. Kaess, and J. J. Leonard, "Consistent unscented incremental smoothing for multi-robot cooperative target tracking," *Robot. Auto. Syst.*, vol. 69, pp. 52–67, Jul. 2015.

[17] A. Ahmad, G. D. Tipaldi, P. Lima, and W. Burgard, "Cooperative robot localization and target tracking based on least squares minimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2013, pp. 5696–5701.

[18] F. M. Mirzaei, A. I. Mourikis, and S. I. Roumeliotis, "On the performance of multi-robot target tracking," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 3482–3489.

[19] N. Atanasov, R. Tron, V. M. Preciado, and G. J. Pappas, "Joint estimation and localization in sensor networks," in *Proc. 53rd IEEE Conf. Decis. Control*, Dec. 2014, pp. 6875–6882.

[20] P. Zhu and W. Ren, "Fully distributed joint localization and target tracking with mobile robot networks," *IEEE Trans. Control Syst. Technol.*, vol. 29, no. 4, pp. 1519–1532, Jul. 2021, doi: 10.1109/TCST.2020.2991126.

[21] Y. Cao and W. Ren, "Distributed coordinated tracking with reduced interaction via a variable structure approach," *IEEE Trans. Autom. Control*, vol. 57, no. 1, pp. 33–48, Jan. 2012.

[22] T. H. Chung, J. W. Burdick, and R. M. Murray, "A decentralized motion coordination strategy for dynamic target tracking," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2006, pp. 2416–2422.

[23] R. Olfati-Saber and P. Jalalkamali, "Coupled distributed estimation and control for mobile sensor networks," *IEEE Trans. Autom. Control*, vol. 57, no. 10, pp. 2609–2614, Oct. 2012.

[24] A. W. Stroupe and T. Balch, "Value-based action selection for observation with robot teams using probabilistic techniques," *Robot. Auto. Syst.*, vol. 50, nos. 2–3, pp. 85–97, Feb. 2005.

[25] K. Zhou and S. I. Roumeliotis, "Multirobot active target tracking with combinations of relative observations," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 678–695, Aug. 2011.

[26] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Information acquisition with sensing robots: Algorithms and error bounds," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 6447–6454.

[27] B. Charrow, N. Michael, and V. Kumar, "Cooperative multi-robot estimation and control for radio source localization," *Int. J. Robot. Res.*, vol. 33, no. 4, pp. 569–580, Apr. 2014.

[28] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, and G. J. Pappas, "Anytime planning for decentralized multirobot active information gathering," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 1025–1032, Apr. 2018.

[29] K. Hausman, J. Müller, A. Hariharan, N. Ayanian, and G. S. Sukhatme, "Cooperative multi-robot control for target tracking with onboard sensing," *Int. J. Robot. Res.*, vol. 34, no. 13, pp. 1660–1677, 2015.

[30] F. Morbidi and G. L. Mariottini, "Active target tracking and cooperative localization for teams of aerial vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 5, pp. 1694–1707, Sep. 2013.

[31] F. Meyer, H. Wymeersch, M. Frohle, and F. Hlawatsch, "Distributed estimation with information-seeking control in agent networks," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 11, pp. 2439–2456, Nov. 2015.

[32] S. Su, P. Zhu, and W. Ren, "An optimization approach to fully distributed active joint localization and target tracking in multi-robot systems," in *Proc. Amer. Control Conf. (ACC)*, Jun. 2022.

[33] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proc. Amer. Control Conf.*, vol. 4, Jun. 1997, pp. 2369–2373.

[34] S. Julier and J. K. Uhlmann, "General decentralized data fusion with covariance intersection," in *Handbook of Multisensor Data Fusion*. Boca Raton, FL, USA: CRC Press, 2017, pp. 339–364.

[35] W. Niehsen, "Information fusion based on fast covariance intersection filtering," in *Proc. 5th Int. Conf. Inf. Fusion*, vol. 2, Jul. 2002, pp. 901–904.

Shaoshu Su received the B.S. degree in aerospace engineering from the Beijing Institute of Technology, Beijing, China, in 2019, and the M.S. degree in electrical engineering from the University of California, Riverside, CA, USA, in 2022.

His research interests include multiagent systems, motion planning, state estimation, and SLAM.

Pengxiang Zhu received the B.Eng. and M.S. degrees in control science and engineering from Harbin Engineering University, Harbin, China, in 2013 and 2016, respectively, and the Ph.D. degree in electrical engineering from the University of California at Riverside, Riverside, CA, USA, in 2022.

His current research interests include state estimation, sensor fusion, and visual-inertial navigation for autonomous robots.

Wei Ren (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Brigham Young University, Provo, UT, USA, in 2004.

He is currently a Professor with the Department of Electrical and Computer Engineering, University of California at Riverside, Riverside, CA, USA. His research focuses on distributed control of multiagent systems and autonomous control of unmanned vehicles.

Dr. Ren was a recipient of the National Science Foundation CAREER Award in 2008 and the IEEE Control Systems Society Antonio Ruberti Young Researcher Prize in 2017. He is currently an IEEE Control Systems Society Distinguished Lecturer.