Efficient Multi-Sensor Localization for the Internet-of-Things

Moe Z. Win, *Fellow, IEEE*, Florian Meyer, *Member, IEEE*, Zhenyu Liu, *Student Member, IEEE*, Wenhan Dai, *Student Member, IEEE*, Stefania Bartoletti, *Member, IEEE*, and Andrea Conti, *Senior Member, IEEE*

***Abstract*—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 as well as 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 multi-sensor fusion algorithms. This paper presents a framework to design network localization and navigation (NLN) for IoT. Multi-sensor 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 five hundreds agents. In particular, it is demonstrated that due to multi-sensor fusion and cooperation, the presented network localization and operation algorithms can provide attractive localization performance and reduce communication overhead and energy consumption.**

***Index Terms*—Internet-of-Things, multi-sensor fusion, network localization and navigation, Bayesian estimation.**

I. Introduction

LOCATION-AWARENESS [1]–[6] is a keystone of the Internet-of-Things (IoT) that fosters a wide range of emerging applications such as crowdsensing [7], big data analysis [8], environmental monitoring [9], and autonomous driving [10]. In particular, the position information of IoT devices can contribute to connect and exchange data more efficiently, preserve communication security, and allow autonomous motion. The increasing number and different types of IoT devices generates scenarios where heterogeneous data are collected via different sensing technologies in a distributed way. Compared to conventional wireless localization networks, typically consisting of a limited number of homogeneous nodes, the scale and heterogeneity of an IoT network impose new challenges that need to be addressed. Specifically, IoT localization and navigation calls for a new class of algorithms tailored to IoT networks.

Date of publication: Nov. 2018.

This research was supported, in part, by the Office of Naval Research under Grants N00014-16-1-2141 and N62909-18-1-2017, the Austrian Science Fund (FWF) under grant J3886-N31, the Hong Kong Innovation and Technology Commission under grant ITS/066/17FP, and the Innovation Programme Marie Skłodowska-Curie under Grant 703893.

M. Z. Win, F. Meyer, Z. Liu, and W. Dai are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139, USA (e-mail: {moewin, fmeyer, zliu14, whdai}@mit.edu). S. Bartoletti and A. Conti are with the Department of Engineering and CNIT, University of Ferrara, 44122 Ferrara, Italy (e-mail: {stefania.bartoletti, andrea.conti}@unife.it).

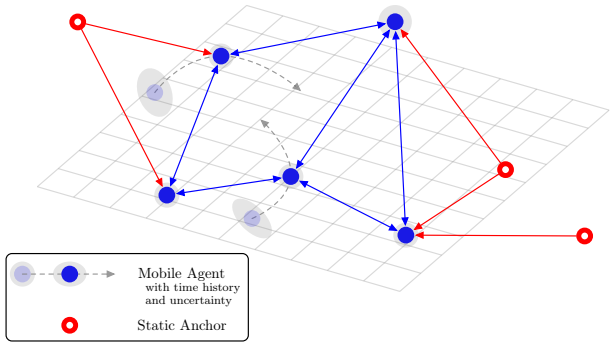
Digital Object Identifier 10.1109/MSP.2018.xxxxxxx

In IoT networks, the sensing capabilities of the devices can vary significantly, providing different kind of measurements carrying positional information such as range, angle-of-arrival, channel state information, or inertial. In addition, 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 multi-sensor fusion and device cooperation. However, state-of-the-art multi-sensor 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. In addition, the high number of devices necessitates network operation strategies to provide inter-device cooperation for an efficient use of the limited battery power and spectral resources. For these reasons, the major difficulties for efficient multi-sensor 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 the 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.

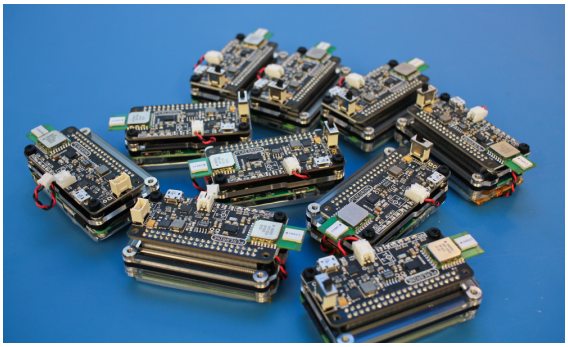
The recently introduced paradigm of network localization and navigation (NLN) [1] has important characteristics that are favorable for multi-sensor localization and navigation in IoT networks. In particular, it can provide technology-agnostic and low-complexity algorithms for heterogeneous multi-sensor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

This paper provides an overview of 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 multi-sensor fusion and cooperation among devices can dramatically increase localization performance in a large-scale scenario with hundreds of



(a)



(b)

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

mobile agents; and

- we quantify how network operation algorithms can reduce the communication overhead and energy consumption of localization networks.

The remainder of the paper is organized as in the following. Section II discusses localization and navigation algorithms for single-node scenarios. Section III describes network localization algorithms for IoT scenarios. Section IV presents efficient operation for location-aware networks. Section V provides localization results for an IoT case study. Finally, Section VI summarizes the main insights offered by the paper.

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, a RV and its realization are denoted by 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 following 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 $\boldsymbol{x}_l$, $l \in \mathcal{L}$ in an arbitrary but known order into a column vector. Finally, the notations of important quantities that are used throughout the paper are summarized in Table I.

II. 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 $\mathcal{N}_\mathrm{a} = \{1\}$) and of N_b anchors at known positions (with index set $\mathcal{N}_\mathrm{b} = \{2, 3, ..., N_\mathrm{b}+1\}$). The agent 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 $\boldsymbol{z}_1^{(n)}$, which is the concatenation of all inter-node measurements $\boldsymbol{z}_{1j}^{(n)}$ with anchors $j \in \mathcal{N}_\mathrm{b}$. 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 $\boldsymbol{z}_1^{(1:n)} \triangleq [\boldsymbol{z}_1^{(1)\mathrm{T}}, \boldsymbol{z}_1^{(2)\mathrm{T}}, \ldots, \boldsymbol{z}_1^{(n)\mathrm{T}}]^\mathrm{T}$).

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

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

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 $f\big(\boldsymbol{x}_1^{(n)}\big|\boldsymbol{x}_1^{(n-1)}; \boldsymbol{u}_1^{(n)}\big)$. 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 prior PDF $f\big(\boldsymbol{x}_1^{(0)}\big)$ at time $n = 0$ is known. The prior information for all times $1, 2, \ldots, n$, i.e., all available information before any measurement is performed, can now be expressed as

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

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

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

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 $f\big(\boldsymbol{z}_1^{(n)}\big|\boldsymbol{x}_1^{(n)}\big)$. 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

Notation	Definition	Notation	Definition
$\mathcal{N}_\mathrm{a}$	Index set of mobile agents	$\mathcal{N}_\mathrm{b}$	Index set of anchors
$\mathbf{x}_i^{(n)}$	Positional state of the ith node at time n	$\mathbf{p}_i^{(n)}$	Position of the ith node at time n
$\mathbf{z}_{ij}^{(n)}$	Inter-node measurement between ith agent and jth node at time n	$\mathbf{z}_i^{(n)}$	All inter-node measurements of the ith agent at time n
$\mathbf{x}_i^{(0:n)}$	All the positional states of the ith node up to time n	$\mathbf{z}_i^{(1:n)}$	All the measurements of the ith agent up to time n
$\alpha_f(\boldsymbol{x}^{(n)})$	Message passed from variable node "$\boldsymbol{x}$" to factor node "f"	$\beta_f(\boldsymbol{x}^{(n)})$	Message passed from factor node "f" to variable node "$\boldsymbol{x}$"
$\boldsymbol{\mu}_\mathrm{P}^{(n)}$	Predicted mean vector	$\boldsymbol{\Sigma}_\mathrm{P}^{(n)}$	Predicted covariance matrix
$\boldsymbol{\mu}^{(n)}$	Posterior mean vector	$\boldsymbol{\Sigma}^{(n)}$	Posterior covariance matrix
$\bar{\mathbf{x}}_i^{(n)}$	Augmented state vector	$\bar{\mathbf{z}}_i^{(n)}$	Augmented measurement vector
$\boldsymbol{Q}^{(n)}$	Localization error matrix	$\boldsymbol{J}^{(n)}$	Fisher information matrix
$\mathscr{P}_\mathrm{NA}^{(n)}$	Optimization problem for node activation	$\mathscr{P}_\mathrm{NP}^{(n)}$	Optimization problem for node prioritization
$\zeta_i^{(n)}$	Channel access probability of agent i	$y_{ij}^{(n)}$	Amount of resources allocated to the measurement link pair (i,j)
$\chi_i^{(n)}$	Potential error reduction of agent i related to inter-node measurements	$\xi_{ij}^{(n)}$	Channel quality between node i and j

TABLE I: Notations of Important Quantities.

$1, 2, \ldots, n$ (i.e., all available information related to the performed measurements) can be expressed as

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

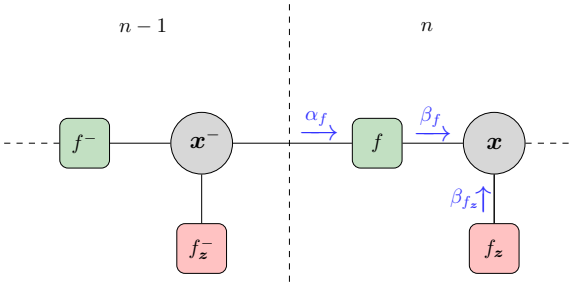


Fig. 2: Factor graph for single-node localization representing the factorization in (5). Nodes in green represent factors related to the state-evolution function, nodes in red represent factors related to the likelihood function,while messages related to the sum-product algorithm (SPA) are in blue. The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

.

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

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

The factor graph [18] representing this joint posterior for SBE is shown in Fig. 2. [1]

A. Temporal Fusion Based on SBE

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

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

The marginal posterior PDF $f(\boldsymbol{x}^{(n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ in (6) can be obtained from the joint posterior PDF $f(\boldsymbol{x}^{(1:n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ in (5) by marginalization. However, direct marginalization of $f(\boldsymbol{x}^{(1:n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ is infeasible 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 $f(\boldsymbol{x}^{(1:n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ has a structure as in (5). The exact calculation of $f(\boldsymbol{x}^{(n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ is then possible sequentially: at each time n, SBE consists of the *prediction step*

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

which is followed by the *update step*

$$\begin{aligned} &f\big(\boldsymbol{x}^{(n)}\big|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\big) \\ &\qquad\propto f\big(\boldsymbol{z}^{(n)}|\boldsymbol{x}^{(n)}\big) f\big(\boldsymbol{x}^{(n)}\big|\boldsymbol{z}^{(1:n-1)};\boldsymbol{u}^{(1:n)}\big). \end{aligned} \tag{8}$$

Contrary to direct marginalization where integration is performed over a 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 $f\big(\boldsymbol{x}^{(n)}\big|\boldsymbol{z}^{(1:n-1)};\boldsymbol{u}^{(n)}\big)$ 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 $f(\boldsymbol{x}^{(n)}|\boldsymbol{z}^{(1:n-1)};\boldsymbol{u}^{(n)})$ and temporal fusion is directly performed in the update step according to (8).

B. 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 way. In particular, the SPA avoids the curse of dimensionality inherent to direct marginalization. Therefore, SPA-based solutions are attractive for high

[1] For simplicity in notation, the index of the agent is dropped in the following, e.g., $\mathbf{x}_1^{(n)}$ is replaced by $\mathbf{x}^{(n)}$.

dimensional inference problems. The SPA is a so called "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 that in Fig. 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 pass 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 in Fig. 2, the message passed from factor node "f" to variable node "$\boldsymbol{x}$" is obtained as [18]

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

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 $\beta_{f^{\boldsymbol{z}}}(\boldsymbol{x}^{(n)}) = f(\boldsymbol{z}^{(n)}|\boldsymbol{x}^{(n)})$. After these two messages are calculated, the belief for "$\boldsymbol{x}$" is finally obtained as

$$b(\boldsymbol{x}^{(n)}) \propto \beta_f(\boldsymbol{x}^{(n)})\beta_{f^{\boldsymbol{z}}}(\boldsymbol{x}^{(n)})\,.$$

For $\alpha_f(\boldsymbol{x}^{(n-1)}) = b(\boldsymbol{x}^{(n-1)})$, it can be seen that $b(\boldsymbol{x}^{(n)}) = f(\boldsymbol{x}^{(n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ 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 Fig. 2.

C. 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 namely Kalman filtering and particle filtering.

1) 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

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

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

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

The EKF and the UKF are versions of the Kalman filter that are suitable for nonlinear state-evolution and measurement models. If $\boldsymbol{a}\big(\mathbf{x}^{(n-1)}, \mathbf{c}^{(n)}; \boldsymbol{u}^{(n)}\big)$ in (1) and $\boldsymbol{h}\big(\mathbf{x}^{(n)}, \mathbf{v}^{(n)}\big)$ in (3) are non-linear functions, the covariance matrices $\boldsymbol{\Sigma}_{\mathrm{p}}^{(n)}$ as well as $\boldsymbol{\Sigma}_{\mathbf{z}}^{(n)}$ and $\boldsymbol{\Sigma}_{\mathbf{xz}}^{(n)}$ 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 $\big[\boldsymbol{\mu}^{(n-1)\mathrm{T}}, \mathbf{0}^{\mathrm{T}}\big]^{\mathrm{T}}$ and $\big[\boldsymbol{\mu}_{\mathrm{p}}^{(n)\mathrm{T}}, \mathbf{0}^{\mathrm{T}}\big]^{\mathrm{T}}$, respectively [19]. In this way, an approximation of the matrices $\boldsymbol{\Sigma}_{\mathrm{p}}^{(n)}$, $\boldsymbol{\Sigma}_{\mathbf{z}}^{(n)}$, and $\boldsymbol{\Sigma}_{\mathbf{xz}}^{(n)}$ 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 where 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 $\boldsymbol{\mu}_{\mathrm{p}}^{(n)}$, $\boldsymbol{\Sigma}_{\mathrm{p}}^{(n)}$ as well as $\boldsymbol{\mu}_{\mathbf{z}}^{(n)}$, $\boldsymbol{\Sigma}_{\mathbf{z}}^{(n)}$, and $\boldsymbol{\Sigma}_{\mathbf{xz}}^{(n)}$ are calculated (see [11] for details). 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.

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

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

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

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

the mean of the weighted particles, i.e.,

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

A large variety of particle filtering algorithms have been introduced. In what follows, we review the prominent sequential importance resampling (SIR) 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 $f(\boldsymbol{x}^{(n)}|\boldsymbol{x}^{(n-1)};\boldsymbol{u}^{(n)})$ 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 $w_l^{(n)} = f\big(\boldsymbol{z}^{(n)}|\boldsymbol{x}_l^{(n)}\big) / \sum_{\ell=1}^{n_\mathrm{p}} f\big(\boldsymbol{z}^{(n)}|\boldsymbol{x}_\ell^{(n)}\big)$. 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 effective sample size is smaller than a threshold. In the resampling step, n_p "resampled" particles are obtained by sampling from $\tilde{f}(\boldsymbol{x}^{(n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ 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 non-linear SBE problems. However, their computational complexity is significantly increased compared to variants of the Kalman filter (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.

III. Network Localization for IoT

Consider the localization of a network of IoT devices that consists of N_a agents (with index set $\mathcal{N}_\mathrm{a} = \{1, 2, \ldots, N_\mathrm{a}\}$) and N_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 $\mathbf{x}^{(n)} \triangleq \big[\mathbf{x}_1^{(n)\mathrm{T}}, \mathbf{x}_2^{(n)\mathrm{T}}, \ldots, \mathbf{x}_{N_\mathrm{a}+N_\mathrm{b}}^{(n)\mathrm{T}}\big]^\mathrm{T}$. At time n, agent $i \in \mathcal{N}_\mathrm{a}$ is able to communicate and perform an inter-node 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 th e 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

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

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 $f\big(\boldsymbol{x}_i^{(n)}\big|\boldsymbol{x}_i^{(n-1)};\boldsymbol{u}_i^{(n)}\big)$. At $n = 0$, the prior PDF of the joint state vector can be expressed as $f\big(\boldsymbol{x}^{(0)}\big) = \prod_{i=1}^{N_\mathrm{a}+N_\mathrm{b}} f\big(\boldsymbol{x}_i^{(0)}\big)$. In particular, anchors $i \in \mathcal{N}_\mathrm{b}$ have perfect knowledge of their state, i.e., their prior PDFs are given by $f\big(\boldsymbol{x}_i^{(0)}\big) = \delta(\boldsymbol{x}_i^{(0)} - \tilde{\boldsymbol{x}}_i^{(0)})$ where $\tilde{\boldsymbol{x}}_i^{(0)}$ is the true state. Furthermore, agents have an uninformative prior information $f\big(\boldsymbol{x}_i^{(0)}\big)$ that is assumed known. For $n > 0$, the joint prior PDF, can be expressed as

$$\begin{aligned} f\big(\boldsymbol{x}^{(0:n)};\boldsymbol{u}^{(1:n)}\big) &= f\big(\boldsymbol{x}^{(0)}\big) \prod_{k=1}^{n} f\big(\boldsymbol{x}^{(k)}\big|\boldsymbol{x}^{(k-1)};\boldsymbol{u}^{(k)}\big) \\ &= \prod_{i=1}^{N_\mathrm{a}+N_\mathrm{b}} f\big(\boldsymbol{x}_i^{(0)}\big) \prod_{k=1}^{n} f\big(\boldsymbol{x}_i^{(k)}\big|\boldsymbol{x}_i^{(k-1)};\boldsymbol{u}_i^{(k)}\big). \end{aligned} \tag{14}$$

Agents $i \in \mathcal{N}_\mathrm{a}$ performs inter-node 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

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

where $\mathbf{v}_{ij}^{(n)}$ is the inter-node 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 also Section IV-B).

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 $f\big(\boldsymbol{z}_{ij}^{(n)}\big|\boldsymbol{x}_i^{(n)}, \boldsymbol{x}_j^{(n)}\big)$. The joint likelihood function can be expressed as

$$f\big(\boldsymbol{z}^{(1:n)}\big|\boldsymbol{x}^{(1:n)}\big) = \prod_{k=1}^{n} \prod_{i=1}^{N_\mathrm{a}} \prod_{j \in \mathcal{A}_i^{(k)}} f\big(\boldsymbol{z}_{ij}^{(k)}\big|\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\big). \tag{16}$$

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

$$\begin{aligned} &f\big(\boldsymbol{x}^{(0:n)}\big|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\big) \\ &\quad\propto f\big(\boldsymbol{z}^{(1:n)}\big|\boldsymbol{x}^{(1:n)}\big) f\big(\boldsymbol{x}^{(0:n)};\boldsymbol{u}^{(1:n)}\big) \\ &\quad= f\big(\boldsymbol{x}^{(0)}\big) \prod_{k=1}^{n} f\big(\boldsymbol{x}^{(k)}\big|\boldsymbol{x}^{(k-1)};\boldsymbol{u}^{(k)}\big) f\big(\boldsymbol{z}^{(k)}\big|\boldsymbol{x}^{(k)}\big) \qquad (17) \\ &\quad= \prod_{i=1}^{N_a+N_b} f\big(\boldsymbol{x}_i^{(0)}\big) \prod_{k=1}^{n} f\big(\boldsymbol{x}_i^{(k)}\big|\boldsymbol{x}_i^{(k-1)};\boldsymbol{u}_i^{(k)}\big) \\ &\qquad\times \prod_{j \in \mathcal{A}_i^{(k)}} f\big(\boldsymbol{z}_{ij}^{(k)}\big|\boldsymbol{x}_i^{(k)}, \boldsymbol{x}_j^{(k)}\big)\,. \qquad (18) \end{aligned}$$

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

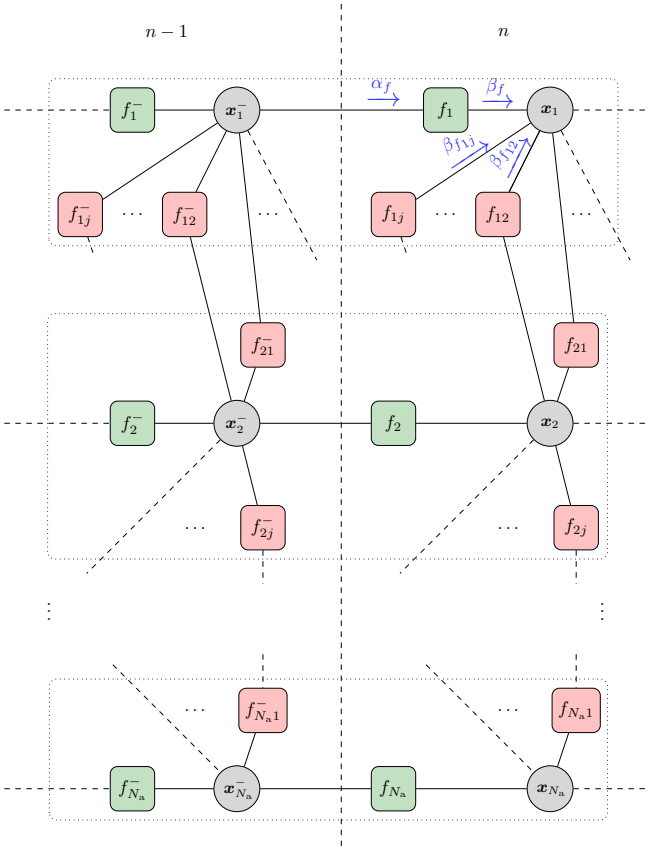


Fig. 3: Factor graph for network localization corresponding to the factorizes (18). Nodes in green represent factors related to the state-evolution function, nodes in red represent factors related to the likelihood function, while SPA messages are in blue. The following short notations are used: $\boldsymbol{x}_i^-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

of the marginal posterior in (18) is shown in Fig. 3. The *spatiotemporal structure* of the marginal posterior allows to develop distributed inference algorithms that are scalable both in time n and in the number of agents N_a as discussed in the next section.

A. Spatiotemporal Fusion Based on the SPA

In a network with multiple agents, state estimation is complicated by the fact that, since inter-node measurements are performed, the posterior distributions $f(\boldsymbol{x}_i^{(n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ 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 $f\big(\boldsymbol{x}^{(0:n)}\big|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\big)$ in (17) to obtain a marginal posterior PDF $f\big(\boldsymbol{x}^{(n)}\big|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)}\big)$ by means of an algorithm presented in Section II-C 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 as 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 Fig. 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[2]. 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 (i) applying SPA [18] to the factor graph in Fig. 3, (ii) performing temporal fusion by sending messages only forward in time, and (iii) 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

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

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

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

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 $b(\boldsymbol{x}_i^{(n)}) \approx f(\boldsymbol{x}_i^{(n)}|\boldsymbol{z}^{(1:n)};\boldsymbol{u}^{(1:n)})$ 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 Fig. 3 also exploits spatial structure. In particular,

[2]In the sense that the uncertainty of the estimates is underestimated by their spread.

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. In addition, as will be discussed next, message passing according to (19)–(22) almost automatically yields to a distributed implementation.

B. 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]. With this framework, spatiotemporal fusion is possible in a scalable and distributed way by directly applying arbitrary existing algorithms based on SBE such as those reviewed in Section II-C.

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

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

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

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

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

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

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

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 exchanging 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 (cf. (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).[3] Then each agent broadcasts $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$ and receives $\beta_{f_j}(\boldsymbol{x}_j^{(n)})$ from its neighbors $j\in\mathcal{A}_i^{(n)}$ such that $f(\bar{\boldsymbol{x}}_i^{(n)})$ in (24) becomes available at agent i.

[3]The prediction step of any algorithm based on SBE such as those presented in Section II-B can be used to calculate $\beta_{f_i}(\boldsymbol{x}_i^{(n)})$.

Step 2: Measurement Phase and State Update. Agent i cooperates with its neighbors $j\in\mathcal{A}_i^{(n)}$ to acquire inter-node measurements $\boldsymbol{z}_{ij}^{(n)}$. Now the likelihood function $f(\bar{\boldsymbol{z}}_i^{(n)}|\bar{\boldsymbol{x}}_i^{(n)})$ 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).[4]

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 $j\in\mathcal{A}_i^{(n)}$.

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 $|\mathcal{A}_i^{(n)}|$ 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 $|\mathcal{A}_j^{(n)}|$ 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 *Step 1 – Step 3* is that existing techniques for single-node localization and navigation can be directly leveraged for scalable and distributed network localization. Note that sigma point belief propagation (SPBP) [5] and the network localization algorithm in [6] have been developed according to Steps 1–3.

IV. Efficient Network Operation

Efficient network operation [21]–[26] is an indispensable part for localization in the IoT. The network operation strategies presented in this paper focus on the coordination of measurements provided by range measurement units (RMUs), i.e., the measurement model in (15) is

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

The performance of RMUs such as ultrawide-band (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; or
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 Fig. 4.

[4]Note that update step of any algorithm based on SBE such as those presented in Section II-B can be used to calculate $b(\bar{\boldsymbol{x}}_i^{(n)})$.

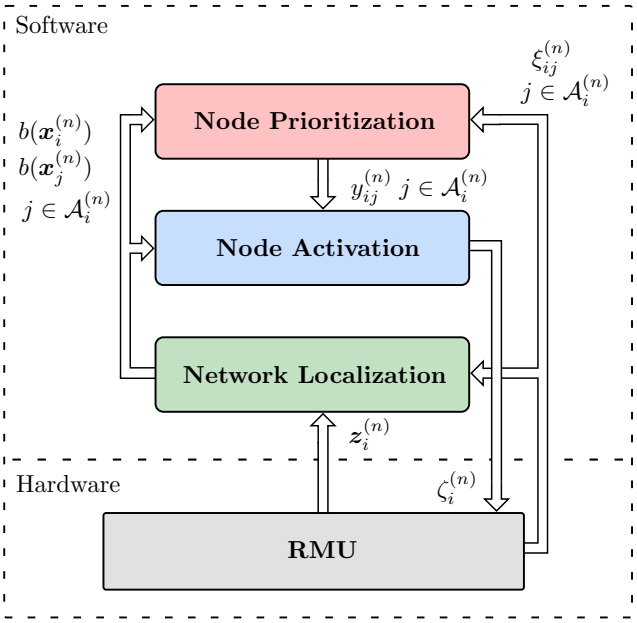


Fig. 4: Flowchart showing the interaction of node activation, node prioritization, network localization, and the RMU.

Note that in what follows, the inverse Fisher information matrix [3] is referred to as error matrix. In particular, all strategies developed in this paper 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 operations strategies [16] these error matrices are approximated by the corresponding covariance matrices which can be provided by network localization algorithms.

A. Node Activation

The goal of node activation strategies is to determine a set of nodes that are permitted to make range measurements, such 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.

1) Centralized Node Activation: If agent i is selected to make inter-node measurements with its neighbors at time n, the error evolution relationship is given by [24]

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

where $\boldsymbol{S}_{ij}^{(n)}$ denotes the information matrix corresponding to the measurement $z_{ij}^{(n+1)}$, and $\boldsymbol{\Delta}^{(n+1)}$ 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 next subsection [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

$$i_n = \max_{i \in \mathcal{N}_\mathrm{a}} \mathrm{tr}\Big(\big(\boldsymbol{Q}^{(n)}\big)^{-1} + \sum_{j \in \mathcal{A}_i^{(n)}} \boldsymbol{S}_{ij}^{(n)} \Big)^{-1}. \tag{27}$$

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 such 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.

2) Distributed Node Activation: Consider the case where 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

$$\begin{aligned} \mathscr{P}_\mathrm{NA}^{(n)}: \quad & \underset{\{\zeta_i^{(n)}\}_{i \in \mathcal{N}_\mathrm{a}}}{\text{minimize}} \quad \mathbb{E}\{\mathrm{tr}\{\mathbf{Q}^{(n+1)}\} | \mathbf{Q}^{(n)}\} \\ & \text{subject to} \quad 0 \leqslant \zeta_i^{(n)} \leqslant 1, \quad i \in \mathcal{N}_\mathrm{a} \end{aligned}$$

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

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

where $\chi_i^{(n)}$ denotes the expected *error reduction* of agent i, if it is activated and successfully makes range measurements with its neighbors $\mathcal{A}_i^{(n)}$ and $\Delta_j^{(n)}$ 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 $\chi_i^{(n)}$ 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 $\mathscr{P}_\mathrm{NA}^{(n)}$ leads to a non-random 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

Algorithm 1 Distributed Node Activation Strategy

for all $i \in \mathcal{N}_\mathrm{a}$ **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 $\zeta_i^{(n)}$ from (28);
    if $\zeta_i^{(n)} = 1$ **then**
      Access the channel and perform inter-node measurements;
    end if
  end if
  Broadcast $\Delta_j^{(n)}$;
end for

cause packet collisions. The possibility of such 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.

B. Node Prioritization

The goal of node prioritization is to achieve the best tradeoff between resource consumption and localization accuracy. In what follows, we again discuss centralized and distributed strategies for node prioritization.

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

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

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, Section III-A] and depends on the relative positions of nodes i and j. Furthermore, $y_{ij}^{(n)}$ are the variables to be optimized.[5]

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

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

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

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

where $\boldsymbol{M}$ is an auxiliary matrix for the SDP formulation [21]. Convex optimization engines [29] can be used to solve the SDP above. Note that similarly to the node activation problem, solving the node prioritization problem above 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 this reasons, centralized node prioritization is not scalable with the network size.

2) 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 ith diagonal $D \times D$ block of $\boldsymbol{Q}^{(n+1)}$, denoted by $\big[\boldsymbol{Q}^{(n+1)}\big]_i$. This error matrix depends on the geometry of the network and the accuracies of all inter-node measurements. Therefore, directly optimizing this error matrix does not lead to distributed implementation. An approximation of $\big[\boldsymbol{Q}^{(n+1)}\big]_i$ that involves only local parameters, can be introduced as follows

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

where

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

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

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

and $\boldsymbol{v}_{ij}^{(n)} \in \mathbb{R}^D$ is a unit vector representing the direction between node i and j. Note that $\widetilde{\boldsymbol{Q}}_i^{(n+1)}$ 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 $j \in \mathcal{A}_i^{(n)}$.

[5] As a special case, if only node i is activated, $\mathcal{E}^{(n)} = \{(i,j) : j \in \mathcal{A}_i^{(n)}\}$.

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

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

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

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

Remark 4: The node prioritization problem in noncooperative scenarios is a special case of $\mathscr{P}_{i,\mathrm{NP-D}}^{(n)}$ 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_tot it 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.

V. Case Study

In this section, we demonstrate the performance benefits of cooperation among devices and multi-sensor 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].[6]

A. 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 $\mathbf{p}_i^{(n)} = \big[p_{i,1}^{(n)}\ p_{i,2}^{(n)}\ p_{i,3}^{(n)}\big]^\mathrm{T} \in \mathbb{R}^3$, 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 $\mathbf{s}_i^{(n)} := \big[[\mathbf{p}_i^{(n)}]^\mathrm{T}\ [\dot{\mathbf{p}}_i^{(n)}]^\mathrm{T}\big]^\mathrm{T}$, are generated by means of a constant velocity motion model [17, Chapter 6.3.2]. 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, Chapter 6.3.2] 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 $\mathbf{f}_i^{(n)} = \big[f_{1,i}^{(n)}\ f_{2,i}^{(n)}\ f_{3,i}^{(n)}\big]^\mathrm{T}$ with elements $f_{k,i}^{(n)\mathrm{T}} = -\gamma_f \dot{\mathsf{p}}_{k,i}^{(n-1)}|\dot{\mathsf{p}}_{k,i}^{(n-1)}|$, $k\in\{1,2,3\}$. 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 maneuverabilities that are reasonable for micro UAVs. In particular, the maximum speed of each agent remains typically below 5.0m/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 $\sigma_\boldsymbol{\omega} = 0.5\mathrm{s}^{-1}$. 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 multi-sensor fusion, in the simulated algorithm, the measurements provided by the IMU are incorporated as deterministic control input $\boldsymbol{u}_i^{(n)} = \big[\boldsymbol{u}_{i,\boldsymbol{\varphi}}^{(n)\mathrm{T}}\ \boldsymbol{u}_{i,\boldsymbol{\omega}}^{(n)\mathrm{T}}\big]^\mathrm{T}$ (cf. (1)). 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 random variables (RVs)

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

where $\boldsymbol{\varphi}_i^{(n)}$ is the true acceleration of agent i in its local reference frame. The IMU noise $\mathbf{c}_i^{(n)} = \big[\mathbf{c}_{i,\boldsymbol{\varphi}}^{(n)\mathrm{T}}\ \mathbf{c}_{i,\boldsymbol{\omega}}^{(n)\mathrm{T}}\big]^\mathrm{T}$ 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 $\mathbf{x}_i^{(n)} = a_i\big(\mathbf{x}_i^{(n-1)}, \mathbf{c}_i^{(n)}; \boldsymbol{u}_i^{(n)}\big)$ is provided in [30].

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

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

[6] 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*.

where $\mathsf{v}_{ij}^{(n)} \sim \mathcal{N}(0, \sigma_{\mathsf{v}}^2)$ is the Gaussian noise with standard deviation $\sigma_{\mathsf{v}} = 0.1\text{m}$. A more detailed, technology-specific model for ranging from wideband radiofrequency (RF) signals can be found in [27], [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 52m. Moreover, if there are more then M potential neighbor nodes, M 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 Fig. 3 and thus the related negative effects such as overconfident beliefs. Note that the communication range of 52m was chosen such 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 Section III-B. Note that in the considered scenario with 512 UAVs, localization algorithms based on SBE are infeasible because they are note scalable in the number of agents [6, Section VII-C]. To the best of our knowledge, SPBP is the only available algorithm for cooperative location and orientation estimation in 3-D. 1200 time steps were simulated and 100 simulation runs were performed. Examples of true and estimated trajectories are shown in Fig. 5. As a metric for localization performance the 3-D localization error outage (LEO) was used.[7]

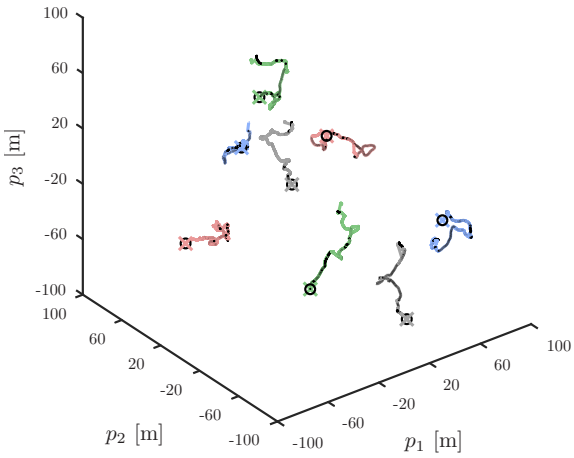


Fig. 5: Trajectories related to eight exemplary agents and one simulation run. Colored curves and black curves represent the estimated and the true trajectories, respectively. Similarly, colored crosses and black circles represent the estimated and true positions at the last time step, respectively.

B. Network Localization Results

To study the impact of cooperation among agents as well as the impact related to multi-sensor 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 $\sigma_{\boldsymbol{\omega}} = 5 \times 10^{-3}\text{s}^{-1}$.

Fig. 6 shows the LEOs (obtained by averaging over 100 simulation runs, 512 agents, and 900 time steps) versus threshold e_{th} for the four simulated configurations[8]. The following key observations can be obtained from these results: (i) very desirable localization performance can be obtained with "Spatial Cooperation + IMU Fusion". Specifically, the threshold e_{th} is 0.11m and 0.17m at a LEO of 10^{-1} and 10^{-2}, respectively. Remarkably, for $e_{\text{th}} \geqslant 0.3\text{m}$ the LEO is 0; (ii) 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_{th} is reduced from 0.54m to 0.11m (by 79.6%) at a LEO of 10^{-1} and from 4.21m to 0.17m (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; (iii) 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_{th} is reduced from 2.92m to 0.11m (by 96.2%) at a LEO of 10^{-1} and from 5.00m to 0.17m (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; (iv) due to the few neighbor nodes available for localization and the high mobility of the agents, the localization performance of "Baseline" is very poor.

[7]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_{th}.

[8]Since SPBP needs a certain number of time steps for initialization, for the network localization results, the first 300 times steps were not incorporate in the LEOs evaluation.

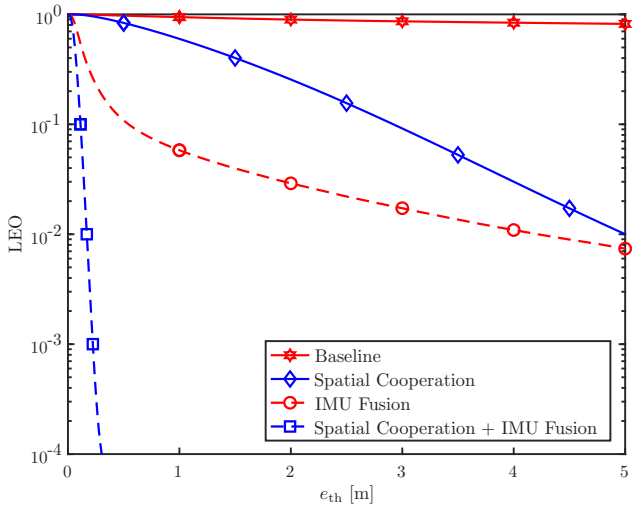


Fig. 6: LEO versus threshold e_{th} for the different simulated network localization configurations.

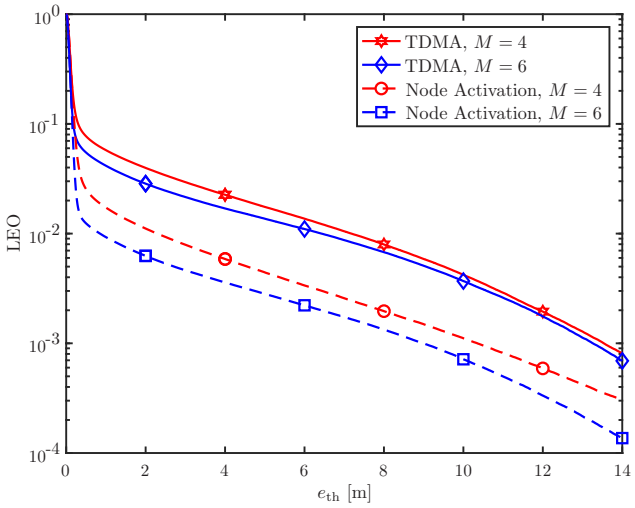


Fig. 7: LEO versus threshold e_{th} for the different channel access strategies and different M.

C. Network Operation Results

To demonstrate the benefits of network operation algorithms, a heterogeneous network that consists of two UAV classes was simulated. There are 256 UAVs in each class. The first class performs IMU measurements with noise standard deviations $\sigma_{\boldsymbol{\varphi}} = 3.3\times10^{-5}\,\text{m/s}^2$ and $\sigma_{\boldsymbol{\omega}} = 1.7\times10^{-3}\,\text{s}^{-1}$; the second class performs IMU measurements with noise standard deviations $\sigma_{\boldsymbol{\varphi}} = 3\times10^{-4}\,\text{m/s}^2$, and $\sigma_{\boldsymbol{\omega}} = 1.5\times10^{-2}\,\text{s}^{-1}$. All other parameters are as described in Section V-A and identical for both classes. For "Node Activation", spatial cooperation and IMU fusion was simulated together with the distributed network activation algorithm described in Section IV-A2 to control the RMU measurements. At each time step n, every UAV determines its channel access probability $\zeta_i^{(n)}$ according to (28) and if $\zeta_i^{(n)} = 1$, it tries to access the channel. In a certain step, if multiple UAVs of the same group (cf. Section V-A) that are also in the same subnetwork try to access the channel, only one randomly selected UAV is able to perform an RMU measurement. As a reference method, "TDMA" was simulated where, as in the previous Section V-B, spatial cooperation and IMU fusion with TDMA for channel access is performed. For both "Node Activation" and "TDMA", $M = 4$ and $M = 6$ were considered.

In the simulated scenario, "Node Activation" has a number of communication links related to RMU measurements that, compared to "TDMA", is 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 is 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 $1.7\times10^{-4}\,\text{J}$ per range measurement is used as RMU [16], "Node Activation" can reduce the overall energy consumption of the network for all 1200 time steps by $2.1\,\text{J}$ for $M = 4$ and by $4.2\,\text{J}$ for $M = 6$.

Fig. 7 shows the LEOs—obtained by averaging over 100 simulation runs, 512 agents, and 1200 time steps—versus threshold e_{th} for the four simulated configurations.[9] The following two observations can be made: (i) "Node Activation" can significantly increase localization accuracy. In particular, at a LEO of 10^{-2}, e_{th} is reduced from $7.18\,\text{m}$ to $2.29\,\text{m}$, i.e., by 68.1% for $M = 4$ and from $6.42\,\text{m}$ to $0.85\,\text{m}$, 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 on average. Note that the improvement in localization performance is most significant at the first time steps, during the initialization phase of the algorithm; (ii) incrementing M from 4 to 6 results in 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 then "TDMA" for $M = 6$. Thus it can be noted that a smart activation of agents can compensate for a low number of neighboring nodes.

VI. Final Remark

The scale and heterogeneity of Internet-of-Things (IoT) networks calls for a new class of localization algorithms. In this paper, we presented network localization and navigation (NLN), a paradigm that introduces scalable and distributed techniques for multi-sensor 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

[9]Note that the "Spatial Cooperation + IMU Fusion" results in Fig. 6 correspond to the identical scenario as the "TDMA, $M = 6$" results in Fig. 7. However, contrary to Fig. 6, in Fig. 7 all 1200 time steps are considered.

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 five hundreds of 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 consolidated that in IoT applications localization and navigation performance can be strongly increased by multi-sensor fusion and cooperation among devices.

ACKNOWLEDGMENT

The authors would like to thank B. Teague and T. Wang for stimulating discussions.

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, Sep. 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, Sep. 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, Jul. 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*. Springer, 2001.

[13] J. H. Kotecha and P. M. Djuric, "Gaussian sum particle filtering," *IEEE Trans. Signal Process.*, vol. 51, no. 10, pp. 2602–1612, 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. Commun.*, Guatemala City, Guatemala, Nov. 2017.

[17] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software*. New York City, NY, USA: John Wiley & Sons, 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*, ser. Prentice-Hall Information and System Sciences Series. Upper Saddle River, New Jersey 07458: Prentice Hall, 2000.

[20] F. Daum and J. Huang, "Curse of dimensionality and particle filters," in *Aerospace Conference, 2003. Proceedings. 2003 IEEE*, vol. 4, Mar. 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] ——, "Energy-efficient network navigation algorithms," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 7, pp. 1418–1430, Jul. 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, Jul. 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. on Ubiquitous Wireless Broadband*, Salamanca, Spain, Sep. 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, special issue on *Ultra-Wide Bandwidth (UWB) Technology & Emerging Applications*.

[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, UK: Cambridge University Press, 2004.

[30] J. D. Hol, F. Dijkstra, H. Luinge, and T. B. Schön, "Tightly coupled UWB/IMU pose estimation," in *Proc. IEEE ICUWB '09*, Vancouver, Canada, Sep. 2009, pp. 688–692.

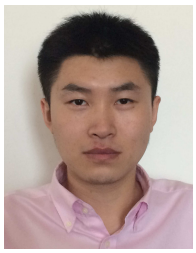
Moe Z. Win (S'85-M'87-SM'97-F'04) is a Professor at the Massachusetts Institute of Technology (MIT) and the founding director of the Wireless Information and Network Sciences Laboratory. Prior to joining MIT, he was with AT&T Research Laboratories and NASA Jet Propulsion Laboratory.

His research encompasses fundamental theories, algorithm design, and network experimentation for a broad range of real-world problems. His current research topics include network localization and navigation, network interference exploitation, and quantum information science. He has served the IEEE Communications Society as an elected Member-at-Large on the Board of Governors, as elected Chair of the Radio Communications Committee, and as an IEEE Distinguished Lecturer. Over the last two decades, he held various editorial positions for IEEE journals and organized numerous international conferences. Currently, he is serving on the SIAM Diversity Advisory Committee.

Dr. Win is an elected Fellow of the AAAS, the IEEE, and the IET. He was honored with two IEEE Technical Field Awards: the IEEE Kiyo Tomiyasu Award (2011) and the IEEE Eric E. Sumner Award (2006, jointly with R. A. Scholtz). Together with students and colleagues, his papers have received numerous awards. Other recognitions include the IEEE Communications Society Edwin H. Armstrong Achievement Award (2016), the International Prize for Communications Cristoforo Colombo (2013), the Copernicus Fellowship (2011) and the *Laurea Honoris Causa* (2008) from the Università degli Studi di Ferrara, and the U.S. Presidential Early Career Award for Scientists and Engineers (2004). He is an ISI Highly Cited Researcher.

Florian Meyer (S'12–M'15) received the Dipl.-Ing. (M.Sc.) and Ph.D. degrees (with highest honors) in electrical engineering from TU Wien, Vienna, Austria in 2011 and 2015, respectively.

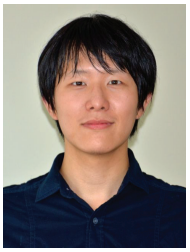
He was a visiting researcher with the Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden in 2013 and with the NATO Centre for Maritime Research and Experimentation (CMRE), La Spezia, Italy in 2014 and 2015. In 2016, he was with CMRE as a research scientist. Currently, Dr. Meyer is an Erwin Schrödinger Fellow at the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. He served as a TPC member of several IEEE conferences and is a co-chair of the IEEE ANLN Workshop at IEEE ICC 2018, Kansas City, MO, USA. His research interests include signal processing for wireless sensor networks, localization and tracking, information-seeking control, message passing algorithms, and finite set statistics.



Zhenyu Liu (S'15) received the B.Sc. (with honor) and the M.Sc. in electronic engineering from Tsinghua University, Beijing, China, in 2011 and 2014, respectively.

Mr. Liu is pursuing the Ph.D. degree with the Wireless Information and Network Sciences Laboratory at Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. His research interests include statistical inference and stochastic optimization, with applications to communications and localization.

Mr. Liu received the Best Paper Award at IEEE LATINCOM in 2017, and the first prize of the IEEE Communications Society Student Competition in 2016. He received academic excellence scholarships from Tsinghua University from 2008 to 2010. He served as a reviewer for the Proceedings of the IEEE, the IEEE Transactions on Vehicular Technology, the IEEE Communications Letters, and the IEEE Wireless Communications Letters.



Wenhan Dai (S'12) received the B.S. degrees in Electronic Engineering and in Mathematics from Tsinghua University in 2011, and the S.M. degree in Aeronautics and Astronautics at the Massachusetts Institute of Technology (MIT) in 2014.

He is pursuing the Ph.D. degree with Wireless Information and Network Sciences Laboratory at MIT. His research interests include communication theory and stochastic optimization with application to wireless communication and network localization. His current research focuses on efficient network localization, cooperative network operation, and ultra-wideband communication.

Mr. Dai was honored by the Marconi Society with the Paul Baran Young Scholar Award (2017). He received the Marconi-BISITE Best Paper Award from the IEEE ICUWB (2017), the Chinese Government Award for Outstanding Student Abroad (2016), the first prize of the IEEE Communications Society Student Competition (2016), and the Student Paper Award (first place) from the IEEE CWIT (2015). He was recognized as the exemplary reviewer of the IEEE Communications Letters (2014).

Stefania Bartoletti (S'12–M'16) received the Laurea (summa cum laude) degree in electronics and telecommunications engineering and the Ph.D. degree in information engineering from the University of Ferrara, Italy, in 2011 and 2015, respectively. Her research interests include the theory and experimentation of wireless networks for passive localization and physical behavior analysis. Dr. Bartoletti served as a Chair of the TPC for the IEEE ICC Workshop on Advances in Network Localization and Navigation in 2017 and 2018, and as a Reviewer for numerous IEEE journals and international conferences. Since 2016, she has been a recipient of the Marie Skłodowska-Curie Global Fellowship within the H2020 European Framework for a research project with the Massachusetts Institute of Technology and the University of Ferrara. She is a recipient of the 2016 Paul Baran Young Scholar Award of the Marconi Society.

Andrea Conti Andrea Conti (S'99–M'01–SM'11) received the Laurea (summa cum laude) degree in telecommunications engineering and the Ph.D. degree in electronic engineering and computer science from the University of Bologna, Bologna, Italy, in 1997 and 2001, respectively. He is currently an Associate Professor with the University of Ferrara, Ferrara, Italy. Prior to joining the University of Ferrara, he was with the Consorzio Nazionale Interuniversitario per le Telecomunicazioni and with the IEIIT-Consiglio Nazionale delle Ricerche. In Summer 2001, he was with the Wireless Systems Research Department, AT&T Research Laboratories. Since 2003, he has been a Frequent Visitor with the Wireless Information and Network Sciences Laboratory, Massachusetts Institute of Technology, Cambridge, MA, USA, where he currently holds the Research Affiliate appointment. His research interests include theory and experimentation of wireless systems and networks including network localization, distributed sensing, adaptive diversity communications, and network secrecy.

Dr. Conti was recipient of the HTE Puskás Tivadar Medal and co-recipient of the IEEE Communications Society's Stephen O. Rice Prize in the field of Communications Theory and of the IEEE Communications Society's Fred W. Ellersick Prize. He was an Editor for IEEE journals, as well as chaired international conferences. He has been Elected Chair of the IEEE Communications Society's Radio Communications Technical Committee. He is a Co-founder and Elected Secretary of the IEEE Quantum Communications & Information Technology Emerging Technical Subcommittee. He is an Elected Fellow of the IET and has been selected as an IEEE Distinguished Lecturer.