Moe Z. Win, Florian Meyer, Zhenyu Liu, Wenhan Dai,
Stefania Bartoletti, and Andrea Conti

# Efficient Multisensor Localization for the Internet of Things

*Exploring a new class of scalable localization algorithms*

INTERNET OF THINGS—ISTOCKPHOTO.COM/IAREMENKO
CIRCUITS—IMAGE LICENSED BY INGRAM PUBLISHING

*Digital Object Identifier 10.1109/MSP.2018.2845907*
*Date of publication: 28 August 2018*

In the era of the Internet of Things (IoT), efficient localization is essential for emerging mass-market services and applications. IoT devices are heterogeneous in signaling, sensing, and mobility, and their resources for computation and communication are typically limited. Therefore, to enable location awareness in large-scale IoT networks, there is a need for efficient, scalable, and distributed multisensor fusion algorithms. This article presents a framework for designing network localization and navigation (NLN) for the IoT. Multisensor localization and operation algorithms developed within NLN can exploit spatiotemporal cooperation, are suitable for arbitrary, large-network sizes, and only rely on an information exchange among neighboring devices. The advantages of NLN are evaluated in a large-scale IoT network with 500 agents. In particular, because of multisensor fusion and cooperation, the presented network localization and operation algorithms can provide attractive localization performance and reduce communication overhead and energy consumption.

## IoT location awareness

Location awareness [1]–[6] is a cornerstone of the IoT and fosters a wide range of emerging applications, such as crowdsensing [7], big data analysis [8], environmental monitoring [9], and autonomous driving [10]. The position information of IoT devices can contribute to connecting and exchanging data more efficiently, preserving communication security, and allowing autonomous motion. The increasing number and different types of IoT devices generate scenarios in which heterogeneous data are collected distributedly using different sensing technologies. Compared to conventional wireless localization networks that typically consist of a limited number of homogeneous nodes, the scale and heterogeneity of an IoT network imposes new challenges that need to be addressed. Specifically, IoT localization and navigation calls for a new class of algorithms tailored to IoT networks.

In IoT networks, the sensing capabilities of the devices can vary significantly, providing different kinds of measurements carrying positional information such as range, angle of arrival,

channel state information, or inertial. Additionally, depending on the specific sensing technology used by each device, communication ranges and measurement accuracies are different. Since IoT devices are typically equipped only with inexpensive sensors having limited capabilities, high-accuracy localization and navigation usually requires multisensor fusion and device cooperation. However, state-of-the-art multisensor fusion algorithms based on sequential Bayesian estimation (SBE) [11]–[13] are often impractical for IoT applications due to their decentralized network topology and the limited processing units of IoT devices. Moreover, the high number of devices necessitates network operation strategies that provide interdevice cooperation for an efficient use of the limited battery power and spectral resources. For these reasons, the major difficulties for efficient multisensor localization and navigation in the IoT lie in fusing data and measurements collected from heterogeneous sensors with low computation and communication capabilities and in designing network operation strategies that can efficiently allocate resources in scenarios with insufficient infrastructure and limited battery power. Addressing these difficulties can overcome key issues in the current IoT networks, including the heterogeneity of sensing technologies and the limited capability of devices in terms of computation, communication, and battery energy.

The recently introduced paradigm of NLN [1] has important characteristics that are favorable for multisensor localization and navigation in IoT networks. In particular, it can provide technology-agnostic and low-complexity algorithms for heterogeneous multisensor fusion [14] and scalable network operation [15], which typically do not require much communication and computation overhead. An NLN scenario involving five devices and three anchors is shown in Figure 1(a). Figure 1(b) shows devices of Peregrine, a system developed for a three-dimensional (3-D) NLN.

This article provides an overview of how IoT location awareness can be enabled by the NLN paradigm.

- We present a framework for developing scalable and distributed inference algorithms for localization in IoT networks.
- We devise centralized and distributed network operation strategies that can increase battery lifetime and localization accuracy.
- We demonstrate that multisensor fusion and cooperation among devices can dramatically increase localization performance in a large-scale scenario with hundreds of mobile agents.
- We quantify how network operation algorithms can reduce the communication overhead and energy consumption of localization networks.

## Notation

Random variables (RVs) are displayed in sans serif, upright fonts; their realizations in serif, italic fonts. Vectors and matrices are denoted by bold lowercase and uppercase letters, respectively. Sets are denoted by calligraphic font. For example, an RV and its realization are denoted by $\mathsf{x}$ and $x$, respectively; a random vector and its realization are denoted by $\mathbf{x}$ and $\boldsymbol{x}$, respectively; a set is denoted by $\mathcal{X}$. The identity matrix is denoted by $\boldsymbol{I}$. For the probability distribution function (PDF) of the random vector $\mathbf{x}$, at $\boldsymbol{x}$, the short notation $f(\boldsymbol{x}) = f_{\mathbf{x}}(\boldsymbol{x})$ is used. Furthermore, $\boldsymbol{x} = [\boldsymbol{x}_i]_{i \in \mathcal{I}}$ denotes vector that is obtained by arranging all the subvectors $\boldsymbol{x}_i, i \in \mathcal{I}$ in an arbitrary but known order into a column vector. Finally, the notations of important quantities that are used throughout the article are summarized in Table 1.

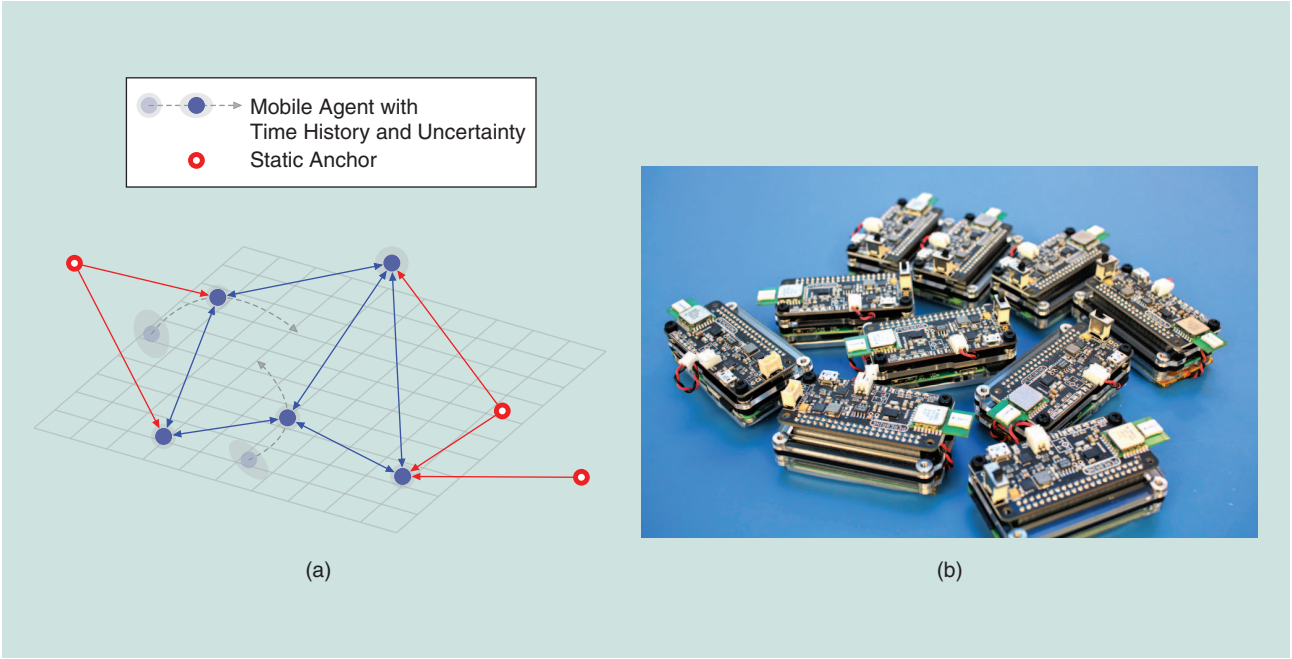


**FIGURE 1.** (a) A graphical depiction of an NLN scenario involving five devices and three anchors. (b) The devices used in the Peregrine, a system for a 3-D NLN [16].

**Table 1. Notations of important quantities.**

| Notation | Definition | Notation | Definition |
|---|---|---|---|
| $\mathcal{N}_a$ | The index set of mobile agents | $\mathcal{N}_b$ | The index set of anchors |
| $\mathbf{x}_i^{(n)}$ | The positional state of the $i$th node at time $n$ | $\mathbf{p}_i^{(n)}$ | The position of the $i$th node at time $n$ |
| $\mathbf{z}_{ij}^{(n)}$ | An internode measurement between $i$th agent and $j$th node at time $n$ | $\mathbf{z}_i^{(n)}$ | All the internode measurements of the $i$th agent at time $n$ |
| $\mathbf{x}_i^{(0:n)}$ | All the positional states of the $i$th node up to time $n$ | $\mathbf{z}_i^{(1:n)}$ | All the measurements of the $i$th agent up to time $n$ |
| $\alpha_f(\boldsymbol{x}^{(n)})$ | The message passed from variable node $\boldsymbol{x}$ to factor node $f$ | $\beta_f(\boldsymbol{x}^{(n)})$ | The message passed from factor node $f$ to variable node $\boldsymbol{x}$ |
| $\boldsymbol{\mu}_{\mathrm{p}}^{(n)}$ | The predicted mean vector | $\Sigma_{\mathrm{p}}^{(n)}$ | The predicted covariance matrix |
| $\boldsymbol{\mu}^{(n)}$ | The posterior mean vector | $\Sigma^{(n)}$ | The posterior covariance matrix |
| $\bar{\mathbf{x}}_i^{(n)}$ | The augmented state vector | $\bar{\mathbf{z}}_i^{(n)}$ | The augmented measurement vector |
| $\boldsymbol{Q}^{(n)}$ | The localization error matrix | $\boldsymbol{J}^{(n)}$ | The Fisher information matrix |
| $\mathcal{P}_{\mathrm{NA}}^{(n)}$ | The optimization problem for node activation | $\mathcal{P}_{\mathrm{NP}}^{(n)}$ | The optimization problem for node prioritization |
| $\zeta_i^{(n)}$ | The channel access probability of agent $i$ | $y_{ij}^{(n)}$ | The amount of resources allocated to the measurement link pair $(i, j)$ |
| $\chi_i^{(n)}$ | The potential error reduction of agent $i$ related to internode measurements | $\xi_{ij}^{(n)}$ | The channel quality between nodes $i$ and $j$ |

## Single-node localization for IoT

This section revises localization and navigation algorithms for single-node scenarios. First, consider a network of IoT devices that consists of a mobile agent (with index set $\mathcal{N}_a = \{1\}$) and of $N_b$ anchors at known positions (with index set $\mathcal{N}_b = \{2, 3, ..., N_b + 1\}$). The agents are localized based on heterogeneous sensor measurements by using the anchors as reference points. Measurements for localization are made at discrete time steps indexed by $n = 1, 2, ..., N$. Let $\mathbf{x}_1^{(n)} \in \mathbb{R}^D$ be the unknown positional state of the agent at time $n$, which includes the position $\mathbf{p}_1^{(n)}$ and other mobility parameters such as velocity, acceleration, orientation, and angular velocity. All measurements made at time $n$ are summarized in the vector $z_1^{(n)}$, which is the concatenation of all internode measurements $z_{1j}^{(n)}$ with anchors $j \in \mathcal{N}_b$. The localization process is essentially the calculation of an estimate $\hat{\boldsymbol{x}}_1^{(n)}$ of $\mathbf{x}_1^{(n)}$ from all available measurements up to time $n$ (denoted as $z_1^{(1:n)} \triangleq [z_1^{(1)\mathrm{T}}, z_1^{(2)\mathrm{T}}, \ldots, z_1^{(n)\mathrm{T}}]^{\mathrm{T}}$).

The relationship of the current state vector with the previous state vector can be described by the state-evolution model

$$\mathbf{x}_1^{(n)} = \boldsymbol{a}(\mathbf{x}_1^{(n-1)}, \mathbf{c}_1^{(n)}; \boldsymbol{u}_1^{(n)}), \tag{1}$$

where $\mathbf{c}_1^{(n)}$ is the state-evolution noise vector that is assumed independent across time $n$ and $\boldsymbol{u}_1^{(n)}$ is a known input [17] that controls the motion of the agent. Note that the PDF $f(\boldsymbol{c}_1^{(n)})$ can be different for distinct time steps $n$. From the state-evolution model (1) one can directly obtain the state-evolution function $f\left(\boldsymbol{x}_1^{(n)} \middle| \boldsymbol{x}_1^{(n-1)}; \boldsymbol{u}_1^{(n)}\right)$. Note that (1) implies a Markov property, i.e., given $\boldsymbol{x}_1^{(n-1)}$, $\mathbf{x}_1^{(n)}$ is statistically independent of previous $\mathbf{x}_1^{(0)}, \mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_1^{(n-2)}$ and future $\mathbf{x}_1^{(n+1)}, \mathbf{x}_1^{(n+2)}, \ldots$ states. The joint prior PDF $f(\boldsymbol{x}_1^{(0)})$ at time $n = 0$ is known. The joint prior information for all times $0, 1, \ldots, n$, i.e., all available information before any measurement is performed, can now be expressed as

$$f(\boldsymbol{x}_1^{(0:n)}; \boldsymbol{u}_1^{(1:n)}) = f(\boldsymbol{x}_1^{(0)}) \prod_{k=1}^{n} f\left(\boldsymbol{x}_1^{(k)} \middle| \boldsymbol{x}_1^{(k-1)}; \boldsymbol{u}_1^{(k)}\right). \tag{2}$$

The relationship of the current measurements with the current state vector is described by the measurement model

$$\mathbf{z}_1^{(n)} = \boldsymbol{h}(\mathbf{x}_1^{(n)}, \mathbf{v}_1^{(n)}), \tag{3}$$

where $\mathbf{v}_1^{(n)}$ is the measurement noise, which is assumed independent across times $n$. Note that the PDF $f(\boldsymbol{v}_1^{(n)})$ can be different for distinct time steps $n$. From the measurement model (3) one can directly obtain the likelihood function $f\left(\boldsymbol{z}_1^{(n)} \middle| \boldsymbol{x}_1^{(n)}\right)$. Note that (3) implies that given $\boldsymbol{x}_1^{(n)}$, $\mathbf{z}_1^{(n)}$ is statistically independent of previous $\mathbf{x}_1^{(0)}, \mathbf{x}_1^{(1)}, \ldots, \mathbf{x}_1^{(n-1)}$ and of future $\mathbf{x}_1^{(n+1)}, \mathbf{x}_1^{(n+2)}, \ldots$ states, as well as of previous $\mathbf{z}_1^{(0)}, \mathbf{z}_1^{(1)}, \ldots, \mathbf{z}_1^{(n-1)}$ and future $\mathbf{z}_1^{(n+1)}, \mathbf{z}_1^{(n+2)}, \ldots$ measurements. Therefore, the likelihood function for all times $1, 2, \ldots, n$ (i.e., all available information related to the performed measurements) can be expressed as

$$f\left(\boldsymbol{z}_1^{(1:n)} \middle| \boldsymbol{x}_1^{(1:n)}\right) = \prod_{k=1}^{n} f\left(\boldsymbol{z}_1^{(k)} \middle| \boldsymbol{x}_1^{(k)}\right). \tag{4}$$

By using Bayes' rules, (2) and (4), the joint posterior PDF of $\mathbf{x}_1^{(0:n)}$ given $\boldsymbol{z}_1^{(1:n)}$ for $n > 0$ results in

$$\begin{aligned} &f\left(\boldsymbol{x}_1^{(0:n)} \middle| \boldsymbol{z}_1^{(1:n)}; \boldsymbol{u}_1^{(1:n)}\right) \\ &\quad \propto f\left(\boldsymbol{z}_1^{(1:n)} \middle| \boldsymbol{x}_1^{(1:n)}\right) f(\boldsymbol{x}_1^{(0:n)}; \boldsymbol{u}_1^{(1:n)}) \\ &\quad = f(\boldsymbol{x}_1^{(0)}) \prod_{k=1}^{n} f\left(\boldsymbol{x}_1^{(k)} \middle| \boldsymbol{x}_1^{(k-1)}; \boldsymbol{u}_1^{(k)}\right) f\left(\boldsymbol{z}_1^{(k)} \middle| \boldsymbol{x}_1^{(k)}\right). \end{aligned} \tag{5}$$

The factor graph [18] representing this joint posterior for SBE is shown in Figure 2. For simplicity in notation, the index of the agent is dropped in the following, e.g., $\mathbf{x}_1^{(n)}$ is replaced by $\mathbf{x}^{(n)}$.

### A temporal fusion based on SBE

Temporal multisensor fusion in a Bayesian setting is accomplished by determining an estimate of $\mathbf{x}^{(n)}$ from the marginal posterior PDF $f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n)}\right)$. For example, the minimum mean-square-error (MMSE) estimate is given by [19]

$$\hat{\boldsymbol{x}}^{(n)}_{\mathrm{MMSE}} \triangleq \int \boldsymbol{x}^{(n)} f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n)};\boldsymbol{u}^{(1:n)}\right)\mathrm{d}\boldsymbol{x}^{(n)}. \tag{6}$$

The marginal posterior PDF $f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n)};u^{(1:n)}\right)$ in (6) can be obtained from the joint posterior PDF $f\left(\boldsymbol{x}^{(1:n)}\middle|z^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ in (5) by marginalization. However, direct marginalization of $f\left(\boldsymbol{x}^{(1:n)}\middle|z^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ is unfeasible in general because it relies on integration over a state space whose dimension grows with the time $n$.

This problem known as the *curse of dimensionality* [20], can be addressed by SBE [12] if the joint posterior PDF $f\left(\boldsymbol{x}^{(1:n)}\middle|z^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ has a structure like (5). The exact calculation of $f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ is then possible sequentially; at each time $n$, SBE consists of the prediction step

$$\begin{aligned} &f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n-1)};\boldsymbol{u}^{(1:n)}\right)\\ &=\int f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{x}^{(n-1)};\boldsymbol{u}^{(n)}\right)f\left(\boldsymbol{x}^{(n-1)}\middle|z^{(1:n-1)};\boldsymbol{u}^{(1:n-1)}\right)\mathrm{d}\boldsymbol{x}^{(n-1)}, \end{aligned} \tag{7}$$

which is followed by the update step

$$f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n)};\boldsymbol{u}^{(1:n)}\right)\propto f\left(z^{(n)}\middle|\boldsymbol{x}^{(n)}\right)f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n-1)};\boldsymbol{u}^{(1:n)}\right). \tag{8}$$

Contrary to direct marginalization in which integration is performed over an $nD$-dimensional state space, SBE involves only operations in $D$-dimensional state spaces that are performed $n$ times. As a consequence, the complexity related to calculating $f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n-1)};\boldsymbol{u}^{(n)}\right)$ scales only linearly with the number of time steps $n$. Note that the information acquired by all sensors up to time $n$, is represented by the low-dimensional predicted posterior PDF $f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n-1)};\boldsymbol{u}^{(n)}\right)$ and temporal fusion is directly performed in the update step according to (8).

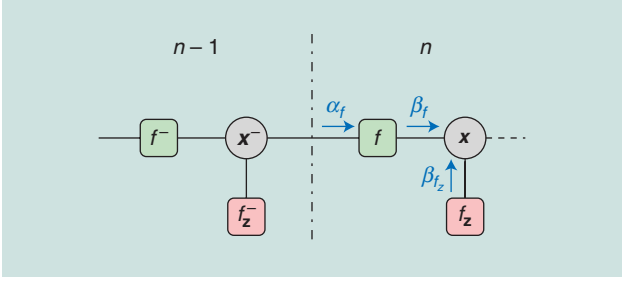


**FIGURE 2.** A factor graph for single-node localization representing the factorization in (5). Nodes in green represent factors related to the state-evolution function, nodes in red represent factors related to the likelihood function, while messages related to the SPA are in blue. The following short notations are used: $\boldsymbol{x}^- \triangleq \boldsymbol{x}_1^{(n-1)}$, $\boldsymbol{x} \triangleq \boldsymbol{x}_1^{(n)}$, $f^- \triangleq f\left(\boldsymbol{x}_1^{(n-1)}\middle|\boldsymbol{x}_1^{(n-2)};\boldsymbol{u}_1^{(n-1)}\right)$, $f \triangleq f\left(\boldsymbol{x}_1^{(n)}\middle|\boldsymbol{x}_1^{(n-1)};\boldsymbol{u}_1^{(n)}\right)$, $f_{\boldsymbol{z}}^- \triangleq f\left(\boldsymbol{z}_1^{(n-1)}\middle|\boldsymbol{x}_1^{(n-1)}\right)$, $f_{\boldsymbol{z}} \triangleq f\left(\boldsymbol{z}_1^{(n)}\middle|\boldsymbol{x}_1^{(n)}\right)$, $\alpha_f \triangleq \alpha_f(\boldsymbol{x}_1^{(n-1)})$, $\beta_f \triangleq \beta_f(\boldsymbol{x}_1^{(n)})$, and $\beta_{f_{\boldsymbol{z}}} \triangleq \beta_{f_{\boldsymbol{z}}}(\boldsymbol{x}_1^{(n)})$.

### Message-passing interpretation of SBE

For an arbitrary estimation problem, the sum-product algorithm (SPA) [18] can calculate exact or approximate marginal posterior PDFs in an efficient manner. In particular, the SPA avoids the curse of dimensionality inherent to direct marginalization. Therefore, SPA-based solutions are attractive for high-dimensional inference problems. The SPA is a message-passing algorithm since its basic operations can be interpreted as an exchange of statistical information on adjacent nodes of a factor graph, i.e., as messages passed along the edges of the graph.

If the factor graph is tree structured, such as the one shown in Figure 2, message updates are performed only once for each node in the graph. The message-passing procedure begins at the variable and factor nodes with only one edge (which passes a constant message and the corresponding factor, respectively) and continues with those nodes where all incoming messages are computed already. According to the SPA message-passing rules, in a factor graph as shown in Figure 2, the message passed from factor node $f$ to variable node $\boldsymbol{x}$ is obtained as [18]

$$\beta_f(\boldsymbol{x}^{(n)}) = \int f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{x}^{(n-1)};\boldsymbol{u}^{(n)}\right)\alpha_f(\boldsymbol{x}^{(n-1)})\mathrm{d}\boldsymbol{x}^{(n-1)}, \tag{9}$$

where $\alpha_f(\boldsymbol{x}^{(n-1)})$ is the message passed from variable node $\boldsymbol{x}^-$ to factor node $f$. Furthermore, the message passed from $f_{\boldsymbol{z}}$ to $\boldsymbol{x}$ is given by $\beta_{f_{\boldsymbol{z}}}(\boldsymbol{x}^{(n)}) = f\left(z^{(n)}\middle|\boldsymbol{x}^{(n)}\right)$. After these two messages are calculated, the belief for $\boldsymbol{x}$ is finally obtained as

$$b(\boldsymbol{x}^{(n)}) \propto \beta_f(\boldsymbol{x}^{(n)})\beta_{f_{\boldsymbol{z}}}(\boldsymbol{x}^{(n)}). \tag{10}$$

For $\alpha_f(\boldsymbol{x}^{(n-1)}) = b(\boldsymbol{x}^{(n-1)})$, it can be seen that $b(\boldsymbol{x}^{(n)}) = f\left(\boldsymbol{x}^{(n)}\middle|z^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ as provided by SBE. Thus, SBE based on prediction and update steps, respectively (7) and (8), is equivalent to calculating the belief $b(\boldsymbol{x}^{(n)})$ by running the SPA on the factor graph in Figure 2.

### Node localization and navigation algorithms

A large variety of filtering algorithms suitable for node localization and navigation are based on SBE according to (7) and (8). Here, we focus on two widely adopted techniques: Kalman filtering and particle filtering.

#### The Kalman filter

Consider the case where the state-evolution model and the measurement model are linear, i.e., (1) and (3) can be expressed as

$$\mathbf{x}^{(n)} = \boldsymbol{A}\mathbf{x}^{(n-1)} + \boldsymbol{B}\boldsymbol{u}^{(n)} + \mathbf{c}^{(n)} \tag{10a}$$

$$\mathbf{z}^{(n)} = \boldsymbol{H}\mathbf{x}^{(n)} + \mathbf{v}^{(n)}, \tag{10b}$$

where the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{H}$ are assumed known. Furthermore, the noise $\mathbf{c}^{(n)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{c}}^{(n)})$ and $\mathbf{v}^{(n)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{v}}^{(n)})$ is Gaussian distributed with noise covariance matrices $\Sigma_{\mathbf{c}}^{(n)}$ and $\Sigma_{\mathbf{v}}^{(n)}$. In this case, closed-form solutions for the prediction (7) and update step (8) of SBE can be obtained. These

closed-from expressions are used within the Kalman filter (KF) [19] that represents posterior PDFs $f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ by second-order statistics, i.e., by means $\boldsymbol{\mu}^{(n)}$ and covariance matrices $\Sigma^{(n)}$. If the prior $f(\boldsymbol{x}^{(0)})$ is also Gaussian, the PDFs $f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n-1)};\boldsymbol{u}^{(1:n)}\right)$ and $f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ are Gaussian as well for arbitrary $n$. In that case, the KF can provide the optimum solution and the exact MMSE estimator $\hat{\boldsymbol{x}}^{(n)}_{\mathrm{MMSE}}$ in (6) is given by $\boldsymbol{\mu}^{(n)}$. The KF consists of two steps: In the prediction step of the KF, the predicted mean $\boldsymbol{\mu}^{(n)}_{\mathrm{p}}$ and covariance matrix $\boldsymbol{\Sigma}^{(n)}_{\mathrm{p}}$ that fully characterize $f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n-1)};\boldsymbol{u}^{(1:n)}\right)$ are calculated based on (10a). In the update step of the KF, first the mean $\boldsymbol{\mu}^{(n)}_{\mathbf{z}}$, the covariance matrix $\Sigma^{(n)}_{\mathbf{z}}$, and the cross-covariance matrix $\Sigma^{(n)}_{\mathbf{xz}}$ are calculated based on (10b), then the posterior mean $\boldsymbol{\mu}^{(n)}$ and posterior covariance matrix $\boldsymbol{\Sigma}^{(n)}$ are obtained using the Kalman update equations [19]. For nonlinear non-Gaussian models inherent to multisensor localization, computationally feasible approximate algorithms include variants of the KF, such as the extended KF (EKF) [19] and the unscented KF (UKF) [11].

The EKF and the UKF are versions of the KF that are suitable for nonlinear state-evolution and measurement models. If $\boldsymbol{a}(\mathbf{x}^{(n-1)}, \mathbf{c}^{(n)}; \mathbf{u}^{(n)})$ in (1) and $\boldsymbol{h}(\mathbf{x}^{(n)}, \mathbf{v}^{(n)})$ in (3) are nonlinear functions, the covariance matrices $\boldsymbol{\Sigma}^{(n)}_{\mathrm{p}}$ as well as $\boldsymbol{\Sigma}^{(n)}_{\mathbf{z}}$ and $\boldsymbol{\Sigma}^{(n)}_{\mathbf{xz}}$ cannot be calculated directly. The EKF and the UKF are still based on the Kalman update equations but perform different approximations to obtain these matrices.

In the EKF, a multivariate Taylor series expansion of (1) and (3) is used to linearize them around $[\boldsymbol{\mu}^{(n-1)T}, \mathbf{0}^T]^T$ and $[\boldsymbol{\mu}^{(n)T}_{\mathrm{p}}, \mathbf{0}^T]^T$, respectively [19]. In this way, an approximation of the matrices $\boldsymbol{\Sigma}^{(n)}_{\mathrm{p}}$, $\boldsymbol{\Sigma}^{(n)}_{\mathbf{z}}$, and $\boldsymbol{\Sigma}^{(n)}_{\mathbf{xz}}$ is obtained. While the EKF is widely adopted, it is accurate only if the system model is moderately nonlinear. Furthermore, the EKF is challenging to implement and difficult to tune. The UKF is a widely adopted solution for applications in which the EKF is not accurate or (1) and (3) are not differentiable. The UKF performs approximate inference by using a minimal set of deterministically chosen samples referred to as *sigma points* (*SPs*) [11]. The nonlinear model (1) and (3) is evaluated at the SPs and from the resulting new SPs, approximate second-order statistics $\boldsymbol{\mu}^{(n)}_{\mathrm{p}}, \boldsymbol{\Sigma}^{(n)}_{\mathrm{p}}$ as well as $\boldsymbol{\mu}^{(n)}_{\mathbf{z}}, \boldsymbol{\Sigma}^{(n)}_{\mathbf{z}}$, and $\boldsymbol{\Sigma}^{(n)}_{\mathbf{xz}}$ are calculated [11]. The UKF can often provide approximations of $\boldsymbol{\mu}^{(n)}$ and $\boldsymbol{\Sigma}^{(n)}$ that are more accurate compared to those provided by the EKF at a comparable computational complexity.

### The particle filter

The particle filter (PF) is an attractive alternative to the EKF and the UKF for applications in which a representation of $f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ using second-order statistics is not accurate. This might be the case if the state-evolution and/or measurement model are highly nonlinear and $f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ is multimodal. The key idea of PFs is to represent the posterior distribution by a set of samples (particles) with associated weights, i.e.,

$$\tilde{f}\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right) \approx \sum_{l=1}^{n_{\mathrm{p}}} w_l^{(n)} \delta(\boldsymbol{x}^{(n)} - \boldsymbol{x}_l^{(n)}), \tag{11}$$

where $n_{\mathrm{p}}$ is the number of particles, $\delta(\cdot)$ is the Dirac delta function, $w_l^{(n)} \geqslant 0$ is the weight of the $l$th particle $\boldsymbol{x}_l^{(n)}$ at time index $n$, and $\Sigma_{l=1}^{n_{\mathrm{p}}} w_l^{(n)} = 1$. Note that the number of randomly sampled particles $n_{\mathrm{p}}$ is typically significantly larger compared to the number of deterministically calculated SPs $n_{\mathrm{s}}$ used in the UKF.

An approximation of the MMSE estimate in (6) is given by the mean of $\tilde{f}\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ in (11), which is equal to the mean of the weighted particles, i.e.,

$$\hat{\boldsymbol{x}}^{(n)} = \int \boldsymbol{x}^{(n)} \tilde{f}\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right) \mathrm{d}\boldsymbol{x}^{(n)} = \sum_{l=1}^{n_{\mathrm{p}}} w_l^{(n)} \boldsymbol{x}_l^{(n)}. \tag{12}$$

A large variety of particle-filtering algorithms have been introduced. In what follows, we review the prominent sequential importance resampling filter [12], which consists of three steps referred to as *sampling*, *weight update*, and *resampling*.

The sampling step corresponds to the prediction step of SBE in (7). For each particle $\boldsymbol{x}_l^{(n-1)}$, a new particle $\boldsymbol{x}_l^{(n)}$ is drawn from the state-evolution PDF $f\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{x}^{(n-1)};\boldsymbol{u}^{(n)}\right)$ evaluated at $\boldsymbol{x}_l^{(n-1)}$. The weight update step corresponds to the update step of SBE in (8). For each particle $\boldsymbol{x}_l^{(n)}$ the updated weight $w_l^{(n)}$ is obtained as $w_l^{(n)} = f\left(\boldsymbol{z}^{(n)}\middle|\boldsymbol{x}_l^{(n)}\right) \Big/ \Sigma_{\ell=1}^{n_{\mathrm{p}}} f\left(\boldsymbol{z}^{(n)}\middle|\boldsymbol{x}_\ell^{(n)}\right)$. Then, particle-based state estimation is performed as in (12). The resampling step is a step that is performed to avoid degeneracy of particles. It is typically executed only if an indicator called the *effective sample size* is smaller than a threshold. In the resampling step, $n_{\mathrm{p}}$ resampled particles are obtained by sampling from $\tilde{f}\left(\boldsymbol{x}^{(n)}\middle|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\right)$ in (11) and setting the weight of the resampled particles to $1/n_{\mathrm{p}}$, with resampled particles used at time $n+1$.

#### *Remark 1*

Most PFs are optimum in the sense that for $n_{\mathrm{p}} \to \infty$ the estimate $\hat{\boldsymbol{x}}^{(n)}$ in (12) converges to the true MMSE estimate $\hat{\boldsymbol{x}}^{(n)}_{\mathrm{MMSE}}$ in (6). Contrary to EKF and the UKF, PFs are also suitable for highly nonlinear SBE problems. However, their computational complexity is significantly increased compared to variants of the KF. In certain settings, PFs can avoid the curse of dimensionality [20]. However, they do not scale well with the dimension of the state to be estimated and are not directly amendable for distributed implementations.

## Network localization for the IoT

Consider the localization of a network of IoT devices that consists of $N_{\mathrm{a}}$ agents (with index set $\mathcal{N}_{\mathrm{a}} = \{1,2,\ldots,N_{\mathrm{a}}\}$) and $N_{\mathrm{b}}$ anchors (with index set $\mathcal{N}_{\mathrm{b}} = \{N_{\mathrm{a}}+1, N_{\mathrm{a}}+2,\ldots,N_{\mathrm{a}}+N_{\mathrm{b}}\}$). Let $\mathbf{x}_i^{(n)} \in \mathbb{R}^D$ be the positional state of the node $i \in \{1,2,\ldots, N_{\mathrm{a}}+N_{\mathrm{b}}\}$. The states of all nodes are represented by the joint state vector $\mathbf{x}^{(n)} \triangleq [\mathbf{x}_1^{(n)\mathrm{T}}, \mathbf{x}_2^{(n)\mathrm{T}}, \ldots, \mathbf{x}_{N_{\mathrm{a}}+N_{\mathrm{b}}}^{(n)\mathrm{T}}]^{\mathrm{T}}$. At time $n$, agent $i \in \mathcal{N}_{\mathrm{a}}$ is able to communicate and perform an internode measurement $\boldsymbol{z}_{ij}^{(n)}$ with nodes $j$ in its neighbor set $\mathcal{A}_i^{(n)}$. For anchors $i \in \mathcal{N}_{\mathrm{b}}$, the neighbor set is empty, i.e., $\mathcal{A}_i^{(n)} = \emptyset$. Agent communication is symmetric, i.e., for $i,j \in \mathcal{N}_{\mathrm{a}}$, $j \in \mathcal{A}_i^{(n)}$ implies $i \in \mathcal{A}_j^{(n)}$. All measurements performed by all agents $i \in \mathcal{N}_{\mathrm{a}}$ at time $n$ are summarized in the joint measurement vector $\boldsymbol{z}^{(n)}$. Every agents aims to calculate an estimate $\hat{\boldsymbol{x}}_i^{(n)}$ of $\mathbf{x}_i^{(n)}$ from all available measurements $\boldsymbol{z}^{(1:n)}$ collected up to time $n$.

For node $i$ at time $n$, the relationship of the current state vector $\mathbf{x}_i^{(n)}$ with the previous state vector $\mathbf{x}_i^{(n-1)}$ is given by the state-evolution model

$$\mathbf{x}_i^{(n)} = \boldsymbol{a}_i(\mathbf{x}_i^{(n-1)}, \mathbf{c}_i^{(n)}; \boldsymbol{u}_i^{(n)}) \tag{13}$$

where the state-evolution noise vector $\mathbf{c}_i^{(n)}$ is assumed independent across $n$ and $i$. Note that the PDF $f(\boldsymbol{c}_i^{(n)})$ can be different for distinct time steps $n$ and agent indexes $i$. In particular, for anchors $i \in \mathcal{N}_b$ it is assumed that $f(\boldsymbol{c}_i^{(n)}) = \delta(\boldsymbol{c}_i^{(n)})$, i.e., $\boldsymbol{c}_i^{(n)}$ is deterministic and equal to zero. From the state-evolution model (13) one can directly obtain the state-evolution function $f\left(\boldsymbol{x}_i^{(n)} \middle| \boldsymbol{x}_i^{(n-1)}; \boldsymbol{u}_i^{(n)}\right)$. At $n = 0$, the prior PDF of the joint state vector can be expressed as $f(\boldsymbol{x}^{(0)}) = \prod_{i=1}^{N_a+N_b} f(\boldsymbol{x}_i^{(0)})$. In particular, anchors $i \in \mathcal{N}_b$ have perfect knowledge of their state, i.e., their prior PDFs are given by $f(\boldsymbol{x}_i^{(0)}) = \delta(\boldsymbol{x}_i^{(0)} - \tilde{\boldsymbol{x}}_i^{(0)})$ where $\tilde{\boldsymbol{x}}_i^{(0)}$ is the true state. Furthermore, agents have uninformative prior information $f(\boldsymbol{x}_i^{(0)})$ that is assumed known. For $n > 0$, the joint prior PDF, can be expressed as

$$\begin{aligned} f(\boldsymbol{x}^{(0:n)}; \boldsymbol{u}^{(1:n)}) &= f(\boldsymbol{x}^{(0)}) \prod_{k=1}^{n} f\left(\boldsymbol{x}^{(k)} \middle| \boldsymbol{x}^{(k-1)}; \boldsymbol{u}^{(k)}\right) \\ &= \prod_{i=1}^{N_a+N_b} f(\boldsymbol{x}_i^{(0)}) \prod_{k=1}^{n} f\left(\boldsymbol{x}_i^{(k)} \middle| \boldsymbol{x}_i^{(k-1)}; \boldsymbol{u}_i^{(k)}\right). \end{aligned} \tag{14}$$

Agents $i \in \mathcal{N}_a$ performs internode measurements $\mathbf{z}_{ij}^{(n)}$, $j \in \mathcal{A}_i^{(n)}$ that are related to the states $\mathbf{x}_i^{(n)}$ and $\mathbf{x}_j^{(n)}$ as

$$\mathbf{z}_{ij}^{(n)} = \boldsymbol{h}_{ij}(\mathbf{x}_i^{(n)}, \mathbf{x}_j^{(n)}, \mathbf{v}_{ij}^{(n)}), \tag{15}$$

where $\mathbf{v}_{ij}^{(n)}$ is the internode measurement noise. Note that the PDF $f(\boldsymbol{v}_{ij}^{(n)})$ can be different for distinct time steps $n$ and agent indexes $i$, and is typically a function of the channel quality $\xi_{ij}^{(n)}$ (see the "Node Prioritization" section).

The measurement noise $\mathbf{v}_{ij}^{(n)}$ is assumed independent across all $(i, j)$ pairs and all times $n$. From the measurement model (15), one can directly obtain the likelihood function $f\left(\boldsymbol{z}_{ij}^{(n)} \middle| \boldsymbol{x}_i^{(n)}, \boldsymbol{x}_j^{(n)}\right)$. The joint likelihood function can be expressed as

$$f\left(\boldsymbol{z}^{(1:n)} \middle| \boldsymbol{x}^{(1:n)}\right) = \prod_{k=1}^{n} \prod_{i=1}^{N_a} \prod_{j \in \mathcal{A}_i^{(k)}} f\left(\boldsymbol{z}_{ij}^{(k)} \middle| \boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\right). \tag{16}$$

Using Bayes' rules together with (14) and (16), the joint posterior PDF of $\mathbf{x}^{(0:n)}$ given $\boldsymbol{z}^{(1:n)}$ for $n > 0$ is obtained as

$$\begin{aligned} &f\left(\boldsymbol{x}^{(0:n)} \middle| \boldsymbol{z}^{(1:n)}; \boldsymbol{u}^{(1:n)}\right) \\ &\quad \propto f\left(\boldsymbol{z}^{(1:n)} \middle| \boldsymbol{x}^{(1:n)}\right) f(\boldsymbol{x}^{(0:n)}; \boldsymbol{u}^{(1:n)}) \\ &\quad = f(\boldsymbol{x}^{(0)}) \prod_{k=1}^{n} f\left(\boldsymbol{x}^{(k)} \middle| \boldsymbol{x}^{(k-1)}; \boldsymbol{u}^{(k)}\right) f\left(\boldsymbol{z}^{(k)} \middle| \boldsymbol{x}^{(k)}\right) \end{aligned} \tag{17}$$

$$\begin{aligned} &= \prod_{i=1}^{N_a+N_b} f(\boldsymbol{x}_i^{(0)}) \prod_{k=1}^{n} f\left(\boldsymbol{x}_i^{(k)} \middle| \boldsymbol{x}_i^{(k-1)}; \boldsymbol{u}_i^{(k)}\right) \\ &\quad \times \prod_{j \in \mathcal{A}_i^{(k)}} f\left(\boldsymbol{z}_{ij}^{(k)} \middle| \boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\right). \end{aligned} \tag{18}$$

*Remark 2*

Note that the factorization of the marginal posterior in (17) has the same temporal structure as the marginal posterior in the single-node localization and navigation problem. The factor graph representing the factorization of the marginal posterior in (18) is shown in Figure 3. The spatiotemporal structure of the marginal posterior allows development of distributed-inference algorithms that are scalable both in time $n$ and in the number of agents $N_a$ as discussed in the next section.

## Spatiotemporal fusion based on the SPA

In a network with multiple agents, state estimation is complicated by the fact that, since internode measurements are performed, the posterior distributions $f\left(\boldsymbol{x}_i^{(n)} \middle| \boldsymbol{z}^{(1:n)}; \boldsymbol{u}^{(1:n)}\right)$ of agents are coupled and thus should be estimated jointly. A naive approach to joint sequential state estimation would be to only exploit the temporal structure of the joint posterior PDF $f\left(\boldsymbol{x}^{(0:n)} \middle| \boldsymbol{z}^{(1:n)}; \boldsymbol{u}^{(1:n)}\right)$ in (17) to obtain a marginal posterior PDF $f\left(\boldsymbol{x}^{(n)} \middle| \boldsymbol{z}^{(1:n)}; \boldsymbol{u}^{(1:n)}\right)$ by means of an algorithm presented in the "Node Localization and Navigation Algorithms" section and then calculating an estimate for the joint agent state $\boldsymbol{x}^{(n)}$. However, this approach is not scalable, as the dimension of $\boldsymbol{x}^{(n)}$ increases with the number of agents $N_a$. In addition, it is not amenable for a distributed implementation because it necessitates the existence of a fusion center that collects all pairwise measurements performed in the network.

Alternatively, distributed and scalable estimation can be performed by running SPA on the factor graph shown in Figure 3. In the case of a factor graph with loops, the beliefs produced by the SPA are generally only approximations of the marginal posterior PDFs and they typically suffer from overconfidence (in the sense that the uncertainty of the estimates is underestimated by their spread). Furthermore, there is no fixed order for message calculation in loopy SPA, and different orders may lead to different beliefs. This means that there is a certain freedom to design the order of messages in the development of SPA algorithms.

The message-passing rules presented next are obtained by 1) applying SPA [18] to the factor graph in Figure 3, 2) performing temporal fusion by sending messages only forward in time, and 3) performing only a single message-passing iteration in the spatial fusion step. In the temporal fusion step at agent $i$ and time $n$, since messages are sent only forward in time, the messages $\alpha_{f_i}(\boldsymbol{x}_i^{(n-1)})$ are equal to the beliefs computed at $n-1$, i.e., [18]

$$\alpha_{f_i}(\boldsymbol{x}_i^{(n-1)}) = b(\boldsymbol{x}_i^{(n-1)}). \tag{19}$$

Therefore, the messages $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$ can be obtained as

$$\begin{aligned} \beta_{f_i}(\boldsymbol{x}_i^{(n)}) &= \int f\left(\boldsymbol{x}_i^{(n)} \middle| \boldsymbol{x}_i^{(n-1)}; \boldsymbol{u}_i^{(n)}\right) \alpha_{f_i}(\boldsymbol{x}_i^{(n-1)}) \mathrm{d}\boldsymbol{x}_i^{(n-1)} \\ &= \int f\left(\boldsymbol{x}_i^{(n)} \middle| \boldsymbol{x}_i^{(n-1)}; \boldsymbol{u}_i^{(n)}\right) b(\boldsymbol{x}_i^{(n-1)}) \mathrm{d}\boldsymbol{x}_i^{(n-1)}. \end{aligned} \tag{20}$$

Note that the calculation of the message $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$ in the temporal fusion step is equivalent to the prediction step of SBE in (7) and its SPA interpretation in (9).

In the spatial fusion step, since only a single message-passing iteration is performed, outgoing messages $\alpha_{f_{ij}}(\boldsymbol{x}_i^{(n)}), i \in \mathcal{A}_j$ passed from variable node $\boldsymbol{x}_i$ to factor nodes $f_{ij}$ are directly given by $\alpha_{f_{ij}}(\boldsymbol{x}_i^{(n)}) = \beta_{f_i}(\boldsymbol{x}_i^{(n)})$. Furthermore, incoming messages $\beta_{f_{ij}}(\boldsymbol{x}_i^{(n)}), j \in \mathcal{A}_i$ can be obtained as

$$\begin{aligned}\beta_{f_{ij}}(\boldsymbol{x}_i^{(n)}) &= \int f\left(\boldsymbol{z}_{ij}^{(n)} \middle| \boldsymbol{x}_i^{(n)}, \boldsymbol{x}_j^{(n)}\right) \alpha_{f_{ji}}(\boldsymbol{x}_j^{(n)}) \mathrm{d}\boldsymbol{x}_j^{(n)} \\ &= \int f\left(\boldsymbol{z}_{ij}^{(n)} \middle| \boldsymbol{x}_i^{(n)}, \boldsymbol{x}_j^{(n)}\right) \beta_{f_i}(\boldsymbol{x}_j^{(n)}) \mathrm{d}\boldsymbol{x}_j^{(n)}. \end{aligned} \tag{21}$$

Finally, the belief of an agent $i$ at time $n$ is calculated as

$$b(\boldsymbol{x}_i^{(n)}) \propto \beta_{f_i}(\boldsymbol{x}_i^{(n)}) \prod_{j \in \mathcal{A}_i^{(n)}} \beta_{f_{ij}}(\boldsymbol{x}_i^{(n)}). \tag{22}$$

The messages $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$ in (20) and the belief $b(\boldsymbol{x}_i^{(n)})$ in (22) are PDFs, i.e., they integrate to one. The belief $b(\boldsymbol{x}_i^{(n)}) \approx f\left(\boldsymbol{x}_i^{(n)} \middle| \boldsymbol{z}^{(1:n)}; \boldsymbol{u}^{(1:n)}\right)$ can now be used to calculate an estimate $\hat{\boldsymbol{x}}_i^{(n)}$ of the positional state of agent $i$ at time $n$. Note that for anchors $i \in \mathcal{N}_\mathrm{b}$, the belief and the messages are given by $b(\boldsymbol{x}_i^{(n)}) = \alpha_{f_i}(\boldsymbol{x}_i^{(n)}) = \beta_{f_i}(\boldsymbol{x}_i^{(n)}) = \delta(\boldsymbol{x}_i^{(n)} - \tilde{\boldsymbol{x}}_i^{(n)})$ and $\mathcal{A}_i^{(n)} = \emptyset$ for all $n$.

Contrary to SBE, which only exploits the temporal structure of the estimation problem, loopy SPA performed on the factor graph in Figure 3 also exploits spatial structure. Increasing the number of agents leads to additional variable nodes in the factor graph but not to a higher dimension of the exchanged SPA messages. Therefore, the curse of dimensionality in time $n$ and in network size $N_\mathrm{a} + N_\mathrm{b}$ is avoided. As will be discussed next, message passing according to (19)–(22) nearly automatically yields to a distributed implementation.

## Distributed network-localization algorithms

We now present a framework for designing network-localization algorithms that is based on a reformulation of SPA for spatiotemporal fusion (19)–(22) as local instances of SBE performed on each agent [5], [6]. Within this framework, spatiotemporal fusion is possible in a scalable and distributed way by directly applying arbitrary existing algorithms based on SBE,

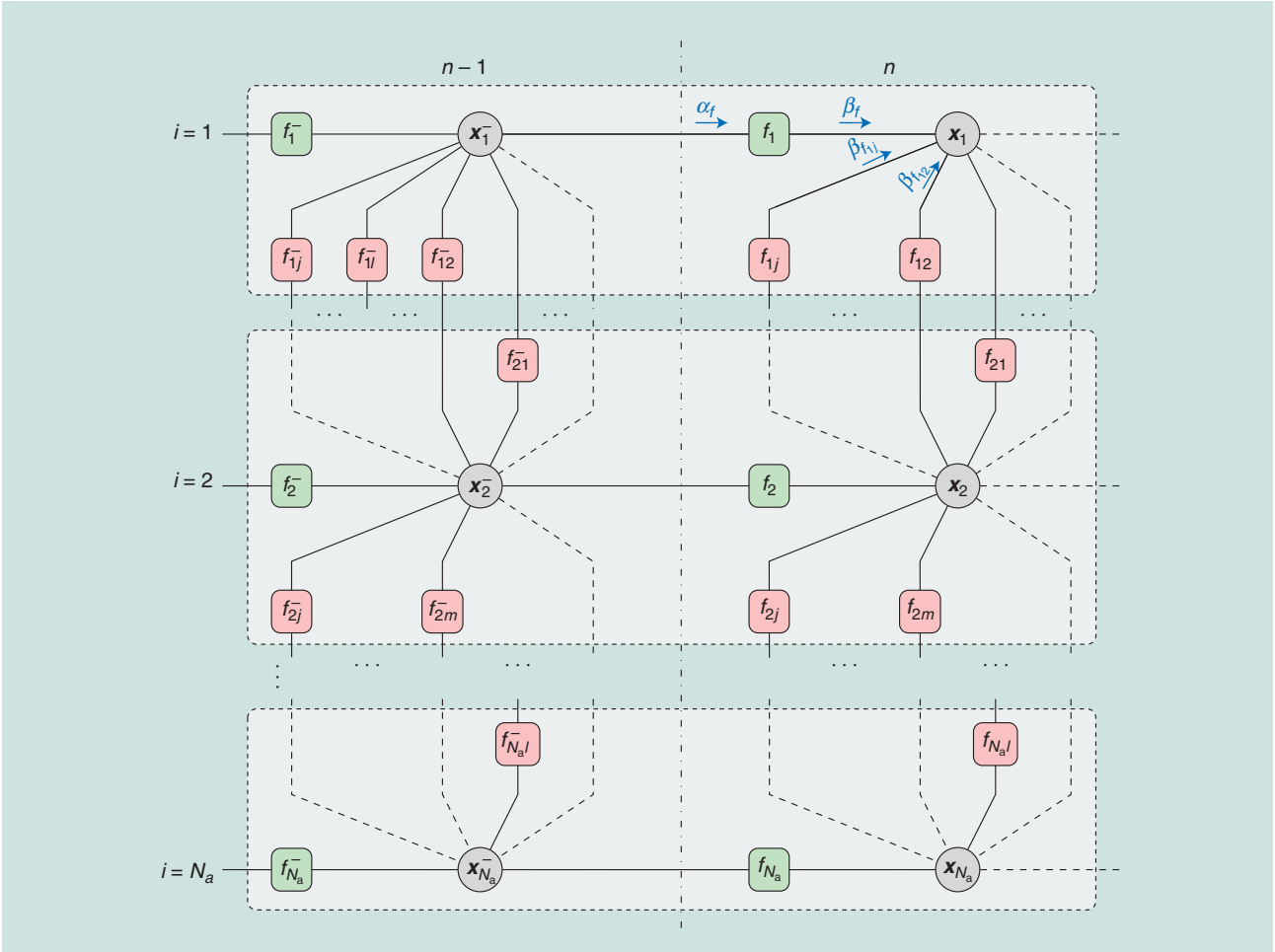


**FIGURE 3.** Two time steps of the factor graph for network localization corresponding to the factorizes (18). Nodes in green represent factors related to the state-evolution function, nodes in red represent factors related to the likelihood function, while SPA messages are in blue. The following short notations are used: $\boldsymbol{x}_{i^-} \triangleq \boldsymbol{x}_i^{(n-1)}$, $\boldsymbol{x}_i \triangleq \boldsymbol{x}_i^{(n)}$, $f_i \triangleq f\left(\boldsymbol{x}_i^{(n)} \middle| \boldsymbol{x}_i^{(n-1)}; \boldsymbol{u}_i^{(n)}\right)$, $f_{i^-} \triangleq f\left(\boldsymbol{x}_i^{(n-1)} \middle| \boldsymbol{x}_i^{(n-2)}; \boldsymbol{u}_i^{(n-1)}\right)$, $f_{ij} \triangleq f\left(\boldsymbol{z}_{ij}^{(n)} \middle| \boldsymbol{x}_i^{(n)}, \boldsymbol{x}_j^{(n)}\right)$, $f_{ij^-} \triangleq f\left(\boldsymbol{z}_{ij}^{(n-1)} \middle| \boldsymbol{x}_i^{(n-1)}, \boldsymbol{x}_j^{(n-1)}\right)$, $\alpha_{f_i} \triangleq \alpha_{f_i}(\boldsymbol{x}_i^{(n-1)})$, $\beta_{f_i} \triangleq \beta_{f_i}(\boldsymbol{x}_i^{(n)})$, and $\beta_{f_{ij}} \triangleq \beta_{f_{ij}}(\boldsymbol{x}_i^{(n)})$.

such as those reviewed in the "Node Localization and Navigation Algorithms" section.

Consider the spatiotemporal fusion at agent $i$, and introduce the augmented state vector $\bar{\mathbf{x}}_i^{(n)}$ and the augmented measurement $\bar{\mathbf{z}}_i^{(n)}$ as

$$\bar{\mathbf{x}}_i^{(n)} = [\mathbf{x}_j^{(n)}]_{j\in\{i\}\cup\mathcal{A}_i^{(n)}} \quad \bar{\mathbf{z}}_i^{(n)} = [\mathbf{z}_{ij}^{(n)}]_{j\in\mathcal{A}_i^{(n)}}.$$

Moreover, the belief $b(\bar{\boldsymbol{x}}_i^{(n)})$ of $\bar{\mathbf{x}}_i^{(n)}$ is introduced as

$$b(\bar{\boldsymbol{x}}_i^{(n)}) \propto f\left(\bar{\boldsymbol{z}}_i^{(n)} \middle| \bar{\boldsymbol{x}}_i^{(n)}\right) f(\bar{\boldsymbol{x}}_i^{(n)}), \tag{23}$$

where the "prior" $f(\bar{\boldsymbol{x}}_i^{(n)})$ and the "likelihood" function $f\left(\bar{\boldsymbol{z}}_i^{(n)} \middle| \bar{\boldsymbol{x}}_i^{(n)}\right)$ are given by

$$f(\bar{\boldsymbol{x}}_i^{(n)}) = \prod_{j\in\{i\}\cup\mathcal{A}_i^{(n)}} \beta_{f_j}(\boldsymbol{x}_j^{(n)}) \tag{24}$$

$$f\left(\bar{\boldsymbol{z}}_i^{(n)} \middle| \bar{\boldsymbol{x}}_i^{(n)}\right) = \prod_{j\in\mathcal{A}_i^{(n)}} f\left(\boldsymbol{z}_{ij}^{(n)} \middle| \boldsymbol{x}_i^{(n)}, \boldsymbol{x}_j^{(n)}\right). \tag{25}$$

Note that here, with an abuse of notation, control inputs $\boldsymbol{u}_i^{(k)}$ and measurements $\boldsymbol{z}_{ij}^{(k)}$ from previous time steps $k \in \{1, 2, \ldots, n-1\}$ are avoided. The expression (23) has the same form as the update step of SBE in (8).

By plugging (21) into (22) and subsequently swapping the order of multiplication and integration, (22) becomes

$$b(\boldsymbol{x}_i^{(n)}) = \int b(\bar{\boldsymbol{x}}_i^{(n)}) \mathrm{d}\bar{\boldsymbol{x}}_{\sim i}^{(n)}, \tag{26}$$

where $\bar{\boldsymbol{x}}_{\sim i}^{(n)}$ is the vector obtained by removing $\boldsymbol{x}_i^{(n)}$. from $\bar{\boldsymbol{x}}_i^{(n)}$.

Equations (23) and (26) indicate that $b(\boldsymbol{x}_i^{(n)})$ can be obtained via an update step (8) followed by marginalization. This observation motivates the following three steps at each agent $i \in \mathcal{N}_\mathrm{a}$ to perform spatiotemporal fusion by means of SPA.

- *Step 1*: *Local Prediction and Information Exchange.* Agent $i$ calculates $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$ locally according to (20) which is equivalent to the prediction step in (7). [The prediction step of any algorithm based on SBE, such as those presented in the "Message-Passing Interpretation of SBE" section, can be used to calculate $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$]. Then each agent broadcasts $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$ and receives $\beta_{f_j}(\boldsymbol{x}_j^{(n)})$ from its neighbors $j \in \mathcal{A}_i^{(n)}$ so that $f(\bar{\boldsymbol{x}}_i^{(n)})$ in (24) becomes available at agent $i$.
- *Step 2*: *Measurement Phase and State Update.* Agent $i$ cooperates with its neighbors $j \in \mathcal{A}_i^{(n)}$ to acquire internode measurements $\boldsymbol{z}_{ij}^{(n)}$. Now the likelihood function $f\left(\bar{\boldsymbol{z}}_i^{(n)} \middle| \bar{\boldsymbol{x}}_i^{(n)}\right)$ in (25) is available at agent $i$ and the belief $b(\bar{\boldsymbol{x}}_i^{(n)})$ of $\bar{\mathbf{x}}_i^{(n)}$ can be calculated locally by performing the update step in (23). Note that the update step of any algorithm based on SBE such as those presented in the "Message-Passing Interpretation of SBE" section can be used to calculate $b(\bar{\boldsymbol{x}}_i^{(n)})$.
- *Step 3*: *Marginalization.* In this step, agent $i$ computes the belief $b(\boldsymbol{x}_i^{(n)})$ from $b(\bar{\boldsymbol{x}}_i^{(n)})$. This typically incurs no computational overhead. For example, if $b(\bar{\boldsymbol{x}}_i^{(n)})$ is represented by the mean vector $\bar{\boldsymbol{\mu}}_i^{(n)}$ and the covariance matrix $\bar{\boldsymbol{\Sigma}}_i^{(n)}$, then the mean vector $\boldsymbol{\mu}_i^{(n)}$ and the covariance matrix $\boldsymbol{\Sigma}_i^{(n)}$ related to $b(\boldsymbol{x}_i^{(n)})$ can be directly extracted from $\bar{\boldsymbol{\mu}}_i^{(n)}$ and $\bar{\boldsymbol{\Sigma}}_i^{(n)}$, respectively. In case a particle representation $\{(\bar{\boldsymbol{x}}_{i,l}^{(n)}, w_{i,l}^{(n)})\}_{l=1}^{L}$ of the belief $b(\bar{\boldsymbol{x}}_i^{(n)})$ is available, a particle representation $\{(\boldsymbol{x}_{i,l}^{(n)}, w_{i,l}^{(n)})\}_{l=1}^{L}$ of the belief $b(\boldsymbol{x}_i^{(n)})$ can be obtained by discarding from the particles $\bar{\boldsymbol{x}}_{i,l}^{(n)}$ all subvectors $\boldsymbol{x}_{j,l}^{(n)}$ with $j \neq i$.

Note that the belief $b(\boldsymbol{x}_i^{(n)})$ can be calculated by only communicating with neighboring agents in the network. For accurate localization and navigation of an agent $i \in \mathcal{N}_\mathrm{a}$, typically only a small number of neighbors $\left|\mathcal{A}_i^{(n)}\right|$ are necessary. Therefore, the communication cost related to the information exchange in Step 1 as well as the computation cost related to calculating the beliefs $b(\bar{\boldsymbol{x}}_i^{(n)})$ remain feasible. More importantly, for a single agent $i \in \mathcal{N}_\mathrm{a}$, these costs only depend on the number of neighbors $\left|\mathcal{A}_j^{(n)}\right|$ but not on the network size $N_\mathrm{a} + N_\mathrm{b}$. An attractive property of calculating $b(\boldsymbol{x}_i^{(n)})$ by means of Steps 1–3 is that existing techniques for single-node localization and navigation can be directly leveraged for scalable and distributed network localization. Note that SP belief propagation (SPBP) [5] and the network-localization algorithm in [6] have been developed according to Steps 1–3.

## Efficient network operation

Network-operation strategies [21], [26] are indispensable for efficient localization and navigation in IoT scenarios. The network-operation strategies presented in this article focus on the coordination of measurements provided by range measurement units (RMUs), i.e., the measurement model in (15) is

$$\mathsf{z}_{ij}^{(n)} = \left\| \mathbf{x}_i^{(n)} - \mathbf{x}_j^{(n)} \right\| + \mathsf{v}_{ij}^{(n)}.$$

The performance of RMUs such as ultrawideband (UWB) radios is often limited by the fact that [16], [27], [28]:

1) Agents often make measurements with nodes with low link quality or poor geometry.
2) Different agents, which simultaneously transmit ranging signals, interfere with each other.

To address these issues, node-activation strategies to reduce interference and node prioritization strategies to allocate resources to measurements with neighbor nodes can be employed. A flowchart that visualizes the interaction of node activation, node prioritization, network localization, and the RMU is shown in Figure 4.

Note that, in what follows, the inverse Fisher information matrix [3] is referred to as an *error matrix*. In particular, all strategies developed in this article rely either on the individual error matrices $\boldsymbol{Q}_i^{(n)}$ related to the positions $\mathbf{p}_i^{(n)}$ of the agents $i \in \mathcal{N}_\mathrm{a}$ or on the joint error matrix $\boldsymbol{Q}^{(n)}$ related to the individual positions of all agents, as defined in [24]. These error matrices are not accessible in real-world localization systems as they rely on the knowledge of true positions. For this reason, in an implementation of the presented node-operation strategies [16], these error matrices are approximated by the corresponding covariance matrices, which can be provided by network-localization algorithms.

## Node activation

Node-activation strategies enable a significant reduction of packet collisions and localization errors in the network. The goal of node-activation strategies is to determine a set of nodes that are permitted to make range measurements so that packet collisions are avoided and the localization error reduction of the network is maximized. In what follows, we discuss centralized and distributed strategies for node activation.

### Centralized node activation

If agent $i$ is selected to make internode measurements with its neighbors at time $n$, the error evolution relationship is given by [24]

$$\boldsymbol{Q}^{(n+1)} = \left((\boldsymbol{Q}^{(n)})^{-1} + \sum_{j\in\mathcal{A}_i^{(n)}} \boldsymbol{S}_{ij}^{(n)}\right)^{-1} + \Delta^{(n+1)},$$

where $\boldsymbol{S}_{ij}^{(n)}$ denotes the information matrix corresponding to the measurement $z_{ij}^{(n+1)}$, and $\Delta^{(n+1)}$ denotes the matrix corresponding to the error introduced in the temporal cooperation step. Note that $\boldsymbol{S}_{ij}^{(n)}$ also depends on the amount of resources $y_{ij}$ allocated to the measurements link $(i, j)$ that can be determined by node prioritization discussed in the "Distributed node Activation" section [24].

Centralized node activation can be performed by calculating the agent index $i_n$ that is optimum, in the sense that the localization error reduction of the network is maximized. The optimum index can be obtained as follows:

$$i_n = \max_{i\in\mathcal{N}_a} \operatorname{tr}\left((\boldsymbol{Q}^{(n)})^{-1} + \sum_{j\in\mathcal{A}_i^{(n)}} \boldsymbol{S}_{ij}^{(n)}\right)^{-1}. \tag{27}$$

This node-activation strategy is one-step optimal because the active node is selected such that the localization error at time $n + 1$ is minimized. Alternatively, one can also try to activate nodes so that the average error over multiple time instants is minimized. Such a problem can be solved through dynamic programming, but the computational complexity increases rapidly with the number of time steps. Note that the evaluation of (27) relies on the joint error matrix $\boldsymbol{Q}^{(n)}$. The centralized node-activation strategy is thus not scalable with the network size since it necessitates a central controller that collects the information of all the agents in the network. For this reason, for large-scale NLN, distributed node-activation strategies are needed.

### Distributed node activation

Consider the case in which the activation set may consist of multiple agents. In particular, at time $n$ every agent $i$ tries to make distance measurements with its neighbors $j \in \mathcal{A}_i^{(n)}$ with a certain channel access probability $\zeta_i^{(n)}$. The one-step optimization problem that minimizes the localization error over the channel access probabilities $\zeta_i^{(n)}$ is given by

$$\mathcal{P}_{\mathrm{NA}}^{(n)}: \ \underset{\{\zeta_i^{(n)}\}_{i\in\mathcal{N}_a}}{\text{minimize}} \ \ \mathbb{E}\left\{\operatorname{tr}\{\mathbf{Q}^{(n+1)}\}\middle|\,\mathbf{Q}^{(n)}\right\}$$
$$\text{subject to} \ \ 0 \le \zeta_i^{(n)} \le 1, \ \ i \in \mathcal{N}_a,$$

where the expectation in the objective function is over the randomness in the channel access event for all the agents. In [26], the optimal channel access probabilities $\zeta_i^{(n)}, i \in \mathcal{N}_a$ resulting from $\mathcal{P}_{\mathrm{NA}}^{(n)}$ can be obtained as

$$\zeta_i^{(n)} = \begin{cases} 1, & \text{if } \mathcal{X}_i^{(n)} > \displaystyle\sum_{j\in\mathcal{A}_i^{(n)}\cup\{i\}} \Delta_j^{(n)} \\ 0, & \text{otherwise} \end{cases}, \tag{28}$$

where $\mathcal{X}_i^{(n)}$ denotes the expected error reduction of agent $i$, if it is activated and successful, makes range measurements with its neighbors $\mathcal{A}_i^{(n)}$ and $\Delta_j^{(n)}$, and denotes the error increase of the agents in the subnetwork $\mathcal{A}_i^{(n)} \cup \{i\}$ during the time range measurements are performed. Note that $\mathcal{X}_i^{(n)}$ and $\Delta_j^{(n)}$ are functions of $\boldsymbol{Q}_i^{(n)}$ and $\boldsymbol{Q}_j^{(n)}$, $j \in \mathcal{A}_i \cup \{i\}$, respectively.

#### *Remark 3*

This optimal strategy $\mathcal{P}_{\mathrm{NA}}^{(n)}$ leads to a nonrandom node activation in the sense that an agent accesses the channel either with probability one or with probability zero. Moreover, the optimal strategy is distributed because for agent $i$, $\chi_i^{(n)}$ and $\Delta_j^{(n)}$ can be determined or accurately approximated using information that is either locally available or has been received from neighboring nodes $j \in \mathcal{A}_i^{(n)}$. Unlike the setting in the centralized node activation, the distributed strategy may activate multiple nodes at the same time and cause packet collisions. The possibility of these collision events can be reduced by incorporating channel sensing in the presented activation strategy. This results in the distributed node-activation strategy presented in Algorithm 1 that has been successfully verified on-the-field with the Peregrine system for 3-D NLN.

## Node prioritization

Node-prioritization strategies provide a desirable tradeoff between resource consumption and localization accuracy. In

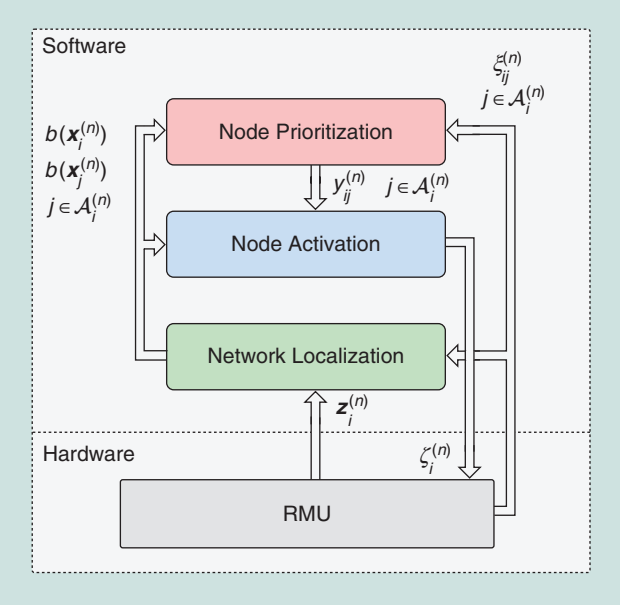


**FIGURE 4.** A flowchart showing the interaction of node activation, node prioritization, network localization, and the RMU.

what follows, we again discuss centralized and distributed strategies for node prioritization.

### Centralized node prioritization

For time $n+1$, the error matrix $\boldsymbol{Q}^{(n+1)}$ can be obtained [24] as

$$\boldsymbol{Q}^{(n+1)} = \left( (\boldsymbol{Q}^{(n)})^{-1} + \sum_{(i,j)\in\mathcal{E}^{(n)}} y_{ij}^{(n)} \xi_{ij}^{(n)} \boldsymbol{u}_{ij}^{(n)} \boldsymbol{u}_{ij}^{(n)\mathrm{T}} \right)^{-1} + \Delta^{(n+1)},$$

where $\mathcal{E}^{(n)} = \{(i,j) \ : \ i \in \mathcal{N}_\mathrm{a}, j \in \mathcal{A}_i^{(n)}, i > j\}$ is the set of candidate measurement link pairs, $y_{ij}^{(n)}$ is the amount of resources allocated to the measurement link pair $(i,j)$, $\xi_{ij}^{(n)}$ represents the channel quality between nodes $i$ and $j$, and $\boldsymbol{u}_{ij}^{(n)}$ is given in [21] and the "Spatiotemporal Fusion Based on the SPA" section and depends on the relative positions of nodes $i$ and $j$. Furthermore, $y_{ij}^{(n)}$ are the variables to be optimized. As a special case, if only node $i$ is activated, $\mathcal{E}^{(n)} = \{(i,j) : j \in \mathcal{A}_i^{(n)}\}$.

Now the following optimization problem for centralized node prioritization can be introduced

$$\mathcal{P}_{\mathrm{NP-C}}^{(n)} \ : \ \underset{\{y_{ij}^{(n)}\}_{(i,j)\in\mathcal{E}^{(n)}}}{\text{minimize}} \ \ \mathrm{tr}(\boldsymbol{Q}^{(n+1)}) \quad \text{subject to} \ \ l_k(\{y_{ij}^{(n)}\}_{(i,j)\in\mathcal{E}^{(n)}}) \leqslant 0, \ k \in \mathcal{L},$$

where $\mathcal{L}$ is the set of linear constraints $l_k(\cdot)$. Due to the special structure of $\boldsymbol{Q}^{(n+1)}$, $\mathcal{P}_{\mathrm{NP-C}}^{(n)}$ can be transformed to the following semidefinite program (SDP):

$$\begin{aligned} \underset{\boldsymbol{M}, \{y_{ij}^{(n)}\}_{(i,j)\in\mathcal{E}^{(n)}}}{\text{minimize}} \quad & \mathrm{tr}(\boldsymbol{M}) \\ \text{subject to} \quad & \begin{bmatrix} \boldsymbol{M} & \boldsymbol{I} \\ \boldsymbol{I} & \boldsymbol{J}^{(n)} \end{bmatrix} \succcurlyeq 0 \\ & \boldsymbol{J}^{(n)} = (\boldsymbol{Q}^{(n)})^{-1} + \sum_{(i,j)\in\mathcal{E}^{(n)}} y_{ij}^{(n)} \xi_{i,j}^{(n)} \boldsymbol{u}_{ij}^{(n)} \boldsymbol{u}_{ij}^{(n)\mathrm{T}} \\ & l_k(\{y_{ij}^{(n)}\}_{(i,j)\in\mathcal{E}^{(n)}}) \leqslant 0, \quad k \in \mathcal{L}, \end{aligned}$$

where $\boldsymbol{M}$ is an auxiliary matrix for the SDP formulation [21]. Convex optimization engines [29] can be used to solve the SDP in the above equation. Note that similarly to the node-activation problem, solving the node prioritization problem $\mathcal{P}_{\mathrm{NP-C}}^{(n)}$ requires obtaining the estimates of $\boldsymbol{Q}^{(n)}$, $y_{ij}^{(n)}$, $\xi_{i,j}^{(n)}$, and $\boldsymbol{U}_{ij}^{(n)}$ for the solution of this SDP. A central controller is needed to collect such information. Moreover, the computational complexity of this SDP largely depends on the dimension of $\boldsymbol{Q}^{(n)}$, which is a $DN_\mathrm{a} \times DN_\mathrm{a}$ matrix. For these reasons, centralized node prioritization does not scale with the size of the network.

**Algorithm 1. Distributed node-activation strategy.**

```
for all i ∈ N_a do
    Agent i listens to the channel;
    if the channel is busy then
        Wait for a certain amount of time;
    else
        Determine the access probability ζ_i^(n) from (28);
        if ζ_i^(n) = 1 then
            Access the channel and perform internode measurements;
        end if
    end if
    Broadcast Δ_i^(n);
end for
```

### Distributed node prioritization

Though the centralized formulation can provide better localization performance, in large networks it incurs in extensive communication overhead and computational complexity. For this reason, fully distributed and thus scalable variants are more amenable in practice.

The error matrix for the position of agent $i$ is the $i$th diagonal $D \times D$ block of $\boldsymbol{Q}^{(n+1)}$, denoted by $[\boldsymbol{Q}^{(n+1)}]_i$. This error matrix depends on the geometry of the network and the accuracies of all internode measurements. Therefore, directly optimizing this error matrix does not lead to distributed implementation. An approximation of $[\boldsymbol{Q}^{(n+1)}]_i$ that involves only local parameters can be introduced as follows:

$$[\boldsymbol{Q}^{(n+1)}]_i \approx \tilde{\boldsymbol{Q}}_i^{(n+1)}, \tag{29}$$

where

$$\tilde{\boldsymbol{Q}}_i^{(n+1)} = \left( ([\boldsymbol{Q}^{(n)}]_i)^{-1} + \sum_{j\in\mathcal{A}_i} y_{ij}^{(n)} \varrho_{ij}^{(n)} \boldsymbol{v}_{ij}^{(n)} [\boldsymbol{v}_{ij}^{(n)}]^\mathrm{T} \right)^{-1} + [\Delta^{(n+1)}]_i.$$

In this expression, $\varrho_{ij}^{(n)}$ is given by

$$\varrho_{ij}^{(n)} = \begin{cases} \xi_{ij}^{(n)}, & \text{if } j \in \mathcal{A}_i^{(n)} \cap \mathcal{N}_\mathrm{b} \\ \dfrac{\xi_{ij}^{(n)}}{1 + y_{ij}^{(n)} \mathrm{tr}\left( \boldsymbol{u}_{ij}^{(n)} [\boldsymbol{u}_{ij}^{(n)}]^\mathrm{T} [\boldsymbol{Q}^{(n)}]_j \right)} & \text{if } j \in \mathcal{A}_i^{(n)} \cap \mathcal{N}_\mathrm{a}, \end{cases}$$

and $\boldsymbol{v}_{ij}^{(n)} \in \mathbb{R}^D$ is a unit vector representing the direction between node $i$ and $j$. Note that $\tilde{\boldsymbol{Q}}_i^{(n+1)}$ involves $\varrho_{ij}^{(n)}$, $\boldsymbol{v}_{ij}^{(n)}$, and $y_{ij}^{(n)}$ for $j \in \mathcal{A}_i^{(n)}$, which are either locally available at agent $i$ or can be received by communicating with neighboring nodes $j \in \mathcal{A}_i^{(n)}$.

Using $\mathrm{tr}(\tilde{\boldsymbol{Q}}_i^{(n+1)})$ as the objective function, a distributed node-prioritization problem is formulated as

$$\mathcal{P}_{i,\mathrm{NP-D}}^{(n)}: \ \underset{\{y_{ij}^{(n)}\}_{j\in\mathcal{N}_\mathrm{a}\cup\mathcal{N}_\mathrm{b}\setminus\{i\}}}{\text{minimize}} \ \ \mathrm{tr}(\tilde{\boldsymbol{Q}}_i^{(n+1)}) \quad \text{subject to} \ \ l_{ik}\left(\{y_{ij}^{(n)}\}_{j\in\mathcal{A}_i}\right) \leq 0, k \in \mathcal{L}_i, \tag{30}$$

where $l_{ik}(\cdot)$ are linear constraints [21]. It can be shown that $\mathcal{P}_{i,\mathrm{NP-D}}^{(n)}$ is a convex problem by performing the same steps as in [21]. Moreover, for a general $D$, one can show that $\mathcal{P}_{i,\mathrm{NP-D}}^{(n)}$ is an SDP. For $D = 2$, $\tilde{\boldsymbol{Q}}_i^{(n+1)}$ is a $2 \times 2$ matrix and $\mathrm{tr}(\tilde{\boldsymbol{Q}}_i^{(n+1)})$ has a simpler explicit expression as a function of $y_{ij}^{(n)}$. As a consequence, $\mathcal{P}_{i,\mathrm{NP-D}}^{(n)}$ can be further transformed into a second-order cone program [22], [29].

So far, we have discussed node prioritization for cooperative IoT networks. In noncooperative scenarios where agents only perform agent-anchor range measurements, the approximation (29) becomes an equality and the error matrix for agent $i$ is

$$[\boldsymbol{Q}^{(n+1)}]_i = \left(([\boldsymbol{Q}^{(n)}]_i)^{-1} + \sum_{j\in\mathcal{N}_\mathrm{b}} y_{ij}^{(n)}\xi_{ij}^{(n)}\boldsymbol{v}_{ij}^{(n)}[\boldsymbol{v}_{ij}^{(n)}]^\mathrm{T}\right)^{-1} + [\Delta^{(n+1)}]_i\,.$$

*Remark 4*

The node-prioritization problem in noncooperative scenarios is a special case of $\mathcal{P}_{i,\mathrm{NP-D}}^{(n)}$ that can be solved even more efficiently by using geometric optimization methods [15]. Furthermore, if the constraint (30) can be expressed as follows

$$\sum_{j\in\mathcal{N}_\mathrm{b}} y_{ij}^{(n)} \leqslant R_\mathrm{tot} \qquad \text{with } y_{ij}^{(n)} \geqslant 0,\ j\in\mathcal{N}_\mathrm{b}\,,$$

the optimal solution is demonstrated to have a sparsity property. Note that here $R_\mathrm{tot}$ is the total amount of available resources. In particular, the optimal set of measurements can be performed with at most $D(D+1)/2$ anchors. This sparsity property provides a theoretical basis for reducing measurement links in localization networks.

## Case study

In this section, we demonstrate the performance benefits of cooperation among devices and multisensor fusion in a large-scale IoT network using simulated measurements. Some of the presented algorithms have also been evaluated in the real-time localization system called *Peregrine* [16]. (A video that demonstrates how this system operates and the performance advantages related to the proposed algorithms is available online at http://winslab.lids.mit.edu/nln-technology-readiness.mp4.)

### Scenario

An IoT network that consists of 512 mobile agents and 27 anchors is considered. The anchors form an equally spaced 3-D grid, where possible coordinate values on each axis in 3-D space are $\{-60, 0, 60\}$ m. Mobile agents are equipped with an inertial measurement unit (IMU) and an RMU, and they infer navigation information every $\Delta T = 0.05$ s. This scenario is inspired by a swarm of micro unmanned aerial vehicles (UAVs) that operate in a large building such as a stadium or warehouse.

The state $\mathbf{x}_i^{(n)}$, of agent $i\in\mathcal{N}_\mathrm{a}$ consists of its position $\mathbf{p}_i^{(n)} = [p_{i,1}^{(n)}\ p_{i,2}^{(n)}\ p_{i,3}^{(n)}]^\mathrm{T} \in \mathbb{R}^3$, velocity $\dot{\mathbf{p}}_i^{(n)} \in \mathbb{R}^3$, and its orientation represented by an unit quaternion $\mathbf{q}_i^{(n)} \in \mathbb{R}^4$. The initial states $\mathbf{x}_i^{(1)}, i\in\mathcal{N}_\mathrm{a}$ are chosen as follows. The initial positions $\mathbf{p}_i^{(1)}$ are sampled from the PDF that is uniform on the 3-D cube $\mathcal{R} = [-60,60]\,\mathrm{m}\times[-60,60]\,\mathrm{m}\times[-60,60]\,\mathrm{m}$; the initial velocity is set to $\dot{\mathbf{p}}_i^{(1)} = \mathbf{0}$ m/s, and the initial quaternion is set to $\mathbf{q}_i^{(1)} = [1\ 0\ 0\ 0]^\mathrm{T}$. The trajectories of the agents are generated randomly. The parts of the trajectories that are related to the substates $\mathbf{s}_i^{(n)} := [[\mathbf{p}_i^{(n)}]^\mathrm{T}\ [\dot{\mathbf{p}}_i^{(n)}]^\mathrm{T}]^\mathrm{T}$ are generated by means of a constant velocity motion model [17]. More specifically, at time $n$ the new substate $\mathbf{s}_i^{(n)}$ of agent $i\in\mathcal{N}_\mathrm{a}$ is obtained from $\mathbf{s}_i^{(n-1)}$ as

$$\mathbf{s}_i^{(n)} = \boldsymbol{A}\mathbf{s}_i^{(n-1)} + \boldsymbol{C}\mathbf{g}_i^{(n)},$$

where matrices $\boldsymbol{A}$ and $\boldsymbol{C}$ are given as in [17] and $\mathbf{g}_i^{(n)} \in \mathbb{R}^3$ is the acceleration vector in the global reference frame.

Vector $\mathbf{g}_i^{(n)}$ consists of the random driving noise $\mathbf{r}_i^{(n)}$ and the drag force $\mathbf{f}_i^{(n)}$, i.e., $\mathbf{g}_i^{(n)} = \mathbf{r}_i^{(n)} + \mathbf{f}_i^{(n)}$. In particular, $\mathbf{r}_i^{(n)}$ is a zero-mean Gaussian random vector, i.e., $\mathbf{r}_i^{(n)} \sim \mathcal{N}(\mathbf{0}, \sigma_\mathbf{r}^2\boldsymbol{I}_3)$ and the drag force is given by $\mathbf{f}_i^{(n)} = [f_{1,i}^{(n)}\ f_{2,i}^{(n)}\ f_{3,i}^{(n)}]^\mathrm{T}$ with elements $f_{k,i}^{(n)\mathrm{T}} = -\gamma_f \dot{\mathrm{p}}_{k,i}^{(n-1)}\left|\dot{\mathrm{p}}_{k,i}^{(n-1)}\right|$, $k\in\{1,2,3\}$. The drag force is introduced to limit the speed of the agents. The following parameters are used: $\sigma_\mathbf{r} = 4.0\ \mathrm{m/s^2}$ and $\gamma_f = 0.2\ \mathrm{m^{-1}}$. These values result in trajectories with speeds and maneuverability that are reasonable for micro UAVs. In particular, the maximum speed of each agent typically remains below 5.0 m/s. The agent orientation $\mathbf{q}_i^{(n)}$ evolves as follows: At each time step $n$, agent $i$ rotates with random turn rate $\boldsymbol{\omega}_i^{(n)} \sim \mathcal{N}(\mathbf{0}, \sigma_{\boldsymbol{\omega}}^2\boldsymbol{I}_3)$, where $\sigma_\omega = 0.5\ \mathrm{s^{-1}}$. Note that $\boldsymbol{\omega}_i^{(n)}$ is the turn rate in the local reference frame of agent $i$. The corresponding state evolution model is provided in [30].

As in most inertial navigation techniques for multisensor fusion, in the simulated algorithm, the measurements provided by the IMU are incorporated as deterministic control input $\boldsymbol{u}_i^{(n)} = [\boldsymbol{u}_{i,\boldsymbol{\varphi}}^{(n)\mathrm{T}}\ \boldsymbol{u}_{i,\boldsymbol{\omega}}^{(n)\mathrm{T}}]^\mathrm{T}$. In particular, the IMU measurement $\boldsymbol{u}_i^{(n)}$ consists of an acceleration measurement $\boldsymbol{u}_{i,\boldsymbol{\varphi}}^{(n)}$ and a turn-rate measurement $\boldsymbol{u}_{i,\boldsymbol{\omega}}^{(n)}$, which are realizations of the RVs

$$\begin{aligned}\mathbf{u}_{i,\boldsymbol{\varphi}}^{(n)} &= \boldsymbol{\varphi}_i^{(n)} + \mathbf{c}_{i,\boldsymbol{\varphi}}^{(n)}\\ \mathbf{u}_{i,\boldsymbol{\omega}}^{(n)} &= \boldsymbol{\omega}_i^{(n)} + \mathbf{c}_{i,\boldsymbol{\omega}}^{(n)}\,,\end{aligned}$$

where $\boldsymbol{\varphi}_i^{(n)}$ is the true acceleration of agent $i$ in its local reference frame. The IMU noise $\mathbf{c}_i^{(n)} = [\mathbf{c}_{i,\boldsymbol{\varphi}}^{(n)\mathrm{T}}\ \mathbf{c}_{i,\boldsymbol{\omega}}^{(n)\mathrm{T}}]^\mathrm{T}$ consists of acceleration $\mathbf{c}_{i,\boldsymbol{\varphi}}^{(n)} \sim \mathcal{N}(\mathbf{0}, \sigma_{\boldsymbol{\varphi}}^2\boldsymbol{I}_3)$ and turn rate $\mathbf{c}_{i,\boldsymbol{\omega}}^{(n)} \sim \mathcal{N}(\mathbf{0}, \sigma_{\boldsymbol{\omega}}^2\boldsymbol{I}_3)$ components. The functional form of the resulting state-evolution model $\mathbf{x}_i^{(n)} = a_i(\mathbf{x}_i^{(n-1)}, \mathbf{c}_i^{(n)}; \boldsymbol{u}_i^{(n)})$ is provided in [30].

The range measurement $\mathrm{z}_{ij}^{(n)}$ made by agent $i\in\mathcal{N}_\mathrm{a}$ with node $j$ at time step $n$ is modeled as

$$\mathrm{z}_{ij}^{(n)} = \left\|\mathbf{p}_i^{(n)} - \mathbf{p}_j^{(n)}\right\| + \mathrm{v}_{ij}^{(n)},$$

where $\mathrm{v}_{ij}^{(n)} \sim \mathcal{N}(0, \sigma_\mathrm{v}^2)$ is the Gaussian noise with standard deviation $\sigma_\mathrm{v} = 0.1$ m. A more detailed, technology-specific model for ranging from wideband radio-frequency signals can be found in [27] and [28].

It is assumed that the number of available channels for performing range measurements is limited to 16, which means that only a subset of 16 agents can perform range measurements at a specific time step $n$. For this reason, time-division multiple access (TDMA) is performed by partitioning 512 agents into 32 disjoint groups, with each group consisting of 16 agents. At each time step $n$, only the agents in one of the 32 groups can make range measurements while all the others remain idle. At each time step $n$, for those 16 agents that perform range measurements, the set $\mathcal{A}_i^{(n)}$ is given as follows: range measurements can only be performed with nodes that are located within a communication range of 52 m. Moreover, if there are more than $M$ potential-neighbor nodes, $M$

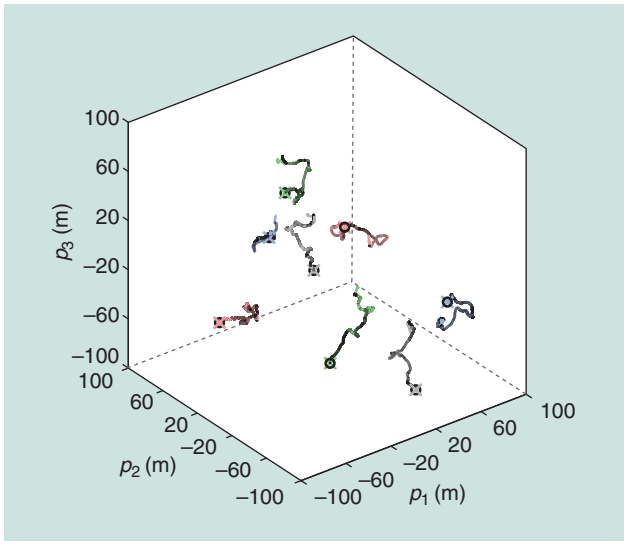


**FIGURE 5.** The trajectories related to eight exemplary agents and one simulation run. Colored and black curves represent the estimated and true trajectories, respectively. Similarly, colored crosses and black circles represent the estimated and true positions at the last time step, respectively.

of them are randomly selected. This selection of at most $M$ neighbor nodes limits the energy consumption. It also reduces the number of loops in the factor graph in Figure 3 and thus the related negative effects, e.g., overconfident beliefs. Note that the communication range of 52 m was chosen so that for agents inside the region $\mathcal{R}$, there is at least one, and at most, four neighbor anchors.

In our simulation, the SPBP algorithm [5] is used, which is based on the design framework presented in the "Distributed Network-Localization Algorithms" section. Note that, in the considered scenario with 512 UAVs, localization algorithms based on SBE are unfeasible because they are not scalable in the number of agents [6, Sec. VII-C]. To the best of our knowledge, SPBP is the only available algorithm for cooperative location and orientation estimation in 3-D. One hundred simulation runs were performed and 1,200 time steps were simulated. Examples of true and estimated trajectories are shown in Figure 5. As a metric for localization performance the 3-D localization error outage (LEO) was used. The outage is a well-established concept in wireless communications; in the context of NLN, the LEO is similarly defined as the empirical probability that the localization error is above the predefined threshold $e_{\text{th}}$.

### Network localization results

To study the impact of cooperation among agents as well as the impact related to multisensor fusion, the following configurations are compared: In the "Baseline" configuration, an agent makes range measurements only with the anchors within its communication range and does not perform IMU measurements. In the "Spatial Cooperation" configuration, additional range measurements are performed by cooperation among agents. In the "IMU Fusion" configuration, IMU measurements are performed but agents do not cooperate. Finally, in the "Spatial Cooperation + IMU Fusion" configuration, cooperation among agents as well as IMU measurements are performed. Note that for the network localization results presented in this section the following parameters were used: $M = 6, \sigma_{\boldsymbol{\varphi}} = 10^{-4}\ \text{m/s}^2$, and $\sigma_{\boldsymbol{\omega}} = 5 \times 10^{-3}\,\text{s}^{-1}$.

Figure 6 shows the LEOs (obtained by averaging more than 100 simulation runs, 512 agents, and 900 time steps) versus threshold $e_{\text{th}}$ for the four simulated configurations. Since SPBP needs a certain number of time steps for initialization, for the network localization results, the first 300 time steps were not incorporated in the LEOs evaluation. The following key observations can be obtained from these results:

1) A very desirable localization performance can be obtained with "Spatial Cooperation + IMU Fusion." Specifically, the threshold $e_{\text{th}}$ is 0.11 m and 0.17 m at a LEO of $10^{-1}$ and $10^{-2}$, respectively. Remarkably, for $e_{\text{th}} \geq 0.3$ m the LEO is 0.
2) The localization error is significantly reduced by spatial cooperation. In particular, by comparing "IMU Fusion" with "Spatial Cooperation + IMU Fusion," it can be seen that the $e_{\text{th}}$ is reduced from 0.54 m to 0.11 m (by 79.6%) at a LEO of $10^{-1}$ and from 4.21 m to 0.17 m (by 96.0%) at a LEO of $10^{-2}$. The reason for the performance gain of "Spatial Cooperation + IMU Fusion" with respect to "IMU Fusion" is that in the former configuration the agents have more neighbor nodes available for localization.
3) Incorporating IMU measurements also significantly reduces the localization error. More specifically, by comparing "Spatial Cooperation" with "Spatial Cooperation + IMU Fusion" it can be seen that the $e_{\text{th}}$ is reduced from 2.92 m to 0.11 m (by 96.2%) at a LEO of $10^{-1}$ and from 5.00 m to 0.17 m (by 96.0%) at a LEO of $10^{-2}$. The performance benefits "Spatial Cooperation + IMU Fusion" can be explained by the fact that the agents only makes range measurements every 32 time steps. Using "Spatial Cooperation" the localization error accumulates rapidly during the time period when no range measurements are performed. In contrast, by incorporating IMU measurements as in "Spatial Cooperation + IMU Fusion," this localization error accumulation can be significantly reduced.
4) Due to the few neighbor nodes available for localization and the high mobility of the agents, the localization performance of "Baseline" is very poor.

### Network operation results

To demonstrate the benefits of network operation algorithms, a heterogeneous network that consists of two UAV classes was simulated. There were 256 UAVs in each class. The first class performed IMU measurements with noise standard deviations $\sigma_{\boldsymbol{\varphi}} = 3.3 \times 10^{-5}\ \text{m/s}^2$ and $\sigma_{\boldsymbol{\omega}} = 1.7 \times 10^{-3}\,\text{s}^{-1}$; the second class performed IMU measurements with noise standard deviations $\sigma_{\boldsymbol{\varphi}} = 3 \times 10^{-4}\ \text{m/s}^2$, and $\sigma_{\boldsymbol{\omega}} = 1.5 \times 10^{-2}\,\text{s}^{-1}$. All other parameters are as described in the "Scenario" section and were identical for both classes. For "Node Activation," spatial cooperation and IMU fusion was simulated together

with the distributed network activation algorithm described in the "Distributed Node Activation" section to control the RMU measurements. At each time step $n$, every UAV determined its channel access probability $\zeta_i^{(n)}$ according to (28) and if $\zeta_i^{(n)} = 1$, it tried to access the channel. In a certain step, if multiple UAVs of the same group (see the "Case Study" section) that were also in the same subnetwork tried to access the channel, only one randomly selected UAV was able to perform an RMU measurement. As a reference method, "TDMA" was simulated where, as in the previous "Network Localization Results" section, spatial cooperation and IMU fusion with TDMA for channel access was performed. For both "Node Activation" and "TDMA," $M = 4$ and $M = 6$ were considered.

In the simulated scenario, "Node Activation" had a number of communication links related to RMU measurements that, compared to "TDMA," was reduced by 14.2% and 19.9% for $M = 4$ and $M = 6$, respectively. The average number of measurements performed per agent and per time step was 0.13 $(M = 4)$ and 0.19 $(M = 6)$ for "TDMA" and 0.11 $(M = 4)$ and 0.15 $(M = 6)$ for "Node Activation." Furthermore, consider a UWB radio that consumes $1.7 \times 10^{-4}$ J per range measurement was used as an RMU [16], "Node Activation" can reduce the overall energy consumption of the network for all 1,200 time steps by 2.1 J for $M = 4$ and by 4.2 J for $M = 6$.

Figure 7 shows the LEOs—obtained by averaging more than 100 simulation runs, 512 agents, and 1,200 time steps—versus threshold $e_{\text{th}}$ for the four simulated configurations. Note that the "Spatial Cooperation + IMU Fusion" results in Figure 6 correspond to the identical scenario as the "TDMA," $M = 6$ results in Figure 7. However, contrary to Figure 6, in Figure 7 all 1,200 time steps are considered. The following two observations can be made:

1) "Node Activation" can significantly increase localization accuracy. In particular, at a LEO of $10^{-2}$, $e_{\text{th}}$ is reduced from 7.18 m to 2.29 m, i.e., by 68.1% for $M = 4$ and from 6.42 m to 0.85 m, i.e., by 86.8% for $M = 6$. This is because with "Node Activation" UAVs in the second class tend to perform more range measurements compared to the ones in the first class, so they can compensate for their larger IMU noise standard deviation. In this way, "Node Activation" can also reduce the overall localization error of the network compared to "TDMA," where the UAVs in both classes make the same number of range measurements

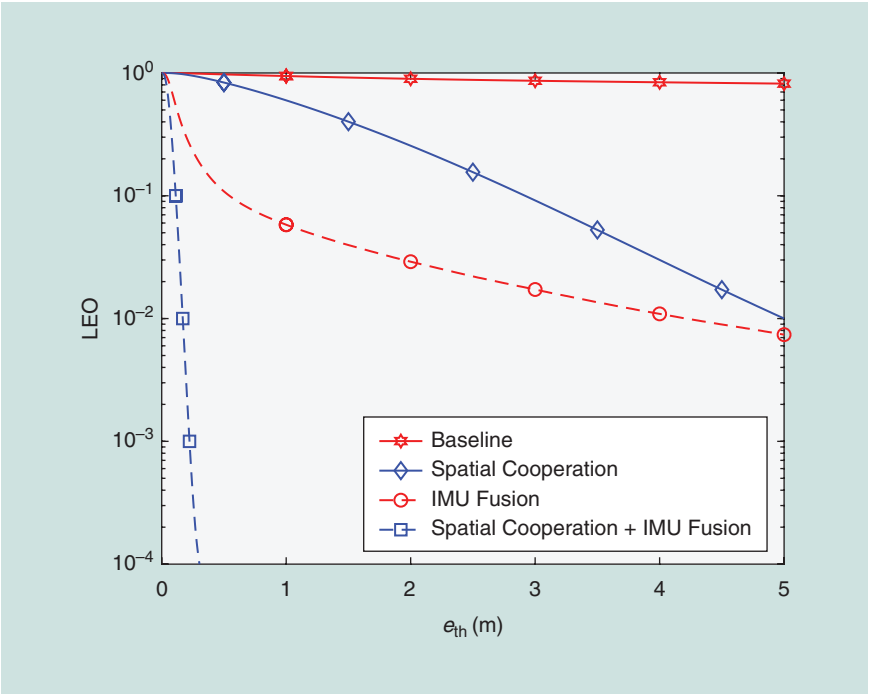


**FIGURE 6.** The LEO versus threshold $e_{\text{th}}$ for the different simulated network localization configurations.

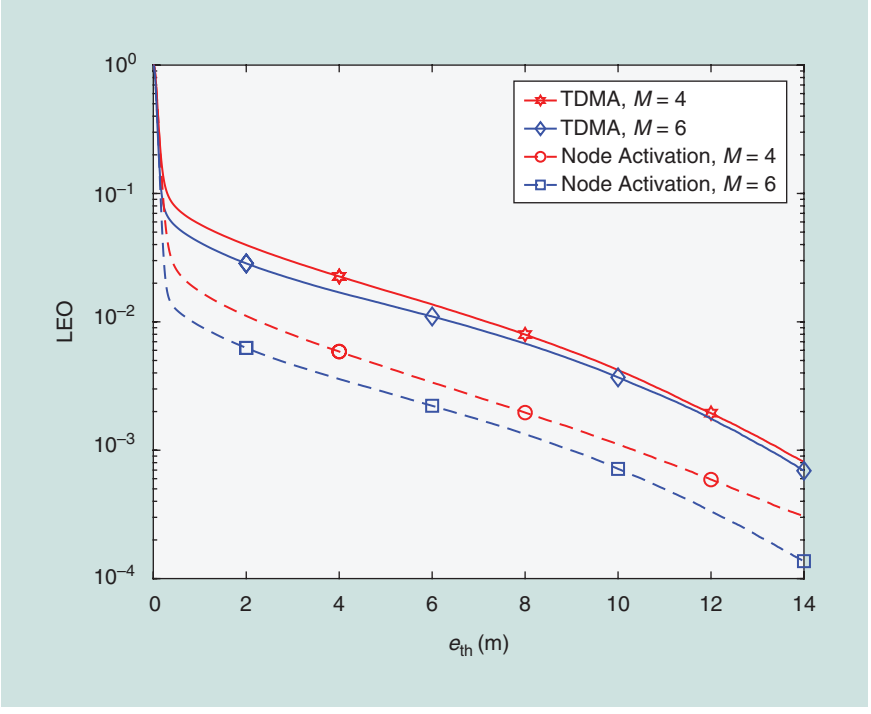


**FIGURE 7.** The LEO versus threshold $e_{\text{th}}$ for the different channel access strategies and different $M$.

on average. Note that the improvement in localization performance is most significant at the first time steps during the initialization phase of the algorithm.

2) Incrementing $M$ from four to six results is a localization error reduction that is small compared to the reduction related to performing "Node Activation" instead of "TDMA."

In particular, "Node Activation" for $M = 4$ performs significantly better than "TDMA" for $M = 6$. Thus it can be noted that a smart activation of agents can compensate for a low number of neighboring nodes.

## Final remarks

The size and heterogeneity of IoT networks calls for a new class of scalable and technology-agnostic localization algorithms. In this article, we presented NLN, a paradigm that introduces scalable and distributed techniques for multisensor fusion in the IoT. NLN can provide technology-agnostic algorithms for IoT networks that exploit spatiotemporal cooperation to reduce the amount of required infrastructure. It also leads to the development of intelligent network operation strategies that allocate localization resources (e.g., transmission power and channel access opportunity) to extend the energy consumption of devices and to increase the localization accuracy. Localization performance and saving in terms of communication costs and energy consumption have been demonstrated in a case study with 500 mobile agents that aim to infer their location and their orientation in 3-D space. In particular, node activation significantly reduced energy consumption and, at the same time, increases the localization performance of the network. These results confirmed that in IoT applications localization and navigation performance can be strongly increased by multisensor fusion and cooperation among devices.

**The size and heterogeneity of IoT networks call for a new class of scalable and technology-agnostic localization algorithms.**


## Acknowledgments

This research was supported, in part, by the Office of Naval Research under Grants N00014-16-1-2141 and N62909-18-1-2017, Defense University Research Instrumentation Program under Grant N00014-17-1-2379, the Austrian Science Fund under Grant J3886-N31, the Hong Kong Innovation and Technology Commission under Grant ITS/066/17FP, and the European Union's H2020 research and innovation programme Marie Skłodowska-Curie under Grant 703893. We also thank B. Teague and T. Wang for stimulating discussions.


## Authors


***Moe Z. Win*** (win@mit.edu) is a professor at the Massachusetts Institute of Technology (MIT), and the founding director of the Wireless Information and Network Sciences Laboratory. Prior to joining MIT, he was with AT&T Research Laboratories and NASA Jet Propulsion Laboratory. His current research interests include network localization and navigation, network-interference exploitation, and quantum-information science. He has been an IEEE Distinguished Lecturer and, currently, is serving on the SIAM Diversity Advisory Committee. He is an elected fellow of the American Association for the Advancement of Science and the Institution of Engineering and Technology. He was honored with two IEEE Technical Field Awards: the IEEE Kiyo Tomiyasu Award and the IEEE Eric E. Sumner Award. Other recognitions include the IEEE Communications Society Edwin H. Armstrong Achievement Award, the International Prize for Communications Cristoforo Colombo, and the Copernicus Fellowship and the Laurea Honoris Causa from the Università degli Studi di Ferrara. He is an ISI highly cited researcher. He is a Fellow of the IEEE.

***Florian Meyer*** (fmeyer@mit.edu) received his Dipl.-Ing. and Ph.D. degrees in electrical engineering from Technische Universität Wien, in 2011 and 2015, respectively. He is currently a postdoctoral associate at the Wireless Information and Network Sciences Laboratory, Massachusetts Institute of Technology. In 2016, he was a research scientist with the NATO Centre for Maritime Research and Experimentation. His research interests include inference on graphs, cooperative navigation, multiobject tracking, and information-seeking control. He served as a technical program committee member of several IEEE conferences and was cochair of the IEEE Workshop on Advances in Network Localization and Navigation at the IEEE International Conference on Communications in 2018. He is an Erwin Schrödinger Fellow and a Member of the IEEE.

***Zhenyu Liu*** (zhenyu.liu.us@ieee.org) received his B.Sc. and M.Sc. degrees in electronic engineering from Tsinghua University, in 2011 and 2014, respectively. He received academic excellence scholarships from Tsinghua University from 2008 to 2010. In 2014, he joined the Wireless Information and Network Sciences Laboratory at the Massachusetts Institute of Technology, where he is pursuing his Ph.D. degree in aeronautics and astronautics. His research interests include statistical inference and stochastic optimization, with application to communication and localization networks. He received the Best Paper Award at the IEEE Latin-American Conference on Communications in 2017. He has translated his research on network localization and navigation into practical demonstration systems for which he received the first prize at the IEEE Communications Society Student Competition in 2016.

***Wenhan Dai*** (whdai@mit.edu) received his B.S. degrees in electronic engineering and mathematics from Tsinghua University, in 2011 and his M.S. degree in aeronautics and astronautics at the Massachusetts Institute of Technology, in 2014, where he is pursuing his Ph.D. degree in aeronautics and astronautics with the Wireless Information and Network Sciences Laboratory. His research interests include communication theory and stochastic optimization with application to wireless communication and network localization. He was honored by the Marconi Society with the Paul Baran

Young Scholar Award in 2017. He received the Marconi-Bioinformatic, Intelligent Systems and Educational Technology Best Paper Award from the IEEE International Conference on Ubiquitous Wireless Broadband in 2017, the Chinese Government Award for Outstanding Student Abroad in 2016, and first prize at the IEEE Communications Society Student Competition in 2016. He was recognized as an exemplary reviewer of *IEEE Communications Letters* in 2014. He is a Student Member of the IEEE.

***Stefania Bartoletti*** (stefania.bartoletti@unife.it) received her laurea degree (summa cum laude) in electronics and telecommunications engineering and her Ph.D. degree in information engineering from the University of Ferrara, in 2011 and 2015, respectively. She is currently a Marie Skłodowska-Curie Global Fellow at the University of Ferrara, within the H2020 European Framework for a research project with the Massachusetts Institute of Technology. Her research interests include the theory and experimentation of wireless networks for passive localization and physical behavior analysis. She served as chair of the Technical Program Committee of the IEEE Workshop on Advances in Network Localization and Navigation at the IEEE International Conference on Communications in 2017 and 2018, respectively. She is a recipient of the 2016 Paul Baran Young Scholar Award of the Marconi Society. She is a Member of the IEEE.

***Andrea Conti*** (a.conti@ieee.org) received his Ph.D. degree in electronic engineering and computer science from the University of Bologna, in 2001. He is currently an associate professor with the University of Ferrara. In the summer of 2001, he was with the Wireless Systems Research Department, AT&T Research Laboratories. He is a frequent visitor of the Wireless Information and Network Sciences Laboratory at the Massachusetts Institute of Technology, where he holds the research affiliate appointment. His research interests include theory and experimentation of wireless systems and networks, including network localization and distributed sensing. He is a recipient of the HTE Puskás Tivadar Medal and of the IEEE Communications Society's Stephen O. Rice Prize in the field of Communications Theory. He is a cofounder and elected secretary of the IEEE Quantum Communications and Information Technology Emerging Technical Subcommittee. He is an elected fellow of the Institution of Engineering and Technology and has been selected as an IEEE Distinguished Lecturer.

## References


[1] M. Z. Win, A. Conti, S. Mazuelas, Y. Shen, W. M. Gifford, D. Dardari, and M. Chiani, "Network localization and navigation via cooperation," *IEEE Commun. Mag.*, vol. 49, no. 5, pp. 56–62, May 2011.

[2] A. T. Ihler, J. W. Fisher, III, R. L. Moses, and A. S. Willsky, "Nonparametric belief propagation for self-localization of sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 4, pp. 809–819, Apr. 2005.

[3] Y. Shen and M. Z. Win, "Fundamental limits of wideband localization—Part I: A general framework," *IEEE Trans. Inf. Theory*, vol. 56, no. 10, pp. 4956–4980, Oct. 2010.

[4] A. Conti, M. Guerra, D. Dardari, N. Decarli, and M. Z. Win, "Network experimentation for cooperative localization," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 2, pp. 467–475, Feb. 2012.

[5] F. Meyer, O. Hlinka, and F. Hlawatsch, "Sigma point belief propagation," *IEEE Signal Process. Lett.*, vol. 21, no. 2, pp. 145–149, Feb. 2014.

[6] F. Meyer, O. Hlinka, H. Wymeersch, E. Riegler, and F. Hlawatsch, "Distributed localization and tracking of mobile networks including noncooperative objects," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 1, pp. 57–71, Mar. 2016.

[7] F. Zabini and A. Conti, "Inhomogeneous Poisson sampling of finite-energy signals with uncertainties in $\mathbb{R}^d$," *IEEE Trans. Signal Process*, vol. 64, no. 18, pp. 4679–4694, Sept. 2016.

[8] A. Sandryhaila and J. M. F. Moura, "Big data analysis with signal processing on graphs: Representation and processing of massive data sets with irregular structure," *IEEE Signal Process. Mag.*, vol. 31, no. 5, pp. 80–90, Sept. 2014.

[9] D. Dardari, A. Conti, C. Buratti, and R. Verdone, "Mathematical evaluation of environmental monitoring estimation error through energy-efficient wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 7, pp. 790–802, July 2007.

[10] R. Karlsson and F. Gustafsson, "The future of automotive localization algorithms: Available, reliable, and scalable localization: Anywhere and anytime," *IEEE Signal Process. Mag.*, vol. 34, no. 2, pp. 60–69, Mar. 2017.

[11] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.

[12] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001.

[13] J. H. Kotecha and P. M. Djuric, "Gaussian sum particle filtering," *IEEE Trans. Signal Process.*, vol. 51, no. 10, pp. 2602–2612, Oct. 2003.

[14] Z. Liu, W. Dai, and M. Z. Win, "Mercury: An infrastructure-free system for network localization and navigation," *IEEE Trans. Mobile Comput.*, vol. 17, no. 5, pp. 1119–1133, May 2018.

[15] W. Dai, Y. Shen, and M. Z. Win, "A computational geometry framework for efficient network localization," *IEEE Trans. Inf. Theory*, vol. 64, no. 2, pp. 1317–1339, Feb. 2018.

[16] B. Teague, Z. Liu, F. Meyer, and M. Z. Win, "Peregrine: 3-D network localization and navigation," in *Proc. IEEE Latin-American Conf. Communications (LATINCOM)*, Nov. 2017.

[17] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. Hoboken, NJ: Wiley, 2004.

[18] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[19] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear Estimation*. Englewood Cliffs, NJ: Prentice Hall, 2000.

[20] F. Daum and J. Huang, "Curse of dimensionality and particle filters," in *Proc. IEEE Aerospace Conf. (AeroConf)*, 2003, pp. 1979–1993.

[21] W. Dai, Y. Shen, and M. Z. Win, "Distributed power allocation for cooperative wireless network localization," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 1, pp. 28–40, Jan. 2015.

[22] W. Dai, Y. Shen, and M. Z. Win, "Energy-efficient network navigation algorithms," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 7, pp. 1418–1430, July 2015.

[23] H. Godrich, A. P. Petropulu, and H. V. Poor, "Power allocation strategies for target localization in distributed multiple-radar architectures," *IEEE Trans. Signal Process.*, vol. 59, no. 7, pp. 3226–3240, July 2011.

[24] T. Wang, Y. Shen, A. Conti, and M. Z. Win, "Network navigation with scheduling: Error evolution," *IEEE Trans. Inf. Theory*, vol. 63, no. 11, pp. 7509–7534, Nov. 2017.

[25] X. Shen and P. K. Varshney, "Sensor selection based on generalized information gain for target tracking in large sensor networks," *IEEE Trans. Signal Process.*, vol. 62, no. 2, pp. 363–375, Jan. 2014.

[26] T. Wang, B. Teague, and M. Z. Win, "Distributed situation-aware scheduling algorithm for network navigation," in *Proc. IEEE Int. Conf. Ubiquitous Wireless Broadband*, Sept. 2017.

[27] D. Dardari, A. Conti, U. J. Ferner, A. Giorgetti, and M. Z. Win, "Ranging with ultrawide bandwidth signals in multipath environments," *Proc. IEEE*, vol. 97, no. 2, pp. 404–426, Feb. 2009.

[28] S. Bartoletti, W. Dai, A. Conti, and M. Z. Win, "A mathematical model for wideband ranging," *IEEE J. Sel. Topics Signal Process*, vol. 9, no. 2, pp. 216–228, Mar. 2015.

[29] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[30] J. D. Hol, F. Dijkstra, H. Luinge, and T. B. Schön, "Tightly coupled UWB/IMU pose estimation," in *Proc. IEEE Int. Conf. Ultra-Wideband (ICUWB)*, 2009, pp. 688–692.


SP