

Distributed Information-Based Source Seeking

Tianpeng Zhang¹, Graduate Student Member, IEEE, Victor Qin², Yujie Tang³, Member, IEEE, and Na Li¹, Senior Member, IEEE

Abstract—In this article, we design an information-based multi-robot source seeking algorithm where a group of mobile sensors localizes and moves close to a single source using only local range-based measurements. In the algorithm, the mobile sensors perform source identification/localization to estimate the source location; meanwhile, they move to new locations to maximize the Fisher information about the source contained in the sensor measurements. In doing so, they improve the source location estimate and move closer to the source. Our algorithm is superior in convergence speed compared with traditional field climbing algorithms, is flexible in the measurement model and the choice of information metric, and is robust to measurement model errors. Moreover, we provide a fully distributed version of our algorithm, where each sensor decides its own actions and only shares information with its neighbors through a sparse communication network. We perform extensive simulation experiments to test our algorithms on large-scale systems and implement physical experiments on small ground vehicles with light sensors, demonstrating success in seeking a light source.

Index Terms—Consensus control, decentralized control, gradient methods, mobile robots, multi-robot systems, sensor placement.

I. INTRODUCTION

MULTIAGENT source seeking is a robotics task that uses autonomous vehicles with sensors to locate a source of interest whose position is unknown. The source of interest can be a light source [1], a radio signal transmitter [2], or a chemical leakage point [3]. The source seeking vehicles, or mobile sensors, can measure the source's influence on the environment and use this information to locate the source.

A large body of source seeking research investigates field climbing methods [4], [5], [6], [7], [8], [9]. Assuming the source signal gets stronger as the sensor–source distance shortens, the mobile sensors can “climb” the source signal field to physically

approach the source. These methods do not require explicit knowledge of the measurement model, making them easy to implement for different applications. However, field climbing methods are not necessarily the most effective source seeking methods. First, they only exploit local information of the source field, with the typical requirement that sensors must maintain a tight formation to make a reasonable ascent direction estimate, as is the case in [6] and [7]. Furthermore, the sensors cannot move too fast as a group for the measurement value to increase stably.

An alternative approach is to perform source identification/localization using various estimation methods, such as extended Kalman filter (EKF) [10], [11], particle filter (PF) [12], [13], and so on, to estimate the source location over time. These methods enable the fusion of measurements from multiple sensors to identify a global view of the measurement field. A central theme in this line of work is to improve the estimation through sensor movements. Many have proposed to move the sensors to optimize specific information metrics, in particular, variants of Fisher information (FIM) measures [2], [3], [11], [12], [14], [15], [16], [17], which relate closely to the famous Cramér–Rao lower bound (CRLB) [18], [19]. However, these studies typically focus on deriving closed-form solutions of optimal sensor placement for particular types of measurement models [2], [12], [16]. Their results are usually not generalizable or robust to modeling error, and closed-form solutions might not exist for many general measurement models.

This article draws advantages from the abovementioned methods (field climbing and source localization) to develop multi-robot source-seeking algorithms. In particular, we assume that the measurement model is available and propose an algorithm using range-based measurements. Each iteration of the algorithm consists of following three steps.

- 1) Collect range-based measurements.
- 2) Perform source location estimation.
- 3) Move along the negative gradient direction of information-based loss function L , defined as the trace of the inverse of FIM.

In particular, Step 3) seeks to increase the information about the source location in the measurement rather than looking for stronger source signals as in field climbing. Therefore, we name our method *information-based source seeking*.

Our contributions: We first introduce our information-based method in the centralized setting (Section III), where we formally define the loss function L and the three-step outline of our method. We also provide theoretical justifications for the choice of loss function L , as minimizing L improves the estimation and

Manuscript received 7 December 2022; revised 2 May 2023; accepted 17 August 2023. Date of publication 13 September 2023; date of current version 6 December 2023. This paper was recommended for publication by Associate Editor R. Tron and Editor P. R. Giordano upon evaluation of the reviewers' comments. This work was supported in part by the AFOSR YIP, under Grant FA9550-18-1-0150, in part by ONR YIP under Grant N00014-19-1-2217, and in part by NSF CNS under Grant 2003111. An earlier version of this paper was presented in part at the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) [DOI: 10.1109/IROS51168.2021.9636841]. (Correspondence author: Tianpeng Zhang.)

Tianpeng Zhang and Na Li are with the Harvard School of Engineering and Applied Sciences, Cambridge, MA 02138 USA (e-mail: tzhang@g.harvard.edu; nali@seas.harvard.edu).

Victor Qin is with the MIT School of Engineering, Cambridge, MA 02139 USA (e-mail: victorqi@mit.edu).

Yujie Tang is with the Department of Industrial Engineering and Management, Peking University, Beijing 100871, China (e-mail: yujietang@pku.edu.cn).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TRO.2023.3309099>.

Digital Object Identifier 10.1109/TRO.2023.3309099

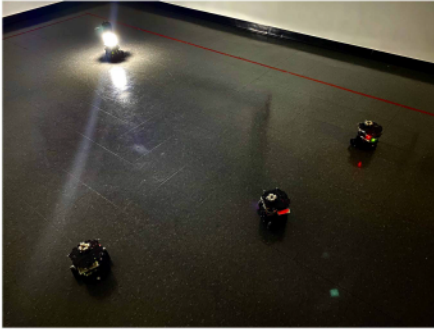


Fig. 1. Snapshot of seeking a light source in a dark room with three mobile light sensors.

gets the sensors close to the source location. Then, we extend our method to the distributed setting (Section IV), which makes our method scalable in the number of sensors. We show how distributed estimation and gradient calculation can be effectively done using simple consensus schemes. In Section V, we conduct extensive numerical experiments to study the performance of our method in both centralized and distributed settings. The results show that our method outperforms field climbing methods and is flexible and robust in multiple aspects. Finally, we implement our algorithm on small ground vehicles carrying light sensors to seek a light source in a dark room (Section V-E), as shown in Fig. 1. The hardware implementation further demonstrates the effectiveness of our algorithm.

The advantages of our methods can be summarized as follows.

- 1) Compared with field climbing algorithms, numerical studies quantitatively demonstrate that our algorithm converges much faster to the source and performs more consistently over repeated trials. See Section V-A2. The algorithm takes advantage of the sensing capacity of multisensors in the sense that the performance improves as the number of sensors increases.
- 2) Our algorithm, especially its gradient-guided movement, provides flexibility in handling various measurement models and picking different information metrics as loss functions. Section V-A3 confirms at least three applicable metrics.
- 3) The algorithm is more robust to modeling error than the source localization with stationary sensors, see Section V-A4.
- 4) Our distributed algorithm achieves comparable performance as the centralized algorithm. See Section V-B. Moreover, results show that the distributed algorithm is more robust than the centralized implementation to the error in initial guesses and communication delay. See Sections V-C and V-D.

Compared with [20], besides including more technical details for the performance of the methods, this article extends the centralized algorithm in [20] to the distributed setting, where a central controller is absent, and the sensors decide their actions individually, as described in Section IV. Numerical results in Sections V-B–V-D provide detailed comparisons between the centralized and distributed algorithm. Moreover, the hardware

implementation migrates from the centralized robotics platform in the previous work to a distributed robotics platform in this article, with the programming environment switching from ROS 1 to ROS 2. Finally, although the algorithms are designed following a rigorous theoretical framework of information and optimization methods, the global theoretical convergence remains an open question because of the nonlinearity and non-convexity associated with the problem. The proof of global convergence is, therefore, left for future work.

A. Related Work

Field Climbing Methods: Field climbing originates from scientific studies of animal behavior in exploring nutrition or chemical concentration fields [21], [22]. The studies inspire source seeking methods that use field value measurements to estimate the field gradient and apply formation control to climb along the gradient [5], [6]. Gradient-free field climbing methods are also studied in the literature: Moore and Canudas-de Wit [7] maintained the sensors in a circular formation and used measured field values as directional weights to guide the overall movement. Later works such as [8] and [9] extend the algorithms above to the distributed setting by applying consensus in the calculation of ascent direction. Single-agent field climbing is also possible if a gradient ascent control law is combined with proper zeroth-order gradient estimation algorithms, for example, using the control framework in [23].

The main differences between field climbing and our algorithm are as follows.

- 1) Field climbing maximizes the source field value directly, whereas our algorithm exploits the FIM, an indicator of both estimation accuracy and source–sensor distance
- 2) Field climbing does not assume a given measurement function, but ours does
- 3) Field climbing algorithms often require a tight sensor formation for a stable field ascent, which only exploits local information, whereas our algorithm uses the sensors to collect global information for source localization.

We will show in Section V that, unlike field climbing, the sensors under our algorithm tend to spread out so that measurements contain more diverse information about the source location.

Source Localization and Optimal Sensor Placement: If a measurement model is available, the source location can be estimated using methods, including EKF [10], [11], [24], [25], PF [12], [13], and so on. It is even possible to reconstruct the entire source field [3], [26]. Such estimation of source location is also known as source localization. An important problem in source localization is how to improve estimation via sensor movements. This problem is often investigated under the optimal sensor placement framework, in which sensors move to optimize various information metrics, including covariance [27], [28], mutual information [13], [29], and FIM measures [2], [3], [11], [12], [14], [15], [16], [17], [24], [25], such as the determinant of FIM, the largest eigenvalue of its inverse, and the trace of its inverse (D-, E-, and A-optimality criterion, respectively).

Our method also employs FIM but is different from the above works in four major ways, given as follows.

- 1) Most of the previous works focus on deriving closed-form solutions of optimal sensor placement for a particular type of measurement model, for example, RSS [2], pollutant diffusion [12], and gamma camera [16]. Our focus is not on providing closed-form solutions. Instead, we provide a gradient-based method applicable to a large class of range-based measurement models. This method also provides flexibility in choosing different information metrics as loss functions.
- 2) Many previous works are about finding the optimal angular placement of the sensors at a fixed distance to the source or in a restricted area [2], [11], [14], [17]. In contrast, we allow the sensors to move freely and eventually reach the source.
- 3) Some studies relax the restrictions on the sensor movement [15], [16]. However, these methods produce a spiraling sensor movement that is inefficient for source seeking purposes. Our method does not produce such movement.
- 4) The line of work by the authors in [24] and [25] also allows the sensors to move freely, but the control framework is very different from our work. The proposed ergodic control, which involves improving a control objective in integral form, is conceptually much more complex than our method—a gradient-based control. Ultimately, the ergodic control is designed for a very general set of mobile sensor applications, while our work is catered for the source seeking problem.

Bayesian Inference and Optimization. The recent advances in Bayesian learning have inspired many source seeking studies to adopt the Bayesian methods [30], [31], [32]. These studies view the environment as a field characterized by an (unknown) density function related to measurement and use a Gaussian process or other likelihood models as a surrogate to guide the sensor movements for new measurement collections. The computation (for running the posterior update and Bayesian optimization) and memory (for storing historical measurements as in nonparametric Bayesian methods) demands of these methods are usually much higher than those of our method. Overall, Bayesian-based methods and our method are designed from different principles. A detailed comparison is left for future work.

Finally, it is worth mentioning that our work is significantly inspired by [11], which studies the optimal sensor placement problem on a surveillance boundary. We leverage the ideas in [11] to introduce the FIM in our objective. However, we change the objective from maximizing the determinant of FIM to minimizing the trace of its inverse. We also generalize the measurement model to fit into our experiments and other real-world measurement settings. To handle such more general measurement models, we let the sensors only compute the gradient of the loss function rather than solving for its optimum at each time step.

II. PROBLEM STATEMENT

Consider the problem of using a team of mobile sensors to find a source whose position is unknown. The source can be any object of interest, such as a lamp in a dark room. Mobile sensors

are robotic vehicles that can measure the influence of the source on the environment, such as small ground vehicles carrying light sensors.

Specifically, we use $q \in \mathbb{R}^k$ to denote the source position, and use $p_1, p_2, \dots, p_m \in \mathbb{R}^k$ to denote the sensor positions, where m is the number of sensors and k is the spatial dimension. The measurement y_i made by the i th sensor is modeled by

$$y_i = h_i(p_i, q) + \nu_i \quad (1)$$

where $h_i : \mathbb{R}^k \times \mathbb{R}^k \rightarrow \mathbb{R}$ is a known continuously differentiable function and ν_i is the measurement noise. The value of h_i depends on the position of sensor i but not on other sensors, and different sensors may have different h_i . We let $y = [y_1, y_2, \dots, y_m]^\top$ denote the vector of all measurement values, let $\nu = [\nu_1, \nu_2, \dots, \nu_m]^\top$ be the noise vector, and use $p = [p_1^\top, p_2^\top, \dots, p_m^\top]^\top$ to denote the joint location of all mobile sensors.

We let $H : \mathbb{R}^{mk} \times \mathbb{R}^k \rightarrow \mathbb{R}^m$ be the mapping that describes the joint measurement made by all the sensors, i.e.,

$$H(p, q) = \begin{bmatrix} h_1(p_1, q) \\ \vdots \\ h_m(p_m, q) \end{bmatrix}. \quad (2)$$

We define the global measurement model as

$$y = H(p, q) + \nu. \quad (3)$$

This model relates all measurements to the source location and all sensor locations.

The source seeking objective is to have at least one of the sensors get within ϵ_0 distance to the source. In other words, we want to achieve

$$\|p_i - q\| \leq \epsilon_0, \exists i \in \{1, 2, \dots, m\} \quad (4)$$

where ϵ_0 is some small positive number.

III. INFORMATION-BASED SOURCE SEEKING

This section presents the key components of our information-based source seeking, including the three-step algorithmic flow and the loss function design. We assume that a central server gathers all sensor locations p and measurements y and decides on new locations for all the sensors. We will remove this assumption in Section IV, where we introduce the distributed setting.

A. Algorithm

Our information-based source seeking consists of three consecutive steps in one iteration: measurement, source location estimation, and sensor movement. The algorithm is illustrated in Fig. 2 and detailed in Algorithm 1.

1) *Measurement (Lines 2 and 3):* The mobile sensors report their locations $\{p_i\}_{i=1}^m$ and latest measurements $\{y_i\}_{i=1}^m$ to the central server, forming the location vector p and the measurement vector y .

2) *Source Location Estimation (Line 4):* Using the information available (i.e., p and y), the location estimation algorithm E generates the estimated source location \hat{q} .

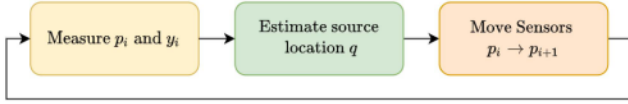


Fig. 2. Three consecutive steps in information-based source seeking.

Algorithm 1: Information-Based Source Seeking.

Input: Small constant $\epsilon_0 > 0$, the location estimator $E : (y, p) \mapsto \hat{q}$, motion planner $MP : (\tilde{p}_i(0), \tilde{p}_i(1), \dots, \tilde{p}_i(T)) \mapsto (u_i(1), \dots, u_i(T))$.

- 1: **repeat**
- 2: Get sensor locations p_i from all mobile sensors i , forming $p = [p_1^\top, p_2^\top, \dots, p_m^\top]^\top$.
- 3: Get measurement y_i from all mobile sensors i , forming $y = [y_1, y_2, \dots, y_m]^\top$.
- 4: Estimate the location of the source by $\hat{q} \leftarrow E(y, p)$.
- 5: Set the initial waypoints $\tilde{p}(0) \leftarrow p$.
- 6: **for** $t = 1$ **to** T **do**
- 7: $\tilde{p}(t+1) \leftarrow \tilde{p}(t) - \alpha_t M_t \nabla_p L|_{\tilde{p}(t), \hat{q}}$.
- 8: **end for**
- 9: Extract waypoints $(\tilde{p}_i(t))_{t=0}^T$ from $(\tilde{p}(t))_{t=0}^T$ for mobile sensor i , and generate the control inputs $(u_i(1), \dots, u_i(T)) \leftarrow MP(\tilde{p}_i(0), \dots, \tilde{p}_i(T))$.
- 10: Each mobile sensor i executes control input $u_i(1)$.
- 11: **until** $\min_{i=1,2,\dots,m} \{ \|p_i - \hat{q}\| \} \leq \epsilon_0$

The EKF is employed as the estimation algorithm in our implementations. We define $z := [q^\top, v^\top]^\top$ as the source state, where v is the velocity of the source. We do not assume any prior knowledge about the motion of the source, except that it evolves according to the second-order dynamics defined by

$$z := \begin{bmatrix} q \\ v \end{bmatrix}$$

$$z^+ = f(z) := \begin{bmatrix} q + v \\ v \end{bmatrix}. \quad (5)$$

Equation (5) models the source moving at some unknown but constant velocity v . The applicability of this motion model is backed by extensive practical evidence in GPS problems [33], [34] and moving target tracking [35]. In general, it is sensible to make a minimal assumption as above on the motion model in source localization. While the actual dynamics of the source might deviate from the model in (5)—for example, in the experiments shown in Fig. 4—the localization performance typically remains satisfactory. Analogous to (5), another prevalent assumption is to define the state $z := q$ so that the transition model is $z^+ = f(z) := z$ (see the third paragraph of [24, Sec. IV-B]), but we observe that the estimation from this model often fails to converge to the true source location. In comparison, estimation with the model in (5) converges to the source location much more consistently.

At each time step, the EKF takes in the sensor measurements and sensor locations and returns an estimate \hat{z} of the true source state z . See [36, Def. 3.1] for details about the EKF update.

3) *Sensor Movement (Lines 5–10)*:: The sensors move along the gradient descent directions of the following loss function:

$$L(p, q) = \text{Tr} \left[(\nabla_p H(p, q) \cdot \nabla_q H(p, q)^\top)^{-1} \right]. \quad (6)$$

We will explain the motivation of this loss function later in Section III-B, by relating it to the FIM matrix and the CRLB [18], [19]. Sensor movement includes two parts: waypoint planning and waypoint tracking.

a) *Waypoint planning*: In Lines 5–8, with the source location estimate \hat{q} obtained in Line 4, we generate a set of waypoints $\tilde{p}(0), \dots, \tilde{p}(T)$ by applying gradient descent on L with respect to p , with $q = \hat{q}$ fixed. In Line 7, $\alpha_t > 0$ is the step size and $M_t \succ 0$ is a directional regularization matrix.

b) *Waypoint tracking*: In Lines 9 and 10, after the joint waypoints $\tilde{p}(0), \dots, \tilde{p}(T)$ have been calculated by the gradient descent on L , we extract the waypoints $\tilde{p}_i(0), \dots, \tilde{p}_i(T)$ for each mobile sensor i , and use the motion planner MP to calculate a sequence of control inputs $u_i(1), \dots, u_i(T)$ and apply the control at the first instance $u_i(1)$ to the corresponding mobile sensor.¹

Remark 1: Note the algorithm aims to minimize $L(p, \hat{q})$ rather than $L(p, q)$ (p being the decision variable), which raises the question of whether improving the former leads to a decrease in the value of the latter. Intuitively, it is expected that if \hat{q} and q are sufficiently close to each other, i.e., the estimation error is small, then changing p to reduce the value of $L(p, \hat{q})$ will make $L(p, q)$ decrease. These two algorithm components rely on each other to function as a whole.

B. Information-Based Loss Function for Sensor Movement

One naive approach to determining sensor movement after Step 2) could be heading directly toward the estimated location. However, we observed that such movement often makes sensors cluster tightly, which reduces the diversity in measurements and is undesirable. As we later show in Figs. 3(c) and 4(c), the resulting estimation and seeking are unstable.

Creating a good condition for estimation through sensor movements is crucial for successful source seeking. Therefore, instead of following the haphazard approach above, our algorithm takes the gradient steps in Step 3). The idea is to let the sensors maximize some information metrics related to the source location. Specifically, we consider the FIM about the source location defined as follows: let y and p be the joint measurement and joint position of *all* the sensors, and H be the corresponding global measurement function, as in (3). We treat y as a random vector, with joint probability density $f(y; q)$ parameterized by the true source location q . The FIM about q is considered to measure the amount of information about the

¹This is inspired by the framework of receding horizon control/model predictive control [37] that works well when the tracking trajectory is time-varying. For more information about motion planning, refer to Appendix A.

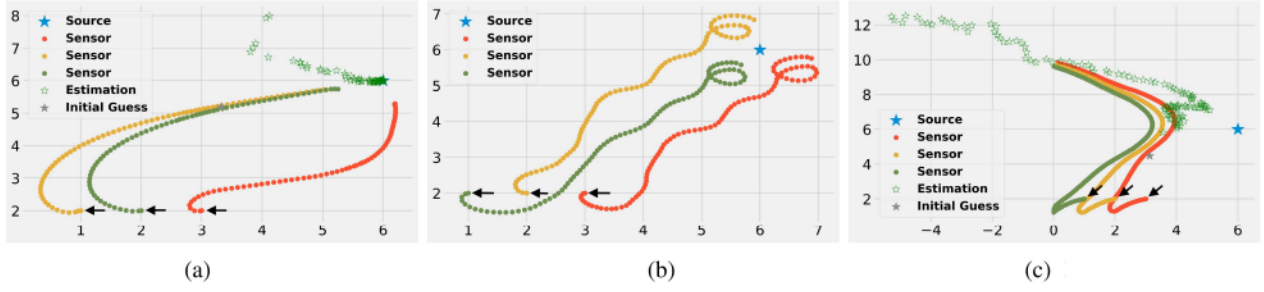


Fig. 3. General behaviors of seeking a stationary source. The black arrows indicate the starting locations of the sensors. The sensors in (b) do not make any estimations, since field climbing only uses measured signal strength to guide sensor movements. (a) Information-based seeking1. (b) Field climbing [7]. (c) Following straight lines to the estimated location.

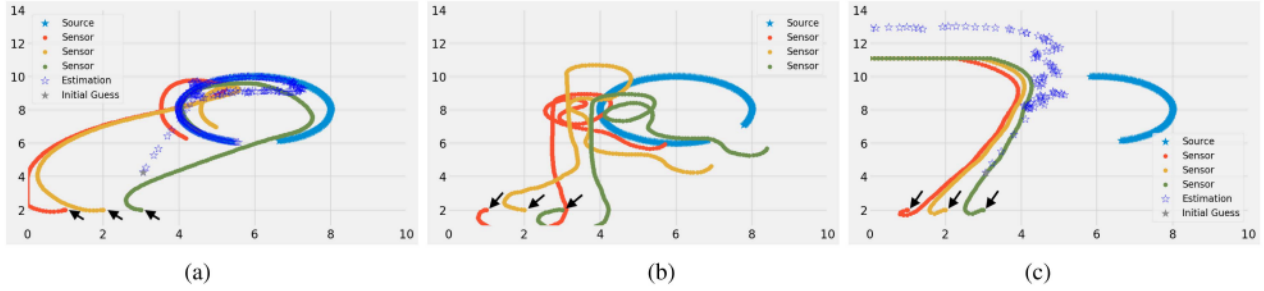


Fig. 4. General behaviors of seeking a moving source. The black arrows indicate the starting locations of the sensors. (a) Information-based seeking1. (b) Field climbing [7]. (c) Following straight lines to the estimated location.

unknown q contained in measurement y . It is defined as

$$\text{FIM}_y(q) := \text{Cov}[\nabla_q \log f(y; q) | q]. \quad (7)$$

FIM is closely connected with estimation quality, as it is a lower bound on the covariance of unbiased estimators. This result is formally known as the CRLB.

Theorem 1 (CRLB [18], [19]): For any unbiased estimator \hat{q} of q , the following matrix inequality holds:

$$\mathbb{E}[(\hat{q} - q)(\hat{q} - q)^\top] \succeq \text{FIM}^{-1}. \quad (8)$$

As a special case, if we assume y follows multivariate Normal distribution described by

$$y = H(q, p) + \nu, \nu \sim \mathcal{N}(0, R) \quad (9)$$

then the FIM takes the form of

$$\text{FIM}_y(q) = 4\nabla_q H(q, p)R^{-1}\nabla_q H(q, p)^\top. \quad (10)$$

Under the further assumption that ν is independent identically distributed (i.i.d.). Normal($R \propto I$), there is

$$\text{FIM} \propto \nabla_q H(p, q) \cdot \nabla_q H(p, q)^\top. \quad (11)$$

Motivated by the form of FIM in (11), we use

$$L(p, q) = \text{Tr} \left[(\nabla_q H(p, q) \cdot \nabla_q H(p, q)^\top)^{-1} \right] \quad (12)$$

as the loss function. Note that by minimizing (12), the CRLB is driven closer to the zero matrix. Furthermore, if H satisfies Assumption 1 below, then minimizing L also results in approaching the source as stated in Propositions 1 and 2 afterward.

Assumption 1: We make the following assumptions on the measurement functions h_i .

- 1) **Isotropic Measurement:** The measurement values depend only on source–sensor distance, i.e.,

$$h_i(p_i, q) = g_i(\|p_i - q\|) = g_i(r_i) \quad (13)$$

for some function $g_i : (0, +\infty) \rightarrow \mathbb{R}$, where $r_i := \|p_i - q\|$.

- 2) **Monotonicity:** The absolute value of the derivative of each function, $|g'_i(r)|$, is monotonically decreasing in r . Here, $g'_i(r)$ is the derivative of g_i with respect to r .
- 3) **Nondegeneracy:** Let \hat{r}_i denote the unit direction vector from the source to the i th mobile sensor, i.e., $\hat{r}_i = (p_i - q)/\|p_i - q\|$. We assume $\sum_{i=1}^m \hat{r}_i \hat{r}_i^\top \succ 0$.

Proposition 1 (Reaching the source I): Under Assumption 1, we have

$$\frac{1}{m \cdot L(p, q)} \leq \max_i |g'_i(r_i)|^2 \quad (14)$$

which means that if $L(p, q)$ decreases, then $\max_i |g'_i(r_i)|$ tends to increase and consequently $\min_i r_i$ tends to decrease.

Proposition 2 (Reaching the source II): Assume $|g'_i(r_i)|$ is strictly monotone in r_i in Assumption 1. Then, if the sequence of sensor locations $\{p(t) : t = 1, 2, 3, \dots\}$ converges and satisfies

$$\lim_{t \rightarrow \infty} L(p(t), q) = \inf_{p: p_i \neq q, \forall i} L(p, q)$$

then there must be

$$\lim_{t \rightarrow \infty} \min_i \|p_i(t) - q\| = 0.$$

Please see Appendix B for the proof of Propositions 1 and 2.

In Proposition 2, we assume that the $p(t)$ sequence converges for technical convenience, but it is also reasonable to assume

the sensors will reach a stationary formation when they have minimized L . Also, the infimum of L is taken over the set $\{\mathbf{p} : p_i \neq q \forall i\}$ because the FIM could be undefined if $p_i = q$ for some i .

Remark 2: Assumption 1.3 is a regularity condition that ensures L is well defined, which is essential for analyzing the relationship between L and the source-sensor distance. We enforce this assumption in the implementations by adding a positive definition matrix δI , with $\delta > 0$ being a small constant, to the estimated FIM when calculating L and its gradient. This technique proves to eliminate most of the pathological numerical behaviors while maintaining the soundness of source-seeking performance.

Remark 3: Despite the properties demonstrated in Propositions 1 and 2, showing the full convergence of Algorithm 1 remains an open theoretical question. We leave this question to future work, but a heuristic justification for the convergence behavior is that climbing the information field and getting more accurate estimations of the source location complement each other (see Remark 1), so the L value decreases over time. Therefore, the sensors get close to the source by the propositions above.

IV. DISTRIBUTED INFORMATION-BASED SOURCE SEEKING

In this section, we present our distributed source seeking algorithm. We no longer assume there is a central controller. Instead, the mobile sensors decide on their actions individually. Meanwhile, the sensors can communicate through a network to exchange information about measurements, sensor positions, and estimations. We model the communication network by a directed graph, denoted by $G = (\mathcal{N}, \mathcal{E})$, where \mathcal{N} is the set of nodes(sensors) and \mathcal{E} is the set of edges. If an edge $(i, j) \in \mathcal{E}$, then sensor j can directly receive information from sensor i . Let $\mathcal{N}_j = \{i | (i, j) \in \mathcal{E}\} \cup \{j\} = \{j, i_1, i_2, \dots, i_{m_j}\}$ be the in-coming neighborhood of j , including j itself. Here, m_j denotes the number of its incoming neighbors except itself. For each $(i, j) \in \mathcal{E}$, there is a consensus weight $w_{ji} > 0$ satisfying $\sum_{i \in \mathcal{N}_j} w_{ji} = 1$ for all j (see [38, Assumptions 2.1–2.3] for conditions on the consensus weights).

We assume the information available to sensor j includes those from its measurements and direct communication. In particular, mobile sensor j knows the following information.

- 1) The measurement functions in its neighborhood, or equivalently the local joint measurement function H_j

$$H_j(q, p_j, p_{i_1}, \dots, p_{i_{m_j}}) := \begin{bmatrix} h_j(q, p_j) \\ h_{i_1}(q, p_{i_1}) \\ \vdots \\ h_{i_{m_j}}(q, p_{i_{m_j}}) \end{bmatrix}. \quad (15)$$

- 2) The joint position and joint measurement of its neighborhood (in bold font).

$$\mathbf{p}_j := [p_j^\top, p_{i_1}^\top, \dots, p_{i_{m_j}}^\top]^\top, \mathbf{y}_j := [\mathbf{y}_j^\top, \mathbf{y}_{i_1}^\top, \dots, \mathbf{y}_{i_{m_j}}^\top]^\top. \quad (16)$$

Algorithm 2: Distributed Information-Based Source Seeking (for sensor j).

Input: Consensus weights $\{w_{jk}\}_{k \in \mathcal{N}_j}$, consensus-based location estimator E , motion planner MP , termination threshold ϵ_0 .

1: Initialize $\hat{\mathbf{z}} := [\hat{q}_j, \hat{v}_j]^\top, \hat{F}_j$.

2: **repeat**

3: Take measurements and communicate with neighbors to obtain $\mathbf{p}_j, \mathbf{y}_j, \{\hat{\mathbf{z}}_k, \hat{F}_k\}_{k \in \mathcal{N}_j}$.

4: Estimate the source location using consensus-based Kalman Filter as shown in (20),

$$\hat{q} \leftarrow E(\mathbf{y}_j, \mathbf{p}_j, \{\hat{\mathbf{z}}_k, w_{jk}\}_{k \in \mathcal{N}_j}).$$

5: Record F_j in the previous step $F_j^- \leftarrow F_j$

6: Compute the new partial FIM

$$F_j \leftarrow \nabla_q h_j(p_j, \hat{q}) \nabla_q h_j(p_j, \hat{q})^\top$$

7: Estimate the global FIM

$$\hat{F}_j \leftarrow \left[\sum_{k \in \mathcal{N}_j} w_{jk} \hat{F}_k \right] + F_j - F_j^-$$

8: Set the initial waypoint $\tilde{p}_j(0) \leftarrow p_j$.

9: Calculate waypoints $\tilde{p}_j(1), \dots, \tilde{p}_j(T)$ by gradient descent

$$A_{j,t} = \nabla_q h_j(\tilde{p}_j(t), \hat{q})$$

$$\mathbf{d}_t = -2 \nabla_p A_{j,t} \hat{F}_j^{-2} A_{j,t}$$

$$\tilde{p}_j(t+1) \leftarrow \tilde{p}_j(t) - \alpha_t M_t \mathbf{d}_t$$

with $M_t \succ 0$ being regularization matrices.

10: Generate the control actions

$$(u_j(1), \dots, u_j(T)) \leftarrow MP(\tilde{p}_j(0), \dots, \tilde{p}_j(T)).$$

11: Execute control action $u_j(1)$.

12: **until** $\|\mathbf{p}_j - \hat{q}\| \leq \epsilon_0$

We assume the local joint measurement model for j is

$$\mathbf{y}_j = H_j(q, \mathbf{p}_j) + \boldsymbol{\nu}_j \quad (17)$$

where $\boldsymbol{\nu}_j = [\nu_{j1}^\top, \nu_{j2}^\top, \dots, \nu_{jw}^\top]^\top$ is a random vector following multivariate normal distribution $\mathcal{N}(0, R_j)$.

We assume the sensors act cooperatively, that is, their objective is still to have some members in the team to reach the source, as defined in (4).

A. Distributed Algorithm

In the distributed algorithm, each sensor j maintains an estimate for the source location and velocity, denoted by $\hat{\mathbf{z}}_j := [\hat{q}_j^\top, \hat{v}_j^\top]^\top$. The sensor also maintains an estimate for the global FIM, denoted by \hat{F}_j . Our source seeking algorithm requires sensor j to repeat three consecutive steps: information gathering, source location estimation, and movement. Algorithm 2 contains the detailed operations.

1) *Information Gathering (Line 3)*: Sensor j takes measurements and communicates with its neighbors to obtain the sensor positions \mathbf{p}_j and measurements \mathbf{y}_j . Sensors also communicate their source location estimates $\hat{\mathbf{z}}_j$ and their FIM estimates $\hat{\mathbf{F}}_j$, as required in the subsequent steps.

2) *Source Location Estimation (Line 4)*: The gathered information is used to form an updated source location estimate $\hat{\mathbf{q}}_j$ through *consensus-based estimation algorithms*, elaborated in Section IV-B. For ease of exposition, we assume the source location estimator to be the consensus EKF (20), although other estimators could also be adapted to the distributed setting following a similar procedure.

3) *Sensor Movement (Lines 6–11)*: Sensor j calculates the partial gradient of the loss function L (6) through *distributed information gradient update* (Lines 6–9), elaborated in Section IV-C. Then, it calls a motion planner MP to compute the corresponding control actions for the gradient steps and executes the first action (Lines 10 and 11).

The key differences between the distributed Algorithms 2 and 1 are as follows.

- 1) Algorithm 1 runs on the central controller. The distributed algorithm runs on individual sensors.
- 2) Comparing Step 1), the Algorithm 1 knows the measurements and positions of all sensors. The distributed algorithm only knows those from the neighborhood. As a result, the Algorithm 2 modifies Steps 2) and 3) to accommodate the distributed setting.

Remark 4: Showing the full convergence of Algorithm 2 also remains an open question. We leave it to future work.

Next, we elaborate on how agents perform distributed source location estimation and distributed information gradient update in Algorithm 2.

B. Distributed Source Location Estimation

In Step 2), each sensor must *locally* form a source location estimate based on the information available to it, and one simple way is to run *local EKF* based on its local information. For each sensor j , the *local EKF* maintains two running variables, $\hat{\mathbf{z}}_j$ and P_j , where

$$\hat{\mathbf{z}}_j := \begin{bmatrix} \hat{\mathbf{q}}_j \\ \hat{\mathbf{v}}_j \end{bmatrix}$$

is sensor j 's estimate of the position–velocity vector of the source, and P_j is a matrix quantifying the uncertainty in the estimate. These two variables are updated according to

$$\begin{aligned} \hat{\mathbf{z}}_j &\leftarrow f(\hat{\mathbf{z}}_j) + K_j(\mathbf{y}_j - H_j(\hat{\mathbf{q}}_j, \mathbf{p}_j)) \\ P_j &\leftarrow A_j P_j A_j^\top + Q_j - K_j(C_j P_j C_j^\top + R_j) K_j^\top \end{aligned} \quad (18)$$

where Q_j and R_j are predefined constant positive matrices, f is given in (5), and H_j is given in (15). Q_j and R_j represent our prior belief of the covariances of process and measurement noises; therefore, they are usually the same among all j . The

quantities A_j , C_j , and K_j are defined by

$$\begin{aligned} A_j &:= \nabla_z f(\hat{\mathbf{z}}_j), C_j := \nabla_z H_j(\hat{\mathbf{q}}_j) = \begin{bmatrix} \nabla_q H_j(\hat{\mathbf{q}}_j, \mathbf{p}_j) \\ 0 \end{bmatrix} \\ K_j &:= A_j P_j C_j^\top (C_j P_j C_j^\top + R_j)^{-1}. \end{aligned} \quad (19)$$

The *local EKF* may be one of the simplest estimation algorithms based on sensors' local information from neighboring sensors. However, its estimation is not great in our source seeking experiments, as we later show in Section V-B. One of the main reasons is that the communication network is typically sparse, and each sensor is connected to only a few neighbors. Therefore, the measurements available to each sensor are few, and the resulting estimation is poor.

The *consensus EKF*, an extension to the Kalman consensus filter II algorithm [39], addresses the limitations of *local EKF* by introducing a consensus procedure among the estimates. The estimation $\hat{\mathbf{z}}_j$ is updated with the following formula:

$$\begin{aligned} \hat{\mathbf{z}}_j &\leftarrow \sum_{i \in \mathcal{N}_j} w_{ji} \hat{\mathbf{z}}_i \\ &\quad + [f(\hat{\mathbf{z}}_j) + K_j(\mathbf{y}_j - H_j(\hat{\mathbf{z}}_j, \mathbf{p}_j)) - \hat{\mathbf{z}}_j] \end{aligned} \quad (20)$$

with $w_{ji} \geq 0$, $\sum_{i \in \mathcal{N}_j} w_{ji} = 1$. The update of P_j remains the same as the *local EKF*. The update in (20) can be viewed as adding a consensus term $\sum w_{ji} \hat{\mathbf{z}}_i$ to the difference between *local EKF* update and current $\hat{\mathbf{z}}_j$. Meanwhile, the *local EKF* can be viewed as a special case of (20) with $w_{jj} = 1$ and $w_{ji} = 0$ for $i \neq j$. We highlight here the advantages of the consensus EKF over the *local EKF* as follows.

- 1) Since the neighbors' estimates use the measurements from possibly outside \mathcal{N}_j , sensor j is indirectly exposed to the data outside of its immediate neighborhood through consensus.
- 2) With properly chosen consensus weights as specified in [38, Assumptions 2.1–2.3], the sensors can reach consensus on estimates much quicker than the rate of change in *local EKF* update. Effectively, the limiting estimate at consensus is an estimator that uses the measurement from all the sensors in the network. It serves as a target location for the sensor to coordinate their movement, and we observe much better convergence to the source using the consensus EKF.

Our consensus EKF algorithm is an extension of the Kalman-consensus filter II algorithm [39, Sec. IV, eq. (20)] to the non-linear measurement setting. Some related studies also employ consensus in extended Kalman-like filters [40], [41], [42], but they take consensus over both the mean estimate and covariance estimate, while our algorithm only takes consensus over the mean estimate to lower the communication load, which improves the responsiveness of source seeking in our real-time environment. This scheme is also backed by the theoretical result in [39, Th. 2] that performing consensus on the mean estimate but not on the covariance estimate can still lead to convergence in estimation. It is also worth pointing out that although our consensus EKF shares similarities with the existing literature,

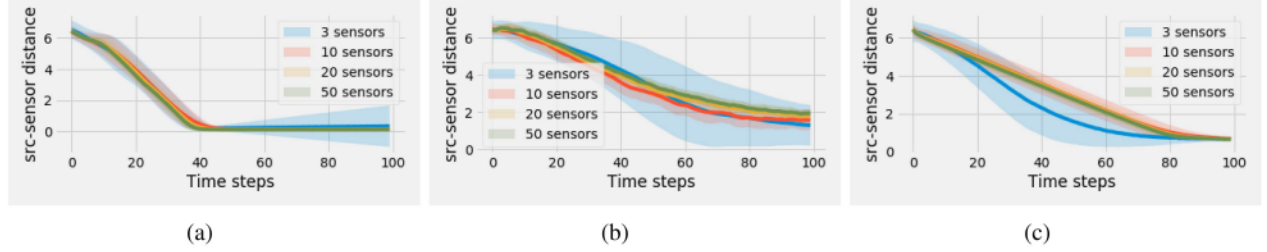


Fig. 5. Time evolution of the source-sensor distance. (a) Information-based seeking1. (b) Climbing estimated gradient [6]. (c) Circular formation field climbing [7].

our consensus-based distributed gradient update (described later in Section IV-C) is a novel contribution.

C. Distributed Information Gradient Update

Similar to the centralized setting, our distributed algorithm requires the sensors to move to new locations along the negative gradient of the FIM-based loss function L (6). However, since L depends on the global measurement function H and all agents' positions \mathbf{p} , the challenge is for individual sensors to compute the correct partial gradients in a distributed fashion. Specifically, the partial gradient of L can be shown to be

$$\nabla_{\mathbf{p}_j} L(\mathbf{p}, \mathbf{q}) = -2\nabla_{\mathbf{p}_j} A_j \text{FIM}^{-2} A_j \quad (21)$$

where $A_j = \nabla_{\mathbf{q}} h_j(\mathbf{p}_j, \mathbf{q})$. Note that both A_j and $\nabla_{\mathbf{p}_j} A_j$ can be computed using the local information of sensor j already. Therefore, the remaining problem is to estimate the value of FIM that depends on global information H and \mathbf{p} . We note that

$$\text{FIM} \propto \sum_{i=1}^m \nabla_{\mathbf{q}} h_j(\mathbf{q}, \mathbf{p}_j) \nabla_{\mathbf{q}} h_j(\mathbf{q}, \mathbf{p}_j)^\top \quad (22)$$

meaning that the FIM is proportional to the equally weighted sum of the rank-one matrices

$$F_j := \nabla_{\mathbf{q}} h_j(\mathbf{q}, \mathbf{p}_j) \nabla_{\mathbf{q}} h_j(\mathbf{q}, \mathbf{p}_j)^\top$$

which we name as *partial FIM*. Note that each F_j is again computable using the local information of sensor j . Therefore, the sensors can recover the global FIM through consensus. Specifically, let each sensor maintain a running estimate \hat{F}_j of the global FIM, and update \hat{F}_j according to the following formula:

$$\hat{F}_j^+ = \left(\sum_{i \in \mathcal{N}_j} w_{ji} \hat{F}_i \right) + F_j - F_j^-. \quad (23)$$

Here, $\{w_{ji}\}_{i \in \mathcal{N}_j}$ are consensus weights, and F_j^- denotes the partial FIM value from the previous time step. To ensure \hat{F}_j converges to an equally weighted sum of F_j 's, the consensus weights can be predefined to form a static doubly stochastic consensus matrix. Alternatively, the update in (23) can be implemented to follow the parallel two-pass algorithm [38].

Finally, sensor j calculates an estimated gradient descent direction by substituting the FIM in (21) with \hat{F}_j , as shown in Line 9 of Algorithm 2.

V. EXPERIMENTS

A. Advantage of Information-Based Source Seeking

This section compares Algorithm 1 with field climbing methods via numerical experiments, showing the advantage of information-based source seeking. All experiments are implemented in a centralized way. The algorithm performance under actual robot dynamics is studied in simulations in Section V-A1. In the subsequent studies, we remove the robot dynamics in simulations to efficiently conduct repetitive trials and assume the sensors follow the gradient steps exactly. We study the influence of the number of sensors in Section V-A2, the difference of various information metrics in Section V-A3, and the robustness to modeling error in Section V-A4.

1) *Gazebo Numerical Experiments*: The following numerical experiments are carried out using the Gazebo simulation toolbox [43], with virtual mobile sensors simulating the same dynamics as the actual robots. We generate simulated measurement values of the sensors by

$$y_i = 1/r_i^2 + \nu_i \quad (24)$$

with ν_i drawn independently from $\mathcal{N}(0, 0.01)$. The measurement function $h_i(\mathbf{p}_i, \mathbf{q}) = 1/\|\mathbf{p}_i - \mathbf{q}\|^2 = 1/r_i^2$ is given to the EKF for estimation.

Stationary source: In the first set of simulations, we use three mobile sensors to seek a stationary source. The source is fixed at position (6.0, 6.0), while the mobile sensors are initially placed at (1.0, 2.0), (2.0, 2.0), and (3.0, 2.0). The initial guess of source location given to the EKF is (3.0, 4.0). The terminal condition threshold $\epsilon_0 = 0.5$. We compare the convergence to a stationary source among three algorithms:

- our algorithm;
- the field climbing algorithm introduced by [7] that only maximizes measured signal strength;
- following straight lines to the estimated location.

The algorithm in (c) is included to show the importance of exploiting FIM in obtaining accurate estimation. The results are displayed in Fig. 3.

First, notice that the straight-line algorithm fails to converge to the source, as shown in Fig. 3(c). We suspect the reason is sensors cluster together quickly as they move to the (same) estimated location and cannot provide sufficiently rich, diverse measurements for a reasonable estimation. Consequently, the estimate gradually deviates from the source location, as do the sensors. On the other hand, if taking a trajectory that

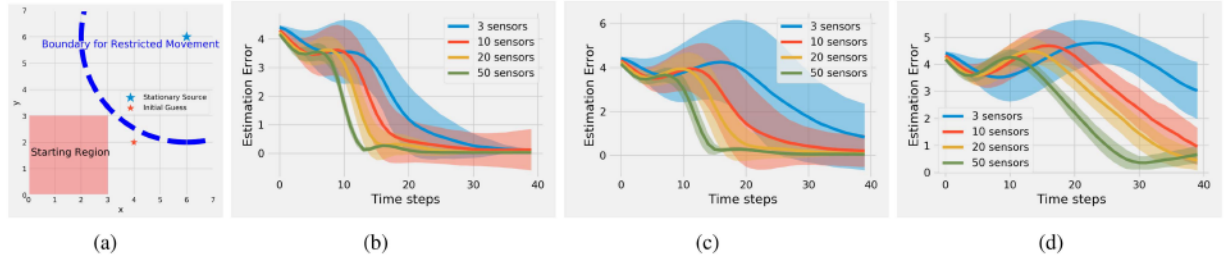


Fig. 6. Time evolution of source location estimation error. In Fig. 6(b), the variance of the 50 sensor curve after ten steps is small but not zero. (a) Experiment Setting. (b) Sensors move freely. (c) Sensors stay outside the boundary. (d) Stationary sensors.

improves the FIM, the sensors cover the space more thoroughly, resulting in a stable decrease in the estimation error and the final success of reaching the source, as shown in Fig. 3(a).

Comparing Fig. 3(a) and (b), note that sensors using our algorithm first spread out to estimate the source location better and then converge to the source, whereas sensors doing field climbing maintain a tight formation while steadily approaching the source. Since we use constant rather than diminishing step sizes for Fig. 3(b), the virtual robots do not stop completely near the source and perform a looping behavior. Although our algorithm and the field climbing algorithm [7] are both successful with a stationary source, our algorithm consistently converges faster over repetitive trials, as is shown later in Section V-A2.

Moving source: In this set of experiments, all parameters are kept the same as the stationary case, except that now the source moves in a circular motion with constant speed. See Fig. 4(a)–(c). Note that the straight-line algorithm again leads to a sensor formation that causes the estimation to deviate from the actual source location. Both our algorithm and the field climbing algorithm [7] successfully get close to the source. However, the field climbing method exhibits unnecessary irregular motion when the sensors are near the source. We suspect that the field climbing direction becomes very sensitive to the source movement as the sensors get close to the source, which leads to such motion. In comparison, sensors following our algorithm trace much more stable paths.

2) **Performance With Sensor Swarms:** We now investigate the influence of the number of mobile sensors on the algorithm performance. We perform repetitive simulations to study the convergence rate of source–sensor distances and the evolution of the estimation error. We use a stationary source fixed at position (6.0, 6.0) and randomize the initial sensor locations following uniform distribution in a 3.0-by-3.0 rectangle, as shown in Fig. 6(a).

The experiments in Fig. 5 study the convergence rate of the sensors' distances to the source. Our algorithm is compared with two field-climbing methods [6], [7]. The performances of 3-, 10-, 20-, and 50-sensor teams are plotted. The solid curves are the average values over 100 repetitive experiments. The color bands indicate one standard deviation. The results show that our algorithm converges much faster than the others. In addition, increasing the number of sensors substantially reduces all algorithms' variance while not affecting the convergence time. Because the speed of sensors has an upper bound regardless of the number of sensors, the overall convergence rate of source–sensor distance

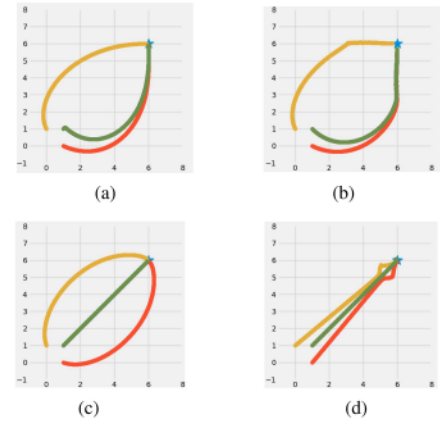


Fig. 7. Assuming the source location is known, the above are the gradient descent trajectories of the tested information metrics. (a) $\text{Tr}(\text{FIM}^{-1})$. (b) $\lambda_{\max}(\text{FIM}^{-1})$. (c) $-\log \det(\text{FIM})$. (d) Covariance.

is limited. Nevertheless, having more sensors provides more measurements, which reduces the variance and results in more consistent, stable trials. In particular, our algorithm benefits the most from having more sensors since more measurements contribute to better estimation, thus faster convergence with smaller variance.

To further test the above conjecture that sensors moving freely and in the direction of improving FIM bring richer measurement and, therefore, enhance the estimation, we conduct the following three experiments.

- 1) The sensors move freely, guided by our proposed algorithm.
- 2) The sensors are restricted to staying outside a radius of 3.0 from the source, performing projected gradient descent of our proposed loss function at the boundary.
- 3) The sensors do not move but only perform location estimation.

The results in Fig. 6 show that using more sensors and allowing sensors to move freely can both lead to a faster decline in the estimation error and a smaller variance.

3) **Comparison of Different Information Metrics:** Although we specify the loss function as the A-optimality criterion in our algorithm, one can, in principle, replace it with other alternatives. In the following experiments, we test our algorithm's performance with four different loss functions:

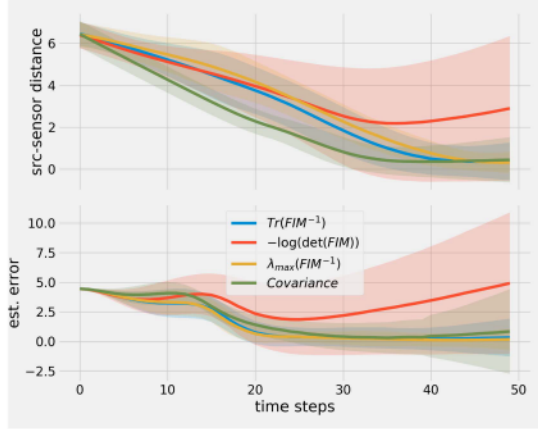


Fig. 8. Comparison of the algorithm performance using different information metrics.

- 1) $\text{Tr}(\text{FIM}^{-1})$;
- 2) $\lambda_{\max}(\text{FIM}^{-1})$;
- 3) $-\log \det(\text{FIM})$;
- 4) $\text{Tr} P$ where P is the posterior covariance of the EKF defined by [27, eq. (15)], $P = (\nabla_q H R^{-1} \nabla_q H^\top + P_0^{-1})^{-1}$, with P_0 being the current estimation covariance and R being the measurement noise parameter of the EKF.

All loss functions are tested with a stationary source and three freely moving sensors for 100 repetitive trials. Each trial is initialized randomly in the same way as in the previous section. Fig. 7 shows the gradient descent trajectories for the tested metrics. Fig. 8 shows that all the tested metrics except $-\log \det(\text{FIM})$ yield relatively good performance when using our algorithm, with $\text{Tr}(\text{FIM}^{-1})$ and $\lambda_{\max}(\text{FIM}^{-1})$ achieving a better balance in convergence and estimation than the covariance metric $\text{Tr}(P)$. These results confirm the generality of our algorithms.

Remark 5 (Rationale for metrics): Assume that the FIM is a $k \times k$ matrix. Since the FIM is always positive semidefinite, it has nonnegative eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_k$. The goal of minimizing CRLB is to make the matrix FIM^{-1} as close to the zero matrix as possible, which means $\frac{1}{\lambda_1}, \dots, \frac{1}{\lambda_k}$ should all be as close to zero as possible. Therefore, $\text{Tr}(\text{FIM}^{-1}) = \sum_{i=1}^k \frac{1}{\lambda_i}$ and $\lambda_{\max}(\text{FIM}^{-1}) = \max_{i=1, \dots, k} \frac{1}{\lambda_i}$ are both reasonable metrics to minimize. Similarly, the third metric $-\log \det(\text{FIM}) = \sum_{i=1}^k \log(\frac{1}{\lambda_i})$ also encourages an overall reduction of $\frac{1}{\lambda_i}$ values, but it may not be as good as the previous two since and its value can be made arbitrarily low by setting one single $\frac{1}{\lambda_i}$ is very close to zero while all others remain large. The covariance metric is essentially $\text{Tr}((\text{FIM} + P_0^{-1})^{-1})$, which can be thought of as a regularized version of the first metric.

Remark 6 (Geometric properties): Although the trajectories in Fig. 7 may look very different, they share some important common geometric properties that make sense intuitively. Specifically, they all encourage a separation among the sensors at some point and create an overall trend of approaching the source. We need to look at the explicit form of FIM to understand these

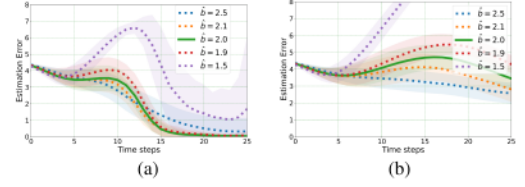


Fig. 9. Robustness. The solid lines correspond to the perfect measurement model; the dotted lines are measurement models with errors. The color bands show the standard deviation of errors across 100 repetitive trials. (a) Our Algorithm. (b) Stationary sensors.

behaviors. In the proof in Appendix B, we show that

$$\text{FIM} = \sum_{i=1}^m |g'_i(r_i)|^2 \hat{r}_i \hat{r}_i^\top$$

where m is the number of sensors, g_i is the measurement function for the i 'th sensor, r_i is the distance between the i 'th sensor and the source, and \hat{r}_i is the unit vector pointing from the source to the i 'th sensor. Having FIM^{-1} close to zero means FIM itself is “very positive definite.” To achieve that either 1) $|g'_i(r_i)|$ are large or 2) $\hat{r}_i \hat{r}_i^\top$ are “very linearly-independent,” or both. Under the assumption that $|g'_i(r_i)|$ becomes larger as r_i decreases, item 1) contributes to the overall trend of approaching the sensor, which seeks a stronger signal-to-noise ratio. Further, item 2) encourages the unit vectors \hat{r}_i to point toward different directions, making the sensors separate from one another and creating different line-of-sights.

The covariance is the one that stands out the most among the metrics as it exhibits some follow-straight-line behavior at the beginning and separates only when close to the source. This phenomenon can be explained by the dominance battle between P_0^{-1} and $\nabla H R^{-1} \nabla H^\top$, as discussed in detail in Appendix D.

4) Robustness to Measurement Modeling Error: We next investigate whether our algorithm can function despite the error in the measurement model. We simulate source seeking with ten mobile sensors and a stationary source, in which the measurement is generated by $y = 1/r^2 + \nu$, $\nu \stackrel{\text{i.i.d.}}{\sim} \mathcal{N}(0, 0.01)$. We provide imperfect measurement models to the EKF in the form of $h(r) = 1/r^{\hat{b}} = 1/r^{2+\Delta b}$, with Δb taking values in $0, \pm 0.1, \pm 0.5$. We study the estimation error in two settings: 1) the sensors move freely using our algorithm; 2) the sensors are stationary. The results are shown in Fig. 9.

The robustness of our algorithm is twofold: In Fig. 9(a), when compared with using the perfect measurement model ($\Delta b = 0$), our algorithm shows no significant degradation in estimation when Δb is small, maintaining reasonable estimation quality despite imperfect measurement models. Besides, when comparing Fig. 9(a) and (b), our algorithm shows more robustness than stationary sensors, whose estimations tend to diverge when Δb is at the value of 0.5.

B. Effectiveness of Our Distributed Algorithm

In this section, we showcase the effectiveness of our algorithm when extended to the distributed setting. In

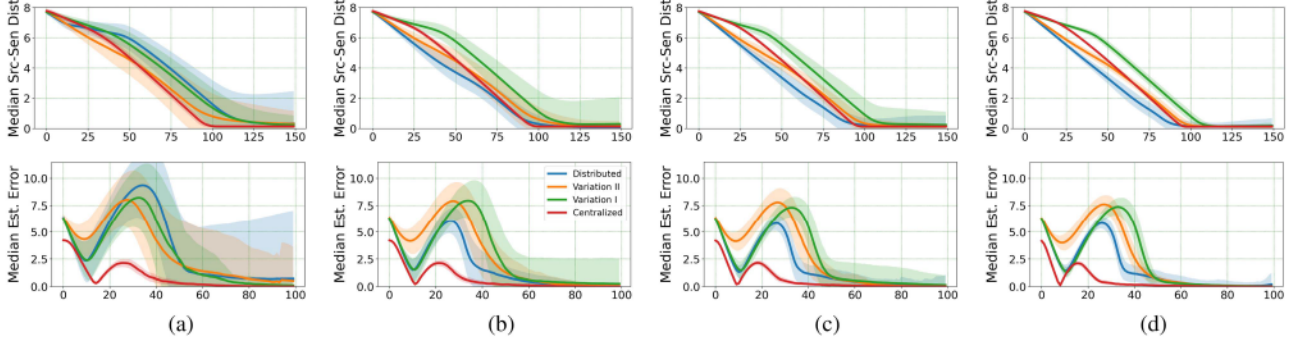


Fig. 10. Comparison of the time evolution of sensor-source distance and estimation error for different variations, with the additional case where four sensors are used. The x -axes correspond to time steps. The “Median Src-Sen Dist” in the y -axis label of the subfigures in the first row is defined as $\text{median}_i \|p_i - q\|$, where $\|p_i - q\|$ is sensor i ’s distance to the source. The “Median Est. Error” in the second row is defined as $\text{median}_i \|\hat{q}_i - q\|$, where $\|\hat{q}_i - q\|$ is sensor i ’s estimation error. Blue curves: our distributed algorithm. Green curves: Variation I. Orange curves: Variation II. Red curves: our centralized algorithm. (a) Four sensors. (b) Ten sensors. (c) 20 sensors. (d) 40 sensors.

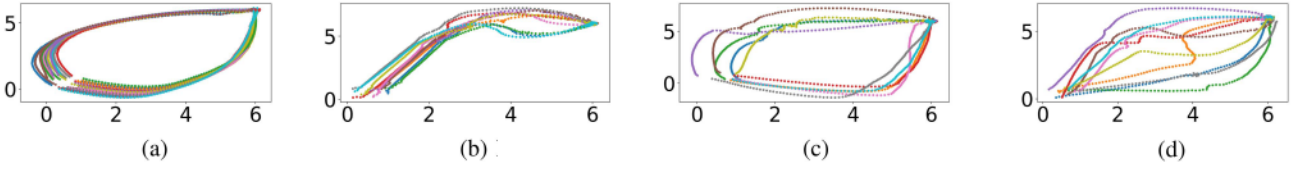


Fig. 11. Sample trajectories from the experiments in Fig. 10. The samples are taken randomly from 100 simulations. There are ten sensors in each figure. The figures illustrate the general trend that the centralized algorithm tends to spread out more than the distributed variations in the initial steps. Also, Variation I spreads out more than Variation II and our distributed algorithm. (a) Centralized. (b) Distributed. (c) Variation I. (d) Variation II.

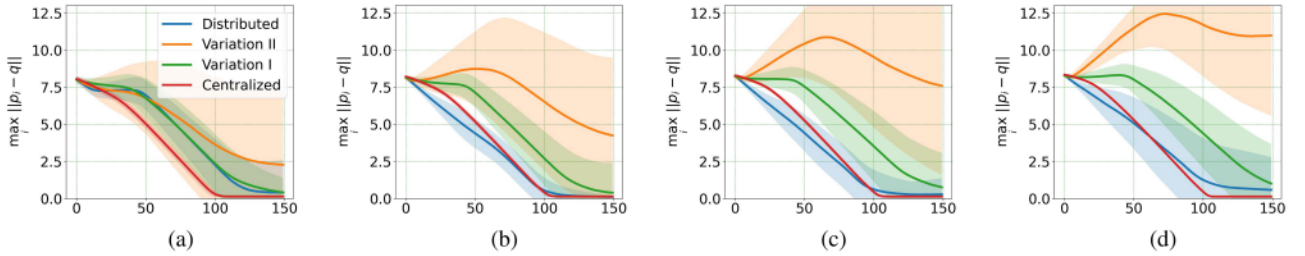


Fig. 12. Comparison of the entire team’s convergence to the source for different variations, in terms of $\max_i \|p_i - q\|$. The figures are generated using the same data in Fig. 10. The x -axes correspond to time steps. (a) Four sensors. (b) Ten sensors. (c) 20 sensors. (d) 40 sensors.

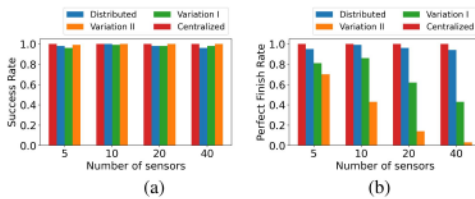


Fig. 13. Success/perfect finish rates for the variations. (a) Success rate. (b) Perfect finish rate.

Figs. 10–16, the x -axes are time steps, and the color bands show the standard deviation across 100 repetitive experiments. Specifically, our experiments compare the following groups.

- **Distributed algorithm 2:** Consensus is used in estimation and gradient update. The sensors have a shared loss function.
- **Variation I:** Consensus is used in estimation but not gradient update. The sensors have different local loss functions.
- **Variation II:** No consensus in either estimation or gradient update. Each sensor gathers measurement information from its neighbors and use local EKF for estimation. The sensors have different local loss functions.
- **Centralized algorithm 1:** The first three groups are implemented as distributed algorithms. The fourth is Algorithm 1 with a central controller, which serves as the control group. Detailed descriptions for the two variants

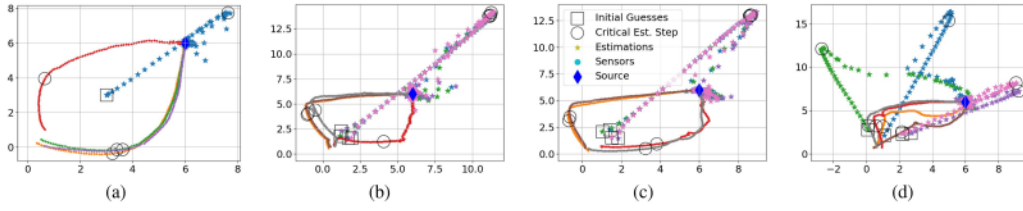


Fig. 14. Sample trajectories and estimations for the variations of our algorithm. The samples are randomly selected from 100 simulations. For visualization clarity, we only show the results for four-sensor teams. The circles with label “Critical Est. Step” indicates the estimations and sensor locations when the estimation error is the largest. (a) Centralized. (b) Distributed. (c) Variation I. (d) Variation II.

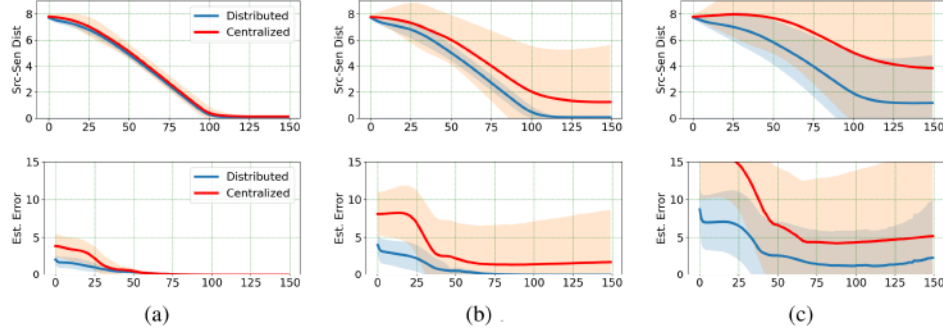


Fig. 15. How the initial guess deviation influences convergence and estimation error. The x -axes correspond to time steps. (a) $D = 10$. (b) $D = 20$. (c) $D = 40$.

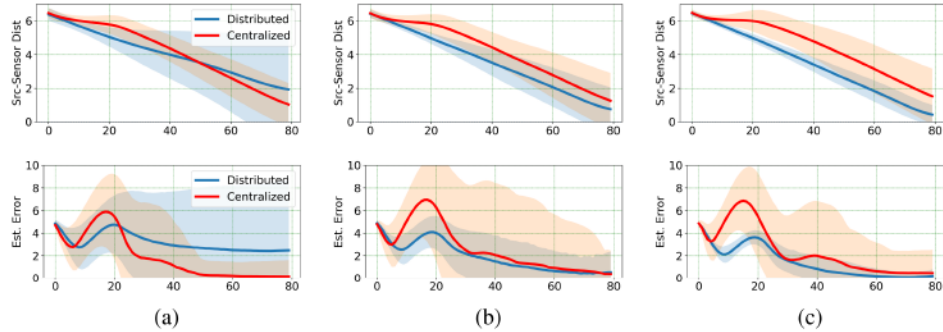


Fig. 16. How the communication delay influences convergence and estimation. The x -axes correspond to time steps. (a) Ten sensors with delay. (b) 20 sensors with delay. (c) 40 sensors with delay.

I and II are provided in Appendix C. All sensors use the same measurement model as defined in (24). We simplify the sensor communication network so that the distributed algorithms use a static, undirected circulant network with each mobile sensor connected with two neighbors.

Although the neighbors of a sensor in physical locations could change over time, the communication neighborhood is fixed and does not vary in our experiments. The robots communicate via a single-hop, static network, where each communicates only with a fixed set of neighbors on the network.² Using a fixed and sparsely connected communication network simulates the effect of limited information in the distributed setting while al-

lowing simpler, more interpretable results from the experiments. However, fixed communication is indeed a limiting assumption. Our algorithm can be revised to accommodate a more general, potentially time-varying network, such as the ad hoc network where the robots can only communicate with teammates within a limited distance. Based on the previous studies on distributed consensus and optimization over time-varying communication graphs, e.g., [44], [45], and [46], we expect our algorithm to achieve good performance as long as the network remains connected for a sufficient fraction of the time. This is left as our future work.

We repeat the experiments of seeking a static source. The sensor locations are initialized in the same way as in Section V-A. The source is still situated at (6, 6). We use the fixed location (3, 3) as the initial guess of the source location for the centralized control group. As for the first three distributed groups, each sensor uses an independent random location in $[0, 3] \times [0, 3]$ as the initial guess.

²This feature is also implementable in our hardware experiments because our lab space is small enough that all sensors are within the maximal range of communication with one another.

The main results are shown in Fig. 10, showing the time evolution of median sensor-source distance and estimation error. All variations converge to the source in around 100–125 steps. Sections V-B2 and V-B3 further analyze the data from Fig. 10 to demonstrate the advantage of our distributed algorithm over Variations I and II in terms of full-team convergence and success/perfect finish rates.

1) *Discussion on the Main Results:* The curves in the first row of Fig. 10 represent $\text{median}_i \|p_i - q\|$, the median of source-sensor distances within the team. The second row shows the median estimation error $\text{median}_i \|\hat{q}_i - q\|$. All the estimation error plots in Fig. 10 show the distributed variations have an overall worse estimation than the centralized algorithm. Also, note that when the number of sensors is small, as in Fig. 10(a), the centralized algorithm has a clear advantage over the distributed variations in estimation and convergence to the source. These results exemplify one of the weaknesses of the distributed variations: each agent has access to much less measurement information than the centralized algorithm.

However, as the number of sensors increases, we can observe that distributed variations improve in both estimation and convergence to the source. As the number of sensors exceeds 10, Variation II and our distributed algorithm achieve comparable convergence to the source as the centralized algorithm. The centralized algorithm still reaches the source the fastest, in around 100 steps, whereas Variation II and our distributed algorithm take slightly longer than that. But interestingly, the distance to the source for the centralized algorithm seems to decrease slower than these two variations in the initial steps, and the reason is likely that the centralized algorithm tends to spread out more than the distributed variations due to the stronger coordination in sensor movements, as shown in the illustrative example in Fig. 11.

It is also interesting that Variation I seems to be consistently worse than other distributed variations in convergence to the source. Recall that Variation I performs consensus in estimation but not in gradient update, giving it more information than Variation II. We do observe from Fig. 10 that in the early stages, its estimation error is comparable with our distributed algorithm and is lower than Variation II. However, as illustrated in Fig. 11, Variation I tends to spread out more than other distributed variations, yet it does not coordinate the movement among the agents, meaning it is slow to converge to the source. These factors potentially explain why Variation I does not have an advantage over Variation II in terms of the $\text{median}_i \|p_i - q\|$ metric.

The main results suggest that with a large number of sensors, even with the limited information for each sensor, as in Variation II, the distributed variations can still perform well. The sheer size of the distributed sensor team allows a high chance of some sensors getting close to the source and obtaining good measurement readings. Nevertheless, our distributed algorithm still outperforms Variations I and II due to the additional consensus in estimation and movement coordination.

Overall, our distributed algorithm converges to the source at a comparable rate as the centralized implementation. It also converges to the source slightly faster than Variations I/II and

exhibits more improvement when using more sensors. Furthermore, our algorithm maintains a clear advantage in estimation accuracy over Variations I/II throughout the seeking process.

2) *Convergence of the Entire Team:* The advantage of our distributed algorithm over Variations I and II is more prominent in terms of the entire team's convergence to the source. Fig. 12 shows the performance of the variations in terms of $\max_i \|p_i - q\|$ using the same data as in Fig. 10, where $\|p_i - q\|$ is the distance between sensor i and the source. The convergence of $\max_i \|p_i - q\|$ to 0, thus, characterizes the entire team's convergence to the source, a condition stronger than the one we proposed in (4). Fig. 12 shows the $\max_i \|p_i - q\|$ of our distributed algorithm converges to the source almost as fast as the centralized algorithm when the number of sensors exceeds 10; meanwhile, as the number of sensors increases, the performance of Variation I gets slightly worse, and Variation II gets significantly worse. These results demonstrate that with consensus in estimation and motion planning, our distributed algorithm can ensure all sensors come to the source sufficiently fast. In contrast, without sufficient coordination via consensus, as in Variations I and II, some sensors could perform much worse than others.

3) *Success and Perfect Finish Rates:* Fig. 13 shows the perfect finish/success rates of different variations of our algorithm. The figures are generated using the same data from Fig. 10. A robot is deemed to have reached the source if its distance from the source is below 0.2. A "success" is defined in (4), corresponding to a trial where *at least one* sensor reaches the source at the final step. A "perfect finish" is a trial where *all* sensors achieve that, which, such as the $\max_i \|p_i - q\|$ metric, is a notion characterizing the entire team's convergence. The success and perfect finish rates are computed over 100 trials. The results show that all variations have close to 100% success rates, which demonstrates the effectiveness of our information-based source-seeking framework. The perfect finish rates of our distributed algorithm are very close to one over any number of sensors. In contrast, the perfect finish rates for Variations I and II drop significantly as the number of sensors increases. The perfect finish rate results again demonstrate the crucial role of consensus in estimation and motion planning in our distributed algorithm: It allows the entire team to reach the source, achieving a more challenging control goal than (4).

4) *Discussion on Nonmonotonic Estimation Error:* In Fig. 10, we observe that the estimation errors do not decrease monotonically. Indeed, classical results for Kalman filters show that monotonic convergence can be achieved, but that applies only to linear systems, where both the measurement and motion models are linear. In contrast, the estimation of EKF on a nonlinear measurement model, the estimation error of which is generally not monotonic and could diverge given bad initialization. Although the criteria for the convergence of EKF is discussed in detail in [36], we are unaware of any literature that discusses the criterion for monotonicity of the estimation error for EKF.

We also observe in Fig. 10 the somewhat counter-intuitive behavior that source-sensor distances can decrease as the estimation errors increase. Fig. 14 unveils the reason for this curious phenomenon. It shows a common situation in the experiments,

where the estimation from the EKF could move toward and overshoot the true source location but then comes back to the source as the sensors spread out and move closer to the source. During the overshoot, the estimation error gets larger, but the estimation still guides the sensors to move toward the direction of the source. That is why we observe that the estimation error increases while the sensor-source distance decreases during this period. We also note that in Fig. 14(d) for Variation II, the estimations of two sensors are in the direction of the source, while the estimations for the other two are not in the direction of the source. But because the two sensors with the “right direction” come closer to the source, their measurements become better/more informative, which is shared with the other two sensors and “saves” them from divergence. This phenomenon is common in experiments and explains why Variation II performs well in terms of the median distance metric in Fig. 10 even though it uses no consensus in estimation and motion planning.

C. Sensitivity to the Initial Guess

Our distributed algorithm is not only as effective in source seeking as the centralized implementation; it is also more robust in multiple ways. For example, the distributed algorithm is much less sensitive to the initial guesses of the source location. In the following experiments, we compare the sensitivity of the distributed and centralized implementation to initial guesses in terms of convergence and estimation error. Five sensors are used in all experiments. The locations of the sensors and the source are initialized in the same way as in the previous experiments. Random initial guesses \hat{q}_0 are given to estimators in both the distributed and centralized implementations. It is defined by

$$\hat{q}_0 = q + \frac{D}{2} \cdot \begin{bmatrix} \Delta_1 \\ \Delta_2 \end{bmatrix}, \Delta_{1,2} \stackrel{i.i.d.}{\sim} \text{Unif}([-1, 1]). \quad (25)$$

In other words, \hat{q}_0 is a random location in a box with side length D centered at q . Different estimators may have different \hat{q}_0 , but the level of deviation D is kept the same among estimators for one experiment.

Fig. 15 shows the distributed algorithm exhibit clear advantages over the centralized implementation in convergence and estimation when deviation D increases. The distributed algorithm prevails because using more estimators adds robustness. Although the centralized implementation has more data for estimation, it runs only one estimator. If the initial guess is poor, the estimator may fail. In contrast, sensors of the distributed algorithm form an ensemble of estimators. As long as the majority of the initial guesses are decent, individual estimations should improve after rounds of consensus.

D. Robustness to Communication Delay

Apart from being robust to errors in initial guesses, the distributed algorithm is more robust than the centralized implementation to communication delay. Note that communication delay typically grows with the incoming information rate [47, M/M/1

system, Sec. 4.1.2], which usually increases with the number of incoming connections. Therefore, the delay increases with the number of incoming connections. The consequence is that the delay in centralized implementation increases with the number of sensors used, whereas the delay in the distributed algorithm can remain low as the network expands as long as the incoming degree of each node stays unchanged.

The following numerical experiments show that the distributed algorithm is more robust to communication delay. We assume each incoming connection to a node brings a 0.5 time step delay in information passing. That is, if m sensors are used in the centralized algorithm, the centralized controller receives all information with a delay of $\lfloor \frac{m}{2} \rfloor$ time steps; if there are k neighbors connected to sensor j in the distributed algorithm, sensor j receives all information with a delay of $\lfloor \frac{k}{2} \rfloor$ time steps. The delay is modeled in this way since each sensor takes and sends out measurements to the neighbors at a roughly constant rate. Initialization is the same as in the previous experiments. In particular, each sensor in the distributed algorithm is connected to two neighbors regardless of the number of sensors.

Fig. 16 shows that when the delay is in effect, the advantage of the centralized implementation over the distributed algorithm dissipates as the number of sensors grows. In the centralized implementation, the benefit of more sensor data does not offset the negative effect of increasing delay, and the performance gradually degrades. On the other hand, distributed sensors benefit from the increasing global information shared in the network while the delay remains low. Therefore, the centralized implementation is eventually overtaken by the distributed algorithm.

E. Lab Implementations

We implement our distributed algorithm on Turtlebot3 robotic ground vehicles to seek a light source in a dark room. The light source is an LED lamp. Light sensors are installed on Turtlebots to measure the local light intensity. The robots communicate through an ad hoc WiFi network, each being an individual node. We only allow each robot to communicate with two other predetermined robots throughout the experiment as a reasonable approximation of a nonfully connected network. An indoor motion capture system captures the positions of the robots and the source.

The algorithm is implemented as a ROS 2 package in Python language [48]. The package is installed on the onboard computer of every robot. By running the package, every robot gathers its position from the motion capture system, communicates information required in estimation and waypoint planning, makes estimations, and decides its control actions.

As a measurement model is needed to estimate the source location, we fit a function for each robot describing the relationship between their measurements and distances to the source before deploying them in source seeking. The function is defined by

$$g_i(r_i) = k_i(r_i - C_{1,i})^{-b_i} + C_{0,i} \quad (26)$$

where y_i and r_i are the measured data from the i th mobile sensor, and $k_i > 0$, $b_i > 0$, $C_{0,i}$, $C_{1,i}$ are the associated model parameters.

In the implementation, a robot is deemed to have reached the source if its distance from the source is below 0.5 m. A separate program that runs independently from the source seeking algorithm monitors whether any robot has reached the source using the motion capture data and notifies the robots if they contact the source. The accompanying video shows that the robots implementing our distributed algorithm converge to the source consistently.

Remark 7: Our lab resources constrain the hardware implementation; thus, the hardware experiments are less comprehensive than the numerical tests, an aspect of the experiments that could be improved. Nevertheless, the hardware experiments serve as a proof-of-concept showing the proposed algorithms are implementable on physical systems with satisfactory performance. We hope the combination of extensive numerical tests and hardware experiments has demonstrated how the algorithm works.

VI. CONCLUSION

This article proposes a method for multirobot source seeking that utilizes the FIM associated with the estimated source location to direct the robots' movements. We show that improving the trace of the inverse of FIM aids estimation while guiding the robots to converge to the source. The method is extended to the distributed setting where a central controller is absent, and the robots make individual decisions while communicating via a distributed network. We propose consensus schemes that make distributed estimation and gradient update effective. The algorithms are verified in numerical experiments and on physical robotic systems.

Our work leads to a few future directions. One is to combine our work with Bayesian optimization methods to learn the measurement model while the robots seek the source. Progress in this direction removes the requirement of a known measurement model, making the method more flexible. It is also imperative to develop the ability to navigate an environment with obstacles during source seeking, especially when deploying our method in complex environments, such as search-and-rescue missions.

APPENDIX A MOTION PLANNING

The motion planner MP can be viewed as a device that transforms the planned waypoints into low-level actuation of a mobile sensor: by applying $(u_i(1), \dots, u_i(T)) = \text{MP}(\tilde{p}_i(0), \dots, \tilde{p}_i(T))$ to sensor i , the sensor will follow the trajectory of the waypoints $\tilde{p}_i(0), \dots, \tilde{p}_i(T)$. The motion planner MP typically requires the knowledge of the sensors' motion dynamics to compute the control inputs, and any method that can fulfill this task can be a motion planner in Algorithm 1. In our implementation, the motion planner combines spline-based motion generation described in [49] and the linear quadratic regulator. Regarding the planning horizon T choice, in the Gazebo simulations and hardware implementation, we set $T = 20$ to ensure stability in sensor movement and robustness to

disturbances. Meanwhile, in numerical studies where the robot dynamics are not simulated, it suffices to set $T = 1$.

APPENDIX B PROOF OF PROPOSITIONS 1 AND 2

Recall that \hat{r}_i is the directional unit vector pointing from q (the source) to p_i (sensor i 's position), i.e.,

$$\hat{r}_i = \frac{p_i - q}{\|p_i - q\|}.$$

We first derive expressions for FIM and the loss function L .

Lemma 1: Suppose individual measurements are isotropic as in Assumption 1. Then,

$$\text{FIM} = \sum_{i=1}^m |g'_i(r_i)|^2 \hat{r}_i \hat{r}_i^T. \quad (27)$$

Moreover,

$$L(p, q) = \text{Tr}(\text{FIM}^{-1}) = \sum_{i=1}^m \frac{1}{\lambda_i(\text{FIM})} > 0 \quad (28)$$

where $\lambda_i(\text{FIM})$ is the i th eigenvalue of FIM.

Proof: By noticing that $r_i = \|p_i - q\|$ and using the chain rule of calculus, one can show that

$$\nabla_q h_i(p_i, q) = -g'_i(r_i) \hat{r}_i.$$

Denote $A = \nabla_q H(p, q)$ and $A_i = \nabla_q h_i(p_i, q)$. Then, from the definition of H in (2), we have

$$A = [A_1 | A_2 | \dots | A_m]$$

and so

$$\text{FIM} = AA^T = \sum_{i=1}^m A_i A_i^T = \sum_{i=1}^m |g'_i(r_i)|^2 \hat{r}_i \hat{r}_i^T.$$

Then, (28) follows since

$$\text{FIM} \succeq \left(\min_i |g'_i(r_i)| \right) \sum_{i=1}^m \hat{r}_i \hat{r}_i^T \succ 0$$

and the eigenvalues of FIM^{-1} are the reciprocals of those of FIM. \square

With the formulas of L and FIM, and by further imposing the monotonicity property of h_i in Assumption 1.2, we can show that minimizing L leads to reaching the source as stated in Proposition 1.

Proof of Proposition 1

First, we take the maximum over all the summation coefficients in FIM to get

$$\text{FIM} \preceq \max_{i=1,2,\dots,m} |g'_i(r_i)|^2 \sum_i \hat{r}_i \hat{r}_i^T.$$

Let $\lambda_{\min}(P)$ denote the smallest eigenvalue of a positive semidefinite matrix P . Then, by (28), we can see that

$$\frac{1}{\lambda_{\min}(\text{FIM})} \leq L(p, q) \leq \frac{k}{\lambda_{\min}(\text{FIM})}$$

where we recall that k is the dimension of q and also the size of FIM. Consequently,

$$\frac{1}{L(\mathbf{p}, q)} \leq \lambda_{\min}(\text{FIM}) \leq \max_i |g'_i(r_i)|^2 \cdot \lambda_{\min} \left(\sum_{i=1}^m \hat{r}_i \hat{r}_i^T \right).$$

Since we assume $\sum_{i=1}^m \hat{r}_i \hat{r}_i^T \succ 0$, there is $\lambda_{\min}(\sum_{i=1}^m \hat{r}_i \hat{r}_i^T) > 0$. Then, because $\lambda_{\min}(\sum_{i=1}^m \hat{r}_i \hat{r}_i^T) \leq m$, we have

$$\frac{1}{m \cdot \max_i |g'_i(r_i)|^2} \leq L(\mathbf{p}, q). \quad (29)$$

So by Assumption 1, $|g'_i(r_i)|$ monotonically increases as r_i decreases, so we have $\min_i r_i$ tends to decrease as $L(\mathbf{p}, q)$ decreases, which completes the proof. \square

Proof of Proposition 2: Let $\bar{\mathbf{p}} = [\bar{p}_1, \bar{p}_2, \dots, \bar{p}_m]^T := \lim_{t \rightarrow \infty} \mathbf{p}(t)$ be the location vector where the sensors converge to.

We prove the theorem by contradiction. Assume $\min_i \|p_i(t) - q\|$ does not converge to 0. Then, there must be constant $\epsilon > 0$ such that

$$\bar{r}_i := \|\bar{p}_i - q\| \geq \epsilon \quad \forall i. \quad (30)$$

Together with the continuity of $L(\mathbf{p}, q)$ in \mathbf{p} on $\{\mathbf{p} : p_i \neq q \forall i\}$, the assumption leads to the conclusion that the infimum of L in the theorem should be replaced with the minimum, and is attainable at $\bar{\mathbf{p}}$.

We now construct a contradiction to the assumption in (30) showing $\bar{\mathbf{p}}$ cannot be the minimum of L . Consider the location vector $\mathbf{b} = [b_1, b_2, \dots, b_m]^T$ defined as

$$b_i := q + \frac{\epsilon}{2} \cdot \frac{\bar{p}_i - q}{\bar{r}_i} \quad \forall i.$$

So that

$$\|b_i - q\| = \frac{\epsilon}{2}.$$

Note that \mathbf{b} satisfies $b_i \neq q$ for all i . We also note that

$$\begin{aligned} \text{FIM}(\mathbf{b}, q) &= \sum_{i=1}^m |g'_i\left(\frac{\epsilon}{2}\right)| \left(\frac{\bar{p}_i - q}{\bar{r}_i} \right) \left(\frac{\bar{p}_i - q}{\bar{r}_i} \right)^T \\ &\succeq \alpha \underbrace{\sum_{i=1}^m |g'_i(\bar{r}_i)| \left(\frac{\bar{p}_i - q}{\bar{r}_i} \right) \left(\frac{\bar{p}_i - q}{\bar{r}_i} \right)^T}_{\text{FIM}(\bar{\mathbf{p}}, q)} \end{aligned}$$

where

$$\alpha = \min_{i=1,2,\dots,m} \frac{|g'_i(\epsilon/2)|}{|g'_i(\bar{r}_i)|}.$$

Note that $\alpha > 1$ since $\bar{r}_i > \epsilon/2$ and $|g'_i(r_i)|$ is assumed strictly monotone in r_i in the theorem. Therefore,

$$\begin{aligned} L(\mathbf{b}, q) &= \sum_{i=1}^m \frac{1}{\lambda_i(\text{FIM}(\mathbf{b}, q))} \\ &\leq \frac{1}{\alpha} \sum_{i=1}^m \frac{1}{\lambda_i(\text{FIM}(\bar{\mathbf{p}}, q))} = \frac{1}{\alpha} L(\bar{\mathbf{p}}, q) \\ &< L(\bar{\mathbf{p}}, q). \end{aligned}$$

Therefore, the location vector \mathbf{b} , which is closer to q than $\bar{\mathbf{p}}$, achieves a lower L value than $\bar{\mathbf{p}}$. We have reached a contradiction, and (30) cannot hold. So $\bar{r}_i = 0$ for some i and $\lim_{t \rightarrow \infty} \min_i \|p_i(t) - q\| = 0$. \square

APPENDIX C

DISTRIBUTED ALGORITHM VARIANTS I AND II

The two variations are formally defined as follows.

- 1) Sensor j in Variations I and II no longer share a loss function with all sensors. Instead, it uses the following local loss that may be different for other sensors:

$$\begin{aligned} L_j(\mathbf{p}_j, q) &= \text{Tr} \left[\left(\nabla_q H_j(\mathbf{p}_j, q) \cdot \nabla_q H_j(\mathbf{p}_j, q)^T \right)^{-1} \right]. \quad (31) \end{aligned}$$

This local loss reflects the FIM about q contained in the neighborhood measurements as opposed to the entirety of sensor measurements.

- 2) Sensor j in Variation II performs the *local Kalman filter* update defined in (18) using neighborhood information \mathbf{p}_j and \mathbf{y}_j . No consensus is required.
- 3) Sensor j in Variations I and II skips lines 5–7 in Algorithm 2, and changes lines 8–9 to be
 - a) $\tilde{\mathbf{p}}_j(0) \leftarrow \mathbf{p}_j$;
 - b) calculate joint-waypoints $\tilde{\mathbf{p}}_j(1), \dots, \tilde{\mathbf{p}}_j(T)$

$$\tilde{\mathbf{p}}_j(t+1) \leftarrow \tilde{\mathbf{p}}_j(t) - \alpha_t M_t \nabla_{\mathbf{p}_j} L_j(\mathbf{p}_j = \tilde{\mathbf{p}}_j(t), \hat{q}_j); \quad (32)$$

- c) Extract sensor j 's waypoints $\tilde{\mathbf{p}}_j(1), \dots, \tilde{\mathbf{p}}_j(T)$ from joint-waypoints $\tilde{\mathbf{p}}_j(1), \dots, \tilde{\mathbf{p}}_j(T)$;

Note that no consensus is required to compute the gradient of L_j . The remaining parts of the algorithm are identical to our algorithm and the variations. In particular, sensor j in Variation I still performs consensus EKF update.

APPENDIX D

UNDERSTANDING THE GEOMETRIC PROPERTIES OF THE TRAJECTORY OF THE COVARIANCE METRIC

Fig. 7(d) shows that the sensors under the covariance metric approach the source in straight lines and then spread out a little when close to the source. We can understand the initial straight-line behavior by analyzing the direction of the gradient when sensors are far away from the source. Recall that the covariance metric defined in Section V-A3 is $\text{Tr}(P) = \text{Tr}((\nabla H R^{-1} \nabla H^T + P_0^{-1})^{-1})$. We set $R = I$ in Fig. 7, so the covariance metric is effectively $\text{Tr}(P) = \text{Tr}((\nabla H \nabla H^T +$

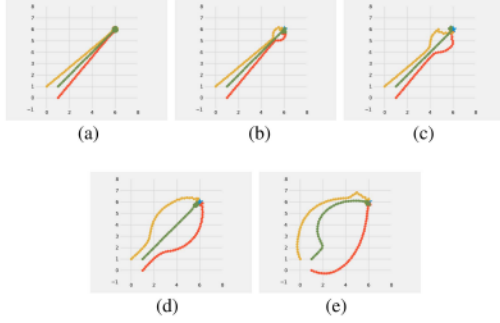


Fig. 17. Effect of P_0^{-1} on the size of the “bump” of the gradient-descent trajectory of the covariance loss function. The matrix R is fixed to be $1.0I$ for all figures above. (a) $P_0^{-1} = 100I$. (b) $P_0^{-1} = 1.0I$. (c) $P_0^{-1} = 0.01I$. (d) $P_0^{-1} = 10^{-4}I$. (e) $P_0^{-1} = 10^{-6}I$.

$P_0^{-1})^{-1}$). Through matrix calculus, we can show that the covariance metric’s gradient is given by the following:

$$\begin{aligned} G_j &:= \nabla_{p_j} \text{Tr}((\nabla H \nabla H^\top + P_0^{-1})^{-1}) \\ &= -2 \nabla_{p_j} A_j (\nabla H \nabla H^\top + P_0^{-1})^{-2} A_j \end{aligned}$$

where $A_j := \nabla_{p_j} g_j(\|p_j - q\|)$, and A_j and $\nabla_{p_j} A_j$ can be shown to be

$$\begin{aligned} A_j &= -g'_j(r_j) \hat{r}_j \\ \nabla_{p_j} A_j &= -g''_j(r_j) \hat{r}_j \hat{r}_j^\top - \frac{g'_j(r_j)}{r_j} \hat{t}_j \hat{t}_j^\top \end{aligned}$$

where $\hat{r}_j = \frac{p_j - q}{\|p_j - q\|}$ is the unit vector pointing from q to p_j , and \hat{t}_j is the unit vector orthogonal to \hat{r}_j .

Since we assume the individual measurement function $|g'_j(r_j)|$ gets smaller as r_j increases, P_0^{-1} dominates $\nabla H \nabla H^\top$ when r_j are large (sensors are far away from the source). We note that we set $P_0 = I$ in Fig. 7(d). Therefore, when sensors are far away from the source, the gradient for sensor j is approximately

$$\begin{aligned} G_j &\approx -2 \nabla_{p_j} A_j (P_0^{-1})^{-2} A_j \\ &= -2 (\nabla_{p_j} A_j) A_j \\ &= -2 g'_j(r_j) g''_j(r_j) \hat{r}_j. \end{aligned}$$

Also, since the g_j we consider are in the form of $g_j(r_j) = k_j(r_j - C_{1,j})^{-b_j} + C_{0,j}$, its first- and second-order derivatives satisfies $g'_j(r_j) \leq 0$ and $g''_j(r_j) \geq 0$. Therefore, the descent direction $-G_j$ is approximately aligned with the direction of $-\hat{r}_j$ when the sensors are far away from the source, causing the straight-line behavior.

When sensors are close to the source, analyzing the gradient direction is less trivial, but we do note that the covariance metric is roughly equivalent to $\text{Tr}(\text{FIM}^{-1})$ in this condition since the $\nabla H \nabla H^\top$ term dominates P_0^{-1} . We argued in Remark 6 that minimizing the $\text{Tr}(\text{FIM}^{-1})$ amounts to approaching the source and creating separation between the sensors simultaneously, which explains why the sensors spread out a little at the end.

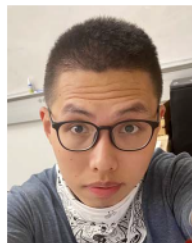
Fig. 17 provides further numerical evidence showing the dominance between P_0^{-1} and $\nabla H \nabla H^\top$ terms influence the sensor movements. It shows that the following holds.

- 1) When P_0^{-1} is large, it dominates the covariance metric and gives rise to straight-line behavior.
- 2) On the other hand, when P_0^{-1} becomes smaller, the $\nabla H \nabla H^\top$ term starts to dominate and the “bump” in the trajectory becomes increasingly larger.
- 3) When P_0^{-1} becomes extremely small, the trajectory shows the behavior of encouraging separation among the sensors from the very start, which matches the behavior of the $\text{Tr}(\text{FIM}^{-1})$ metric, which is unsurprising since $\text{FIM} = \nabla H \nabla H^\top$.

REFERENCES

- [1] B. P. Duisterhof et al., “Learning to seek: Autonomous source seeking with deep reinforcement learning onboard a nano drone microcontroller,” 2021, *arXiv:1909.11236*.
- [2] S. Xu, Y. Ou, and W. Zheng, “Optimal sensor-target geometries for 3-D static target localization using received-signal-strength measurements,” *IEEE Signal Process. Lett.*, vol. 26, no. 7, pp. 966–970, Jul. 2019.
- [3] R. Khodayi-Mehr, W. Aquino, and M. M. Zavlanos, “Model-based active source identification in complex environments,” *IEEE Trans. Robot.*, vol. 35, no. 3, pp. 633–652, Jun. 2019.
- [4] B. A. Angélico, L. F. O. Chamon, S. Paternain, A. Ribeiro, and G. J. Pappas, “Source seeking in unknown environments with convex obstacles,” in *Proc. Amer. Control Conf. (ACC)*, New Orleans, LA, USA, 2021, pp. 5055–5061, doi: [10.23919/ACC50511.2021.9483164](https://doi.org/10.23919/ACC50511.2021.9483164).
- [5] R. Bachmayer and N. E. Leonard, “Vehicle networks for gradient descent in a sampled environment,” in *Proc. 41st IEEE Conf. Decis. Control*, 2002, pp. 112–117.
- [6] P. Ogren, E. Fiorelli, and N. E. Leonard, “Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment,” *IEEE Trans. Autom. Control*, vol. 49, no. 8, pp. 1292–1302, Aug. 2004.
- [7] B. J. Moore and C. Canudas-de Wit, “Source seeking via collaborative measurements by a circular formation of agents,” in *Proc. Amer. Control Conf.*, 2010, pp. 6417–6422.
- [8] S. Li, R. Kong, and Y. Guo, “Cooperative distributed source seeking by multiple robots: Algorithms and experiments,” in *IEEE/ASME Trans. Mechatron.*, vol. 19, no. 6, pp. 1810–1820, Dec. 2014.
- [9] L. Briñón-Arranz, L. Schenato, and A. Seuret, “Distributed source seeking via a circular formation of agents under communication constraints,” in *IEEE Trans. Control Netw. Syst.*, vol. 3, no. 2, pp. 104–115, Jun. 2016.
- [10] 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.
- [11] S. Martínez and F. Bullo, “Optimal sensor placement and motion coordination for target tracking,” *Automatica*, vol. 42, no. 4, pp. 661–668, 2006.
- [12] B. Bayat, N. Crasta, H. Li, and A. Ijspeert, “Optimal search strategies for pollutant source localization,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 1801–1807.
- [13] G. M. Hoffmann and C. J. Tomlin, “Mobile sensor network control using mutual information methods and particle filters,” *IEEE Trans. Autom. Control*, vol. 55, no. 1, pp. 32–47, Jan. 2010.
- [14] D. Moreno-Salinas, A. Pascoal, and J. Aranda, “Optimal sensor placement for multiple target positioning with range-only measurements in two-dimensional scenarios,” *Sensors*, vol. 13, no. 8, pp. 10674–10710, 2013.
- [15] A. N. Bishop and P. N. Pathirana, “Optimal trajectories for homing navigation with bearing measurements,” *IFAC Proc. Volumes*, vol. 41, no. 2, pp. 12117–12123, 2008.
- [16] M. S. Lee, D. Shy, W. R. Whittaker, and N. Michael, “Active range and bearing-based radiation source localization,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 1389–1394.
- [17] S. Ponda, R. Kolacinski, and E. Frazzoli, “Trajectory optimization for target localization using small unmanned aerial vehicles,” in *Proc. AIAA Guid. Navigation, Control Conf.*, 2009, pp. 1–25, doi: [10.2514/6.2009-6015](https://doi.org/10.2514/6.2009-6015).

- [18] H. Cramér, *Mathematical Methods of Statistics*. Princeton, NJ, USA: Princeton Univ. Press, 1946.
- [19] C. R. Rao, "Information and the accuracy attainable in the estimation of statistical parameters," *Bull. Calcutta Math. Soc.*, vol. 37, no. 3, pp. 81–91, 1945.
- [20] T. Zhang, V. Qin, Y. Tang, and N. Li, "Source seeking by dynamic source location estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 2598–2605.
- [21] V. Gazi and K. Passino, "Stability analysis of social foraging swarms: Combined effects of attractant/repellent profiles," in *Proc. IEEE 41st Conf. Decis. Control*, 2002, pp. 2848–2853.
- [22] A. Okubo, "Dynamical aspects of animal grouping: Swarms, schools, flocks, and herds," *Adv. Biophys.*, vol. 22, pp. 1–94, 1986.
- [23] D. Baronov and J. Baillieul, "A motion description language for robotic reconnaissance of unknown fields," *Eur. J. Control*, vol. 17, no. 5/6, pp. 512–525, 2011.
- [24] A. Mavrommati, E. Tzorakoleftherakis, I. Abraham, and T. D. Murphey, "Real-time area coverage and target localization using receding-horizon ergodic exploration," *IEEE Trans. Robot.*, vol. 34, no. 1, pp. 62–80, Feb. 2018.
- [25] I. Abraham and T. D. Murphey, "Decentralized ergodic control: Distribution-driven sensing and exploration for multiagent systems," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 2987–2994, Oct. 2018.
- [26] R. Khodayi-Mehr, W. Aquino, and M. M. Zavlanos, "Distributed reduced order source identification," in *Proc. Annu. Amer. Control Conf.*, 2018, pp. 1084–1089.
- [27] C. Yang, L. Kaplan, and E. Blasch, "Performance measures of covariance and information matrices in resource management for target state estimation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 3, pp. 2594–2613, Jul. 2012.
- [28] 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.
- [29] B. Chawrow et al., "Information-theoretic planning with trajectory optimization for dense 3D mapping," in *Proc. Robot.: Sci. Syst.*, 2015, pp. 3–12, doi: [10.15607/RSS.2015.XI.003](https://doi.org/10.15607/RSS.2015.XI.003).
- [30] Y. A. Prabowo, R. Ranasinghe, G. Dissanayake, B. Riyanto, and B. Yulianto, "A Bayesian approach for gas source localization in large indoor environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 4432–4437.
- [31] C. Sánchez-Garrido, J. G. Monroy, and J. G. Jiménez, "Probabilistic estimation of the gas source location in indoor environments by combining gas and wind observations," in *Proc. Conf. APPIS*, 2018, pp. 110–121.
- [32] A. Benevento, M. Santos, G. Notarstefano, K. Paynabar, M. Bloch, and M. Egerstedt, "Multi-robot coordination for estimation and coverage of unknown spatial fields," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 7740–7746.
- [33] R. G. Brown and P. Y. Hwang, "The global positioning system: A case study," in *Introduction to Random Signals and Applied Kalman Filtering: With MATLAB Exercises and Solutions*. Hoboken, NJ, USA: Wiley, 1997, ch. 11, pp. 419–460.
- [34] M. Wickert and C. Siddappa, "Exploring the extended Kalman filter for GPS positioning using simulated user and satellite track data," in *Proc. 17th Python Sci. Conf.*, 2018, pp. 84–90.
- [35] Q. Shi, X. Cui, S. Zhao, and M. Lu, "Sequential TOA-based moving target localization in multi-agent networks," *IEEE Commun. Lett.*, vol. 24, no. 8, pp. 1719–1723, Aug. 2020.
- [36] K. Reif, S. Gunther, E. Yaz, and R. Unbehauen, "Stochastic stability of the discrete-time extended Kalman filter," *IEEE Trans. Autom. Control*, vol. 44, no. 4, pp. 714–728, Apr. 1999.
- [37] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.
- [38] A. Olshevsky and J. N. Tsitsiklis, "Convergence speed in distributed consensus and averaging," *SIAM J. Control Optim.*, vol. 48, no. 1, pp. 33–55, 2009.
- [39] R. Olfati-Saber, "Kalman-consensus filter: Optimality, stability, and performance," in *Proc. IEEE 48th Conf. Decis. Control held Jointly 28th Chin. Control Conf.*, 2009, pp. 7036–7042.
- [40] H. Long, Z. Qu, X. Fan, and S. Liu, "Distributed extended Kalman filter based on consensus filter for wireless sensor network," in *Proc. IEEE 10th World Congress Intell. Control Autom.*, 2012, pp. 4315–4319.
- [41] C. Ding, B. Song, A. Morye, J. A. Farrell, and A. K. Roy-Chowdhury, "Collaborative sensing in a distributed PTZ camera network," in *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3282–3295, Jul. 2012.
- [42] S. Katragadda, J. C. SanMiguel, and A. Cavallaro, "Consensus protocols for distributed tracking in wireless camera networks," in *Proc. 17th Int. Conf. Inf. Fusion*, 2014, pp. 1–8.
- [43] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2004, pp. 2149–2154.
- [44] K. You, Z. Li, and L. Xie, "Consensus condition for linear multi-agent systems over randomly switching topologies," *Automatica*, vol. 49, no. 10, pp. 3125–3132, 2013.
- [45] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 601–615, Mar. 2015.
- [46] T. Yang et al., "A survey of distributed optimization," *Annu. Rev. Control*, vol. 47, pp. 278–305, 2019.
- [47] C.-H. Ng and S. Boon-Hee, *Queueing Modelling Fundamentals: With Applications in Communication Networks*. Hoboken, NJ, USA: Wiley, 2008.
- [48] Z. Tianpeng, "ROS 2 package for distributed information-based source seeking," Nov. 2021. Accessed: Nov. 28, 2021. [Online]. Available: https://github.com/lina-robotics-lab/fim_track_2
- [49] R. Walambe, N. Agarwal, S. Kale, and V. Joshi, "Optimal trajectory generation for car-type mobile robot using spline interpolation," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 601–606, 2016.



Tianpeng Zhang (Graduate Student Member, IEEE) received the B.S. degree in computer science and mathematics from The Hong Kong University of Science and Technology, Hong Kong, in 2015. He is currently working toward the Doctoral degree in applied mathematics with Harvard University, Cambridge, MA, USA.

His research interests include multiagent vehicle teams, multiarmed bandit learning, and robotic manipulators.



Victor Qin is currently working toward the Ph.D. degree in aeronautics and astronautics with the Massachusetts Institute of Technology, Cambridge, MA, USA.

His interdisciplinary research focuses on the control, learning, and optimization of multiagent networks, especially through incentive mechanisms and with an emphasis on unmanned air traffic management. The goal of his research is to highlight innovations in future semiautonomous cyberphysical networks, and understand their interaction with economic and regulatory structures.

His research interests include cost-aware decentralized airspace allocation methods, cost-sharing frameworks for optimal network routing, and collision avoidance analysis of satellite constellations.

Dr. Qin was the recipient of the NSF Graduate Research Fellowship and a Mathworks Fellow.



Yujie Tang (Member, IEEE) received the bachelor's degree in electronic engineering from Tsinghua University, Beijing, China, in 2013, and the Ph.D. degree in electrical engineering from the California Institute of Technology, Pasadena, CA, USA, in 2019.

He is currently an Assistant Professor with the Department of Industrial Engineering and Management, Peking University, Beijing. Before joining Peking University, he was a Postdoctoral Fellow with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA. His research interests include distributed optimization, control and learning, and their applications in cyber-physical systems.



Na Li (Senior Member, IEEE) received the bachelor's degree in mathematics from Zhejiang University, Hangzhou, China, in 2007, and the Ph.D. degree in control and dynamical systems from the California Institute of Technology, Pasadena, CA, USA, in 2013.

She is currently the Winokur Family Professor of electrical engineering and applied mathematics at Harvard University, Cambridge, MA, USA. From 2013 to 2014, she was a Postdoctoral Associate with the Massachusetts Institute of Technology, Cambridge. She has held a variety of short-term visiting appointments, including the Simons Institute for the Theory of Computing, MIT, and Google Brain. Her research interests include the control, learning, and optimization of networked systems, including theory development, algorithm design, and applications to real-world cyber-physical societal system.

Dr. Li was the recipient of the NSF Career Award (2016), AFSOR Young Investigator Award (2017), ONR Young Investigator Award (2019), Donald P. Eckman Award (2019), McDonald Mentoring Award (2020), IFAC Manfred Thoma Medal (2023), along with some other awards.