IEEE Trans. Signal Process., vol. 62, no. 12, June 15, 2014, pp. 3029–3041

Consensus-based Distributed Particle Filtering with Distributed Proposal Adaptation

Ondrej Hlinka, *Member, IEEE*, Franz Hlawatsch, *Fellow, IEEE*, and Petar M. Djurić, *Fellow, IEEE*

***Abstract*— We develop a distributed particle filter for sequential estimation of a global state in a decentralized wireless sensor network. A global state estimate that takes into account the measurements of all sensors is computed in a distributed manner, using only local calculations at the individual sensors and local communication between neighboring sensors. The paper presents two main contributions. First, the *likelihood consensus* scheme for distributed calculation of the joint likelihood function (used by the local particle filters) is generalized to arbitrary local likelihood functions. This generalization overcomes the restriction to exponential-family likelihood functions that limited the applicability of the original likelihood consensus (Hlinka *et al.*, "Likelihood consensus and its application to distributed particle filtering," *IEEE Trans. Signal Process.*, vol. 60, pp. 4334–4349, Aug. 2012). The second contribution is a consensus-based distributed method for adapting the proposal densities used by the local particle filters. This adaptation takes into account the measurements of all sensors, and it can yield a significant performance improvement or, alternatively, a significant reduction of the number of particles required for a given level of accuracy. The performance of the proposed distributed particle filter is demonstrated for a target tracking problem.**

***Index Terms*— Distributed particle filter, distributed proposal adaptation, distributed sequential estimation, likelihood consensus, target tracking, wireless sensor network.**

I. Introduction

For distributed sequential estimation in wireless sensor networks without a fusion center, distributed particle filters (DPFs) [1]–[8] are frequently the algorithms of choice because of their excellent performance in nonlinear/non-Gaussian scenarios. Applications of DPFs include machine and structural health monitoring, pollution source localization, surveillance, and target tracking. Wireless sensor networks are usually composed of battery-powered sensing/processing nodes, briefly referred to as "sensors" hereafter. To save battery power and prolong operation lifetime, DPFs must be able to operate with low-rate, short-distance communication.

In wireless sensor networks, the measurements are dispersed among the sensors, rather than available at one central location. The DPFs proposed in [2]–[8] use consensus algorithms to diffuse the locally available information throughout the network. Consensus algorithms allow an iterative distributed computation of networkwide aggregates such as sums, averages, and maxima relying only on communication between neighboring sensors [9]–[11]. There is no bottleneck or single point of failure, no routing algorithms are needed, and the algorithms are robust to changing network topologies and unreliable network conditions such as link failures.

In this paper, building on our previous work in [6] and [7], we develop a consensus-based DPF that uses only local processing and communication. Each sensor runs a local particle filter that computes a *global* estimate (i.e., reflecting the measurements of all sensors). As in the DPF presented in [6], this is possible because each sensor has knowledge of the joint likelihood function (JLF), which summarizes the measurements of all sensors. In [6], the *likelihood consensus* (LC) was proposed for a consensus-based distributed calculation of (an approximation of) the JLF. The LC uses only communication between neighboring sensors, without the need for a routing protocol. However, in its original form, the LC was restricted to likelihood functions from the exponential family [6]. Another restriction of the DPF proposed in [6] is that the proposal densities used by the local particle filters are not adapted to the measurements.

Here, we present two contributions, partially presented in [1] and [7], which constitute two advances over the DPF of [6]. First, we extend the LC to a general measurement model. This generalized LC (which is simply called LC in what follows) overcomes the restriction to likelihood functions from the exponential family. Second, we develop a distributed, consensus-based scheme for adapting the proposal densities used by the local particle filters. Adapted proposal densities can yield a significant performance improvement or, alternatively, a significant reduction of the required number of particles [12]. Furthermore, in our LC-based DPF, proposal density adaptation improves the accuracy of the JLF approximation underlying the LC method. Our distributed adaptation scheme provides each local particle filter with a global proposal density reflecting all the measurements. It differs from the scheme proposed in [13] in that it is distributed and enables a consensus-based computation of global proposal densities.

Consensus-based DPFs with proposal density adaptation have been recently proposed in [3], [5], and [8]. In [3], products of local particle weights are calculated by executing one consensus algorithm per particle. To reduce the number of particles and, thus, the communication requirements, a distributed proposal density adaptation scheme is employed. In

O. Hlinka and F. Hlawatsch are with the Institute of Telecommunications, Vienna University of Technology, A-1040 Vienna, Austria (e-mail: {ondrej.hlinka, franz.hlawatsch}@nt.tuwien.ac.at). P. M. Djurić is with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY, USA (e-mail: petar.djuric@stonybrook.edu). This work was supported by the FWF under Award S10603 within the National Research Network SISE and by the NSF under Awards CCF-1320626, ECCS-1346854, and CNS-1354614. This work was presented in part at IEEE ICASSP 2012, Kyoto, Japan, March 2012.

this scheme, a set capturing most of the posterior probability mass is constructed by means of min- and max-consensus algorithms. This set is used to calculate a distorted state-transition density, which serves as a proposal density. Our distributed proposal density adaptation scheme has a lower complexity than that of [3]. The communication requirements are somewhat higher, but the overall communication requirements of our DPF are still much lower than those of the DPF of [3], and simulation results demonstrate a comparable estimation performance. Furthermore, unlike our DPF, the DPF of [3] requires that the random number generators of the individual sensors be synchronized. In [5], a DPF constituting a distributed implementation of the so-called unscented particle filter is presented. This method employs a proposal density adaptation which, however, is not distributed, i.e., the proposal densities used at the various sensors are only based on local measurements. Simulation results demonstrate a better estimation performance of our DPF relative to that of [5], at the cost of higher communication requirements. Finally, for discrete-valued state vectors, a DPF using the optimal global proposal density computed by means of consensus algorithms is presented in [8].

This paper is organized as follows. The system model is described in Section II, and the basic DPF scheme using an adapted proposal density is presented in Section III. In Section IV, we consider the consensus-based distributed calculation of the JLF. Certain aspects of the underlying basis expansion approximation of the JLF are discussed in Section V. In Section VI, the LC method for a general JLF is formulated. In Section VII, a distributed proposal density adaptation scheme is presented. Finally, in Section VIII, the proposed LC-based DPF with proposal density adaptation is applied to a target tracking problem, and simulation results are reported.

II. System Model

We consider a wireless sensor network consisting of K sensors. A global state $\mathbf{x}_n = (x_{n,1} \cdots x_{n,M})^\top \in \mathbb{R}^M$ evolves with discrete time n according to

$$\mathbf{x}_n = \mathbf{g}_n(\mathbf{x}_{n-1}, \mathbf{u}_n)\,, \quad n = 1, 2, \ldots\,, \tag{1}$$

where $\mathbf{g}_n(\cdot,\cdot)$ is a nonlinear state-transition function and $\mathbf{u}_n$ is white driving noise with a known probability density function (pdf) $f(\mathbf{u}_n)$. At time n, the kth sensor ($k \in \{1,\ldots,K\}$) acquires a measurement $\mathbf{z}_{n,k} \in \mathbb{R}^{N_{n,k}}$ according to

$$\mathbf{z}_{n,k} = \mathbf{h}_{n,k}(\mathbf{x}_n, \mathbf{v}_{n,k})\,, \quad k \in \{1,\ldots,K\}, \tag{2}$$

where $\mathbf{h}_{n,k}(\cdot,\cdot)$ is a nonlinear measurement function and $\mathbf{v}_{n,k}$ is measurement noise with a known pdf $f(\mathbf{v}_{n,k})$. We assume that (i) $\mathbf{v}_{n,k}$ and $\mathbf{v}_{n',k'}$ are independent unless $(n,k)=(n',k')$; (ii) the initial state $\mathbf{x}_0$ and the sequences $\mathbf{u}_n$ and $\mathbf{v}_{n,k}$ are all independent; and (iii) sensor k knows $\mathbf{g}_n(\cdot,\cdot)$ and $\mathbf{h}_{n,k}(\cdot,\cdot)$ for all n, but it does not know $\mathbf{h}_{n,k'}(\cdot,\cdot)$ for $k' \neq k$.

The nonlinear and non-Gaussian state-space model (1), (2) together with our statistical assumptions determines the *state-transition pdf* $f(\mathbf{x}_n|\mathbf{x}_{n-1})$, the *local likelihood function* $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$, and the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$. Here, $\mathbf{z}_n \triangleq (\mathbf{z}_{n,1}^\top \cdots \mathbf{z}_{n,K}^\top)^\top$ contains all the sensor measurements at time n. Due to (2) and the independence of all $\mathbf{v}_{n,k}$, the JLF is equal to the product of all the local likelihood functions, i.e.,

$$f(\mathbf{z}_n|\mathbf{x}_n) \;=\; \prod_{k=1}^{K} f(\mathbf{z}_{n,k}|\mathbf{x}_n)\,. \tag{3}$$

We write $\mathbf{z}_{1:n} \triangleq (\mathbf{z}_1^\top \cdots \mathbf{z}_n^\top)^\top$ for the vector of the measurements of all sensors up to time n. From (1), (2), and our statistical assumptions, it also follows that

$$f(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{z}_{1:n-1}) \;=\; f(\mathbf{x}_n|\mathbf{x}_{n-1})\,, \tag{4}$$

$$f(\mathbf{z}_n|\mathbf{x}_n, \mathbf{z}_{1:n-1}) \;=\; f(\mathbf{z}_n|\mathbf{x}_n)\,. \tag{5}$$

Each sensor estimates the state $\mathbf{x}_n$ based on the measurements of all the sensors up to time n, $\mathbf{z}_{1:n}$, via an approximation of the minimum mean-square error (MMSE) estimator [14]

$$\hat{\mathbf{x}}_n^{\text{MMSE}} \;\triangleq\; \mathrm{E}\{\mathbf{x}_n|\mathbf{z}_{1:n}\} \;=\; \int \mathbf{x}_n\, f(\mathbf{x}_n|\mathbf{z}_{1:n})\, d\mathbf{x}_n\,. \tag{6}$$

Here, using (4) and (5), the current posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ can be obtained sequentially from the previous posterior $f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1})$, the state-transition pdf $f(\mathbf{x}_n|\mathbf{x}_{n-1})$, and the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ [15]. A computationally feasible approximation of this sequential MMSE state estimation is provided by the particle filter [15]–[18]. In a particle filter, the posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ is represented by a set of samples (or particles) $\{\mathbf{x}_n^{(j)}\}_{j=1}^J$ and corresponding weights $\{w_n^{(j)}\}_{j=1}^J$.

III. Consensus-based Distributed Particle Filtering

The DPF scheme we present extends that of [6] to a general JLF and to distributed proposal density adaptation. Each sensor tracks a particle representation of the global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ using a *local particle filter*. Each local particle filter calculates a global state estimate $\hat{\mathbf{x}}_{n,k}$ that is based on $\mathbf{z}_{1:n}$, i.e., the measurements of *all* the sensors up to time n. This requires knowledge of the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ (or of an approximation of $f(\mathbf{z}_n|\mathbf{x}_n)$) as a function of the state $\mathbf{x}_n$. This knowledge is obtained in a distributed manner by means of the LC described in Section VI. Furthermore, each local particle filter draws its particles from a globally adapted proposal density, which is calculated via the distributed adaptation scheme described in Section VII. The resulting DPF does not require communication between distant sensors or complex routing protocols; moreover, no particles, local state estimates, or measurements are communicated between the sensors. In Algorithm 1, we provide a statement of the proposed DPF scheme.

Algorithm 1: LC-based DPF

The local particle filter at sensor k is initialized at time $n=0$ by J particles $\{\tilde{\mathbf{x}}_{0,k}^{(j)}\}_{j=1}^J$ that are drawn from a prior pdf $f(\mathbf{x}_0)$. At time $n \geq 1$, the local particle filter at sensor k performs the following steps, which are identical for all k.

1) Temporary particles $\{\mathbf{x}_{n,k}^{\prime(j)}\}_{j=1}^J$ are randomly drawn from $f(\mathbf{x}_n|\tilde{\mathbf{x}}_{n-1,k}^{(j)}) \;\equiv\; f(\mathbf{x}_n|\mathbf{x}_{n-1})\big|_{\mathbf{x}_{n-1}=\tilde{\mathbf{x}}_{n-1,k}^{(j)}}$, and

a Gaussian approximation $f_{\mathrm{G}}(\mathbf{x}_n|\mathbf{z}_{1:n-1}) \triangleq \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}'_{n,k}, \mathbf{C}'_{n,k})$ of the "predicted posterior" $f(\mathbf{x}_n|\mathbf{z}_{1:n-1})$ is calculated according to

$$\boldsymbol{\mu}'_{n,k} = \frac{1}{J}\sum_{j=1}^{J} \mathbf{x}'^{(j)}_{n,k} \tag{7}$$

$$\mathbf{C}'_{n,k} = \frac{1}{J}\sum_{j=1}^{J} \mathbf{x}'^{(j)}_{n,k}\mathbf{x}'^{(j)\top}_{n,k} - \boldsymbol{\mu}'_{n,k}\boldsymbol{\mu}'^{\top}_{n,k}\,. \tag{8}$$

2) An adapted Gaussian proposal density involving all sensor measurements,

$$q(\mathbf{x}_n; \mathbf{z}_n) \triangleq \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_n, \mathbf{C}_n)\,, \tag{9}$$

is computed from all $\boldsymbol{\mu}'_{n,k'}$, $\mathbf{C}'_{n,k'}$, and $\mathbf{z}_{n,k'}$ (with $k' \in \{1,\ldots,K\}$) as described in Section VII. This step is jointly performed by all the sensors and requires communication with the respective neighbors.

3) J particles $\{\mathbf{x}^{(j)}_{n,k}\}_{j=1}^{J}$ are drawn from the proposal density $q(\mathbf{x}_n; \mathbf{z}_n)$.

4) Using these particles, an approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ is computed via the LC described in Section VI. This step is jointly performed by all the sensors and requires communication with the respective neighbors.

5) Nonnormalized weights associated with the particles $\mathbf{x}^{(j)}_{n,k}$ drawn in Step 3 are calculated as (cf. [13])

$$\tilde{w}^{(j)}_{n,k} = \frac{\tilde{f}(\mathbf{z}_n|\mathbf{x}^{(j)}_{n,k})\; f_{\mathrm{G}}(\mathbf{x}^{(j)}_{n,k}|\mathbf{z}_{1:n-1})}{q(\mathbf{x}^{(j)}_{n,k}; \mathbf{z}_n)}\,, \tag{10}$$

for $j \in \{1,\ldots,J\}$. This involves the approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ computed in Step 4 and the proposal density $q(\mathbf{x}_n; \mathbf{z}_n)$ computed in Step 2, which are evaluated at all particles $\mathbf{x}^{(j)}_{n,k}$. The weights $\tilde{w}^{(j)}_{n,k}$ are then normalized, i.e., $w^{(j)}_{n,k} = \tilde{w}^{(j)}_{n,k}/W_{n,k}$ with $W_{n,k} = \sum_{j=1}^{J}\tilde{w}^{(j)}_{n,k}$. The set $\{(\mathbf{x}^{(j)}_{n,k}, w^{(j)}_{n,k})\}_{j=1}^{J}$ provides a particle representation of the current global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$.

6) From $\{(\mathbf{x}^{(j)}_{n,k}, w^{(j)}_{n,k})\}_{j=1}^{J}$, an approximation of the global state estimate $\hat{\mathbf{x}}^{\mathrm{MMSE}}_n$ in (6) is computed as

$$\hat{\mathbf{x}}_{n,k} = \sum_{j=1}^{J} w^{(j)}_{n,k}\mathbf{x}^{(j)}_{n,k}\,.$$

7) Resampling of $\{(\mathbf{x}^{(j)}_{n,k}, w^{(j)}_{n,k})\}_{j=1}^{J}$ produces J resampled particles $\tilde{\mathbf{x}}^{(j)}_{n,k}$ with identical weights [17]. The $\tilde{\mathbf{x}}^{(j)}_{n,k}$ are obtained by sampling with replacement from the set $\{\mathbf{x}^{(i)}_{n,k}\}_{i=1}^{J}$, where $\mathbf{x}^{(i)}_{n,k}$ is sampled with probability $w^{(i)}_{n,k}$. They are then used in Step 1 at the next particle filter recursion (i.e., at time $n+1$).

In Algorithm 1, only Steps 2 and 4 require communication with neighboring sensors; all other steps are performed locally at each sensor. The computational complexity is linear in the number of particles J. The complexities and communication requirements of the proposal density adaptation (Step 2) and of the LC algorithm (Step 4) are briefly discussed in Sections VII-B and VI, respectively. Improved performance can be obtained by replacing in (10) the Gaussian approximation $f_{\mathrm{G}}(\mathbf{x}_n|\mathbf{z}_{1:n-1})$ of the predicted posterior pdf with a Gaussian mixture approximation as suggested in [13] or a particle-based approximation $\sum_{j=1}^{J} w^{(j)}_{n-1,k} f(\mathbf{x}_n|\mathbf{x}^{(j)}_{n-1,k})$ [19]. However, in these cases the complexity is increased since a mixture pdf needs to be evaluated for each particle $\mathbf{x}^{(j)}_{n,k}$.

IV. Distributed Calculation of the JLF

The JLF (or, at least, an approximation thereof) is needed at each sensor in Step 5 of Algorithm 1. A straightforward approach to distributed calculation of the JLF is to transmit the measurement of each sensor to all other sensors. This approach presupposes that each sensor knows the local likelihood functions of all other sensors, and it may require an excessive amount of communication, especially in the case of high-dimensional measurements. Therefore, we pursue an alternative, consensus-based approach to distributed JLF calculation and distributed particle filtering.

In the consensus-based technique for distributed JLF calculation presented in [3] and [4], each sensor locally evaluates its local likelihood function at each particle, i.e., $f(\mathbf{z}_{n,k}|\mathbf{x}^{(j)}_n)$ $\forall j \in \{1,\ldots,J\}$. Then, the JLF value $f(\mathbf{z}_n|\mathbf{x}^{(j)}_n) = \prod_{k=1}^{K} f(\mathbf{z}_{n,k}|\mathbf{x}^{(j)}_n)$ (see (3)) is computed for each particle (i.e., for each j) using a separate consensus algorithm. This technique, too, may require an excessive amount of communication, since the number of consensus algorithms executed in parallel is proportional to the number of particles, which is typically high. Furthermore, synchronized random number generators are required to ensure that identical particle sets $\{\mathbf{x}^{(j)}_n\}_{j=1}^{J}$ are drawn at each sensor.

In this section and in Sections V and VI, we present a consensus-based technique for distributed JLF calculation that typically has much lower communication requirements. As in [6], the basic idea is to employ consensus algorithms to calculate a *sufficient statistic* of the JLF at each sensor. Using the sufficient statistic, each sensor is then able to locally evaluate the JLF at each particle. Since the dimension of the sufficient statistic is typically much lower than the number of particles, the communication requirements are reduced. In this section, we discuss the definition of a suitable sufficient statistic. We first consider the special case where the JLF can be exactly characterized by a sufficient statistic allowing a consensus-based distributed calculation. For the general case, we then propose an approximation of the JLF in terms of a sufficient statistic (partially presented in [1] and [7]). Finally, we briefly review a special sufficient statistic approximation for JLFs belonging to the exponential family [6].

A. JLF with an Additive Sufficient Statistic

Let $\mathbf{t}_n(\mathbf{z}_n) = \big(t_{n,1}(\mathbf{z}_n) \cdots t_{n,R}(\mathbf{z}_n)\big)^{\top}$ be a sufficient statistic for the estimation problem corresponding to the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$. According to the Neyman-Fisher factorization theorem [14], the JLF can then be written as

$$f(\mathbf{z}_n|\mathbf{x}_n) = f_1(\mathbf{z}_n)\, f_2\big(\mathbf{t}_n(\mathbf{z}_n), \mathbf{x}_n\big)\,. \tag{11}$$

Thus, the global measurement $\mathbf{z}_n$ is summarized by the sufficient statistic $\mathbf{t}_n(\mathbf{z}_n)$ (note that the factor $f_1(\mathbf{z}_n)$ is irrelevant because it does not depend on $\mathbf{x}_n$). Therefore, a sensor that knows $\mathbf{t}_n(\mathbf{z}_n)$ and the function $f_2(\cdot,\cdot)$ is able to evaluate the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ (up to an irrelevant factor) for any value of $\mathbf{x}_n$.

Suppose further that the components of $\mathbf{t}_n(\mathbf{z}_n)$ have the additive form

$$t_{n,r}(\mathbf{z}_n) \,=\, \sum_{k=1}^{K} \tau_{n,k,r}(\mathbf{z}_{n,k})\,, \qquad \forall\, r \in \{1,\dots,R\}\,, \tag{12}$$

with some local, generally sensor-dependent functions $\tau_{n,k,r}(\mathbf{z}_{n,k})$. We assume that sensor k knows $f_2(\cdot,\cdot)$ and its own functions $\tau_{n,k,r}(\cdot)$ but not the functions of the other sensors, i.e., $\tau_{n,k',r}(\cdot)$ for $k' \neq k$. Note that $f_2(\cdot,\cdot)$ represents a global knowledge that must be provided to each sensor beforehand. Because of the sum expression (12), consensus algorithms [9]–[11] can be used to calculate $\mathbf{t}_n(\mathbf{z}_n)$ and, thus, the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ in a distributed manner. After convergence of the consensus algorithms, each sensor obtains the JLF as a function of the state $\mathbf{x}_n$. Hence, it is able to compute the particle weights of its local particle filter according to Step 5 of Algorithm 1. After a finite number of consensus iterations, the sufficient statistic is obtained only approximately, with slight differences between the various sensors. This may lead to a slight degradation in estimation performance but does not represent a fundamental problem.

Example—Identical Local Likelihood Functions Belonging to the Exponential Family: An example of an additive sufficient statistic of the form (12) is given by local likelihood functions that are all identical (note that the local measurements $\mathbf{z}_{n,k}$ are still different in general) and belong to the exponential family of distributions [20], i.e.,

$$f(\mathbf{z}_{n,k}|\mathbf{x}_n) \,=\, c_n(\mathbf{z}_{n,k})\, \exp\big(\mathbf{a}_n^{\mathsf{T}}(\mathbf{x}_n)\mathbf{b}_n(\mathbf{z}_{n,k}) - d_n(\mathbf{x}_n)\big)\,, \qquad \forall\, k \in \{1,\dots,K\}, \tag{13}$$

with some generally time-dependent but sensor-independent functions $c_n(\cdot)\in\mathbb{R}_+$, $\mathbf{a}_n(\cdot)\in\mathbb{R}^R$, $\mathbf{b}_n(\cdot)\in\mathbb{R}^R$, and $d_n(\cdot)\in\mathbb{R}_+$. In particular, $\mathbf{b}_n(\mathbf{z}_{n,k})$ is the sufficient statistic of the local likelihood function (13). Using (3), we obtain for the JLF

$$f(\mathbf{z}_n|\mathbf{x}_n) \,\propto\, \exp\big(\mathbf{a}_n^{\mathsf{T}}(\mathbf{x}_n)\mathbf{t}_n(\mathbf{z}_n) - K d_n(\mathbf{x}_n)\big)\,, \tag{14}$$

where $\mathbf{t}_n(\mathbf{z}_n) = \sum_{k=1}^{K} \mathbf{b}_n(\mathbf{z}_{n,k})$ is the associated sufficient statistic. Thus, the JLF, too, belongs to the exponential family, and its sufficient statistic is obtained as the sum of the local sufficient statistics $\mathbf{b}_n(\mathbf{z}_{n,k})$. The components of the sufficient statistic of the JLF are hence given by

$$t_{n,r}(\mathbf{z}_n) \,=\, \sum_{k=1}^{K} b_{n,r}(\mathbf{z}_{n,k})\,, \qquad \forall\, r \in \{1,\dots,R\}\,, \tag{15}$$

where $b_{n,r}(\mathbf{z}_{n,k})$ is the rth component of the local sufficient statistic $\mathbf{b}_n(\mathbf{z}_{n,k})$. It is seen that the $t_{n,r}(\mathbf{z}_n)$ are in additive form (cf. (12)), and thus they can be calculated using consensus algorithms. Comparing (14) with (11), we furthermore see that $f_2\big(\mathbf{t}_n(\mathbf{z}_n),\mathbf{x}_n\big) = \exp\big(\mathbf{a}_n^{\mathsf{T}}(\mathbf{x}_n)\mathbf{t}_n(\mathbf{z}_n) - K d_n(\mathbf{x}_n)\big)$.

We note that an additive sufficient statistic of the form (12) is obtained even if, in (13), $c_n(\cdot)$, $\mathbf{b}_n(\cdot)$, and $d_n(\cdot)$ are replaced by sensor-dependent functions $c_{n,k}(\cdot)$, $\mathbf{b}_{n,k}(\cdot)$, and $d_{n,k}(\cdot)$. Here, (15) becomes $t_{n,r}(\mathbf{z}_n) = \sum_{k=1}^{K} b_{n,k,r}(\mathbf{z}_{n,k})$, and the term $K d_n(\mathbf{x}_n)$ in the exponent in (14) is replaced by $\sum_{k=1}^{K} d_{n,k}(\mathbf{x}_n)$. Thus, calculation of the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ according to (14) requires the function $\bar{d}_n(\cdot) = \sum_{k=1}^{K} d_{n,k}(\cdot)$ to be known at each sensor. This issue will be addressed in a generalized setting in Section IV-C.

B. General JLF

The assumption of an additive sufficient statistic in (12) is quite restrictive in practice. In particular, the example just considered is not encountered very often because the local likelihood functions at the various sensors are usually different (e.g., because they depend on the sensor location).

For a general JLF, i.e., a JLF without an additive sufficient statistic (12), a consensus-based distributed computation can still be obtained by introducing a suitable approximation of the JLF. To derive this approximation, we first take the logarithm of the product (3), assuming that $f(\mathbf{z}_{n,k}|\mathbf{x}_n) > 0$ for all $\mathbf{x}_n$:

$$\log f(\mathbf{z}_n|\mathbf{x}_n) \,=\, \sum_{k=1}^{K} \log f(\mathbf{z}_{n,k}|\mathbf{x}_n)\,. \tag{16}$$

We then use the following approximate (finite-order) basis expansions of the local log-likelihood functions $\log f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ viewed as functions of $\mathbf{x}_n$:

$$\log f(\mathbf{z}_{n,k}|\mathbf{x}_n) \,\approx\, \sum_{r=1}^{R} \lambda_{n,k,r}(\mathbf{z}_{n,k})\, \varphi_{n,r}(\mathbf{x}_n)\,. \tag{17}$$

Here, the $\varphi_{n,r}(\mathbf{x}_n)$ are fixed basis functions that are sensor-independent and assumed to be known to all sensors; the $\lambda_{n,k,r}(\mathbf{z}_{n,k})$ are expansion coefficients that contain all sensor-local information, including the sensor measurement $\mathbf{z}_{n,k}$; and R is the order of the basis expansion. Note that the basis functions $\varphi_{n,r}(\cdot)$ may be time-varying, although usually time-invariant basis functions $\varphi_r(\cdot)$ are more convenient. The choice of the basis functions is considered in Section V-B. The expansion coefficients $\lambda_{n,k,r}(\mathbf{z}_{n,k})$ are calculated locally at each sensor as discussed in Section V-A.

Substituting (17) into (16) and changing the order of summation, we obtain for the log-JLF

$$\log f(\mathbf{z}_n|\mathbf{x}_n) \,\approx\, \sum_{r=1}^{R} t_{n,r}(\mathbf{z}_n)\, \varphi_{n,r}(\mathbf{x}_n)\,, \tag{18}$$

with

$$t_{n,r}(\mathbf{z}_n) \,=\, \sum_{k=1}^{K} \lambda_{n,k,r}(\mathbf{z}_{n,k})\,. \tag{19}$$

By exponentiating (18), we finally obtain the following approximation of the JLF, denoted $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$:

$$f(\mathbf{z}_n|\mathbf{x}_n) \,\approx\, \tilde{f}(\mathbf{z}_n|\mathbf{x}_n) \,\triangleq\, \exp\left(\sum_{r=1}^{R} t_{n,r}(\mathbf{z}_n)\, \varphi_{n,r}(\mathbf{x}_n)\right) \tag{20}$$

$$= \exp\big(\boldsymbol{\varphi}_n^{\mathsf{T}}(\mathbf{x}_n)\mathbf{t}_n(\mathbf{z}_n)\big)\,. \tag{21}$$

Here, $\mathbf{t}_n(\mathbf{z}_n) = \big(t_{n,1}(\mathbf{z}_n)\cdots t_{n,R}(\mathbf{z}_n)\big)^{\mathsf{T}}$, with the components $t_{n,r}(\mathbf{z}_n)$ given by (19), and $\boldsymbol{\varphi}_n(\mathbf{x}_n) = \big(\varphi_{n,1}(\mathbf{x}_n)\cdots$

$\varphi_{n,R}(\mathbf{x}_n)\big)^\top$. The sum over all sensors in (19) and, thus, the "sufficient statistic" $\mathbf{t}_n(\mathbf{z}_n)$ can be computed in a distributed manner by means of consensus algorithms. Once $\mathbf{t}_n(\mathbf{z}_n)$ is available, each sensor is able to calculate the approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ for all values of $\mathbf{x}_n$ by evaluating (21), using the known basis $\boldsymbol{\varphi}_n(\cdot)$. This distributed calculation of the general JLF is further discussed in Sections V and VI.

The following observations can be made:

1) The consensus-based distributed calculation of the approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ via (21) and (19) is based on the fact that in each term of the sum (17), the known sensor-local information (in particular, the measurement $\mathbf{z}_{n,k}$) and the unknown state $\mathbf{x}_n$ are contained in separate factors. As a consequence, the sufficient statistic components $t_{n,r}(\mathbf{z}_n)$ in (19) do not depend on the state $\mathbf{x}_n$ but contain the information from all sensors (i.e., the expansion coefficients $\lambda_{n,k,r}(\mathbf{z}_{n,k})$ for all k), and they are sums over all sensors in which each term contains only local information of a single sensor. Furthermore, the unknown state $\mathbf{x}_n$ enters into the expression of $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ only via the basis functions $\varphi_{n,r}(\cdot)$, which are sensor-independent and known to each sensor.
2) The approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ in (21) conforms to the exponential-family expression (14), with the basis function vector $\boldsymbol{\varphi}_n(\mathbf{x}_n)$ taking the place of $\mathbf{a}_n(\mathbf{x}_n)$ and $\mathbf{t}_n(\mathbf{z}_n)$ assuming the interpretation of an (approximate) sufficient statistic. In fact, the basis expansion approximation (17) can be viewed as a means of enforcing the exponential-family structure (14).
3) The basis expansion approximation (17) distorts the shape of the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$. This distortion becomes weaker as the approximation (17) becomes more accurate (usually, for a higher expansion order R).

C. JLF Belonging to the Exponential Family—Case of Non-identical Local Likelihood Functions

Let us now consider the practically important special case where the local likelihood functions $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ belong to the exponential family but, in contrast to the example in Section IV-A, are not identical for all sensors k. That is,

$$f(\mathbf{z}_{n,k}|\mathbf{x}_n) = c_{n,k}(\mathbf{z}_{n,k}) \exp\big(\mathbf{a}_{n,k}^\top(\mathbf{x}_n)\mathbf{b}_{n,k}(\mathbf{z}_{n,k}) - d_{n,k}(\mathbf{x}_n)\big), \quad \forall\, k \in \{1,\dots,K\}, \quad (22)$$

with some possibly time- and sensor-dependent functions $c_{n,k}(\cdot) \in \mathbb{R}_+$, $\mathbf{a}_{n,k}(\cdot) \in \mathbb{R}^q$, $\mathbf{b}_{n,k}(\cdot) \in \mathbb{R}^q$, and $d_{n,k}(\cdot) \in \mathbb{R}_+$, with arbitrary $q \in \mathbb{N}$. This case was studied in our previous work [6]. We next show how it is related to the general basis function expansion considered in Section IV-B.

Inserting (22) into (3), we obtain the JLF as

$$f(\mathbf{z}_n|\mathbf{x}_n) = \bar{c}_n(\mathbf{z}_n) \exp\big(\bar{\mathbf{a}}_n^\top(\mathbf{x}_n)\bar{\mathbf{b}}_n(\mathbf{z}_n) - \bar{d}_n(\mathbf{x}_n)\big),$$

with

$$\bar{c}_n(\mathbf{z}_n) = \prod_{k=1}^K c_{n,k}(\mathbf{z}_{n,k})$$
$$\bar{\mathbf{a}}_n(\mathbf{x}_n) = \big(\mathbf{a}_{n,1}^\top(\mathbf{x}_n) \cdots \mathbf{a}_{n,K}^\top(\mathbf{x}_n)\big)^\top$$
$$\bar{\mathbf{b}}_n(\mathbf{z}_n) = \big(\mathbf{b}_{n,1}^\top(\mathbf{z}_{n,1}) \cdots \mathbf{b}_{n,K}^\top(\mathbf{z}_{n,K})\big)^\top \quad (23)$$
$$\bar{d}_n(\mathbf{x}_n) = \sum_{k=1}^K d_{n,k}(\mathbf{x}_n).$$

It is seen that the JLF, too, belongs to the exponential family. The sufficient statistic of the JLF, $\bar{\mathbf{b}}_n(\mathbf{z}_n)$ in (23), is the stacked vector containing the sufficient statistic vectors of the local likelihood functions of the individual sensors k, $\mathbf{b}_{n,k}(\mathbf{z}_{n,k})$; it is not given by a sum over all sensors as was the case in Section IV-A (cf. (12) and (15)). Hence, a direct consensus-based computation is not possible, and we again resort to an approximation. Although we could use the general basis expansion approximation (17), in [6] we pursued an alternative approach by using separate basis expansion approximations of the functions $\mathbf{a}_{n,k}(\mathbf{x}_n)$ and $d_{n,k}(\mathbf{x}_n)$ [6, equations (11) and (12)]. Substituting these approximations in (22), we obtain the following approximation of the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ [6, equation (16)]:

$$\tilde{f}(\mathbf{z}_n|\mathbf{x}_n) \propto \exp\left(\sum_{r=1}^{R_a} A_{n,r}(\mathbf{z}_n)\phi_{n,r}(\mathbf{x}_n) - \sum_{r=1}^{R_d} \Gamma_{n,r}\psi_{n,r}(\mathbf{x}_n)\right). \quad (24)$$

Here, both $A_{n,r}(\mathbf{z}_n)$ and $\Gamma_{n,r}$ are given by sums [6, equation (15)], which can again be computed in a distributed manner using consensus algorithms. We can rewrite (24) as

$$\tilde{f}(\mathbf{z}_n|\mathbf{x}_n) \propto \exp\big(\boldsymbol{\zeta}_n^\top(\mathbf{x}_n)\mathbf{t}_n(\mathbf{z}_n)\big),$$

with $\boldsymbol{\zeta}_n(\mathbf{x}_n) = \big(\boldsymbol{\phi}_n^\top(\mathbf{x}_n) \;\; \boldsymbol{\psi}_n^\top(\mathbf{x}_n)\big)^\top$ and $\mathbf{t}_n(\mathbf{z}_n) = \big(\boldsymbol{\alpha}_n^\top(\mathbf{z}_n) \;\; \boldsymbol{\gamma}_n^\top\big)^\top$, where $\boldsymbol{\phi}_n(\mathbf{x}_n) = \big(\phi_{n,1}(\mathbf{x}_n) \cdots \phi_{n,R_a}(\mathbf{x}_n)\big)^\top$, $\boldsymbol{\psi}_n(\mathbf{x}_n) = \big(\psi_{n,1}(\mathbf{x}_n) \cdots \psi_{n,R_d}(\mathbf{x}_n)\big)^\top$, $\boldsymbol{\alpha}_n(\mathbf{z}_n) = \big(A_{n,1}(\mathbf{z}_n) \cdots A_{n,R_a}(\mathbf{z}_n)\big)^\top$, and $\boldsymbol{\gamma}_n = -\big(\Gamma_{n,1} \cdots \Gamma_{n,R_d}\big)^\top$. This is seen to be of the same form as (21), and thus, within the limits of our approximation, $\mathbf{t}_n(\mathbf{z}_n)$ can again be interpreted as a sufficient statistic of the JLF. (Note that $\boldsymbol{\gamma}_n$ is constant in $\mathbf{z}_n$, but is included in the sufficient statistic to obtain $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ in the desired form of (21).) The sufficient statistic $\mathbf{t}_n(\mathbf{z}_n) = \big(\boldsymbol{\alpha}_n^\top(\mathbf{z}_n) \;\; \boldsymbol{\gamma}_n^\top\big)^\top$ is generally different from the sufficient statistic in (21), because the two sufficient statistics are induced by two different types of basis expansion approximations. However, both sufficient statistics can be computed in a distributed manner using consensus algorithms.

Special Case—Additive Gaussian Noise: As an important special case of the class of exponential-family local likelihood functions, we next consider the measurement model (cf. (2))

$$\mathbf{z}_{n,k} = \mathbf{m}_{n,k}(\mathbf{x}_n) + \mathbf{v}_{n,k}, \quad k \in \{1,\dots,K\}, \quad (25)$$

where $\mathbf{m}_{n,k}(\cdot)$ is some nonlinear function and $\mathbf{v}_{n,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{n,k})$ is Gaussian measurement noise, with $\mathbf{v}_{n,k}$ and $\mathbf{v}_{n',k'}$ independent unless $(n,k) = (n',k')$. The local likelihood function $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ corresponding to (25) belongs to the exponential family and is given by (22) with

$$\mathbf{a}_{n,k}(\mathbf{x}_n) = \mathbf{m}_{n,k}(\mathbf{x}_n),$$
$$\mathbf{b}_{n,k}(\mathbf{z}_{n,k}) = \mathbf{Q}_{n,k}^{-1}\mathbf{z}_{n,k},$$

Type of JLF	Sensor k calculates an approximation of:	Sensor k knows:
Additive sufficient statistic (Section IV-A)	—	$f_2(\cdot,\cdot)$, $\{\tau_{n,k,r}(\mathbf{z}_{n,k})\}_{r=1}^R$
Exponential family—identical local likelihood functions (Section IV-A)	—	$\mathbf{a}_n(\cdot)$, $\mathbf{b}_n(\mathbf{z}_{n,k})$, $d_n(\cdot)$
General JLF (Section IV-B)	$\log f(\mathbf{z}_{n,k}\vert\mathbf{x}_n)$	$f(\mathbf{z}_{n,k}\vert\,\cdot)$
Exponential family—nonidentical local likelihood functions (Section IV-C)	$\mathbf{a}_{n,k}(\mathbf{x}_n)$, $d_{n,k}(\mathbf{x}_n)$	$\mathbf{a}_{n,k}(\cdot)$, $\mathbf{b}_{n,k}(\mathbf{z}_{n,k})$, $d_{n,k}(\cdot)$
Exponential family—additive Gaussian noise (Section IV-C)	$\mathbf{m}_{n,k}(\mathbf{x}_n)$	$\mathbf{m}_{n,k}(\cdot)$, $\mathbf{Q}_{n,k}$

TABLE I
JLF TYPES, FUNCTIONS APPROXIMATED BY A BASIS EXPANSION, AND REQUIRED INFORMATION.

$$c_{n,k}(\mathbf{z}_{n,k}) \;=\; \tilde{c}_{n,k}\exp\!\left(-\frac{1}{2}\mathbf{z}_{n,k}^{\mathsf{T}}\mathbf{Q}_{n,k}^{-1}\mathbf{z}_{n,k}\right),$$
$$d_{n,k}(\mathbf{x}_n) \;=\; \frac{1}{2}\mathbf{m}_{n,k}^{\mathsf{T}}(\mathbf{x}_n)\,\mathbf{Q}_{n,k}^{-1}\,\mathbf{m}_{n,k}(\mathbf{x}_n)\,, \qquad (26)$$

with $\tilde{c}_{n,k} = 1/\sqrt{(2\pi)^M \det \mathbf{Q}_{n,k}}$. As proposed in our previous work [6], an approximation of $d_{n,k}(\mathbf{x}_n)$ can be obtained in an indirect way by substituting in (26) a basis expansion approximation of $\mathbf{m}_{n,k}(\mathbf{x}_n)$. This means that the basis expansion approximations of the functions $\mathbf{a}_{n,k}(\mathbf{x}_n) = \mathbf{m}_{n,k}(\mathbf{x}_n)$ and $d_{n,k}(\mathbf{x}_n)$ are coupled in that they are both induced by the basis expansion approximation of $\mathbf{m}_{n,k}(\mathbf{x}_n)$. Since only one basis approximation has to be calculated, this approach results in a reduced computational complexity. In Section VIII-C, we compare the estimation performance of the LC-based DPF using this approximation with that of the LC-based DPF using the general approximation (17).

D. Discussion

The different JLF types and basis expansion approximations considered above are summarized in Table I. For each case, we specify the function(s) of the state $\mathbf{x}_n$ (if any) that are approximated at each sensor using a basis expansion, as well as the information that is known at each sensor (besides the local measurement $\mathbf{z}_{n,k}$). Note that except for the function $f_2(\cdot,\cdot)$ required in the "additive sufficient statistic" case, which reflects properties of all sensors in the network, all other functions and quantities that a sensor needs to know are either available locally or can be calculated locally. Hence, they need not be disseminated in a preparatory stage.

V. BASIS EXPANSION APPROXIMATION

In this section, we discuss the calculation of the expansion coefficients and the choice of the basis functions. We consider the case of a general JLF (see Section IV-B) and the basis expansion approximation of $\log f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ in (17). However, our discussion also applies with obvious modifications to the other JLF types considered in Section IV.

A. Least Squares Approximation

The calculation of the local expansion coefficients $\{\lambda_{n,k,r}(\mathbf{z}_{n,k})\}_{r=1}^R$ in (17) is a straightforward adaptation of the least squares calculation described in [6] and is summarized only briefly. Recall that the particles $\{\mathbf{x}_{n,k}^{(j)}\}_{j=1}^J$ drawn in Step 3 of Algorithm 1 are part of a representation of the global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$. The $\lambda_{n,k,r}(\mathbf{z}_{n,k})$ are now calculated such that the sum of the squared errors of the approximation (17) at the $\mathbf{x}_{n,k}^{(j)}$, i.e., $\sum_{j=1}^J\big[\log f(\mathbf{z}_{n,k}|\mathbf{x}_{n,k}^{(j)}) - \sum_{r=1}^R \lambda_{n,k,r}(\mathbf{z}_{n,k})\,\varphi_{n,r}(\mathbf{x}_{n,k}^{(j)})\big]^2$, is minimized. (We assume that $f(\mathbf{z}_{n,k}|\mathbf{x}_{n,k}^{(j)}) \neq 0$; particles for which $f(\mathbf{z}_{n,k}|\mathbf{x}_{n,k}^{(j)}) = 0$ are discarded.) Note that with this cost function, the approximation of $\log f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ is best around those state values where the approximate JLF $\tilde{f}(\mathbf{z}_{n,k}|\mathbf{x}_n)$ is actually evaluated by the local particle filters (cf. (10)). The resulting least squares solution is given by [21]

$$\boldsymbol{\lambda}_{n,k} \;=\; \left(\boldsymbol{\Phi}_{n,k}^{\mathsf{T}}\boldsymbol{\Phi}_{n,k}\right)^{-1}\boldsymbol{\Phi}_{n,k}^{\mathsf{T}}\,\boldsymbol{\eta}_{n,k}\,,$$

with $\boldsymbol{\lambda}_{n,k} = \big(\lambda_{n,k,1}(\mathbf{z}_{n,k})\cdots\lambda_{n,k,R}(\mathbf{z}_{n,k})\big)^{\mathsf{T}} \in \mathbb{R}^R$, $\boldsymbol{\eta}_{n,k} = \big(\log f(\mathbf{z}_{n,k}|\mathbf{x}_{n,k}^{(1)})\cdots\log f(\mathbf{z}_{n,k}|\mathbf{x}_{n,k}^{(J)})\big)^{\mathsf{T}} \in \mathbb{R}^J$, and

$$\boldsymbol{\Phi}_{n,k} = \begin{pmatrix} \varphi_{n,1}(\mathbf{x}_{n,k}^{(1)}) & \cdots & \varphi_{n,R}(\mathbf{x}_{n,k}^{(1)}) \\ \vdots & & \vdots \\ \varphi_{n,1}(\mathbf{x}_{n,k}^{(J)}) & \cdots & \varphi_{n,R}(\mathbf{x}_{n,k}^{(J)}) \end{pmatrix} \in \mathbb{R}^{J\times R}.$$

Here, we assume that $J \geq R$—i.e., the number of particles $\mathbf{x}_{n,k}^{(j)}$ is not smaller than the number of basis functions $\varphi_{n,r}(\mathbf{x}_n)$—and that the columns of $\boldsymbol{\Phi}_{n,k}$ are linearly independent, so that $\boldsymbol{\Phi}_{n,k}^{\mathsf{T}}\boldsymbol{\Phi}_{n,k}$ is nonsingular. Also, the basis functions $\{\varphi_{n,r}(\cdot)\}_{r=1}^R$ are assumed known to sensor k.

B. Basis Functions

Possible choices of the basis functions $\varphi_{n,r}(\cdot)$ used in (17) include monomials (see [6]), orthogonal polynomials, and Fourier basis functions (to be discussed below). Our simulations suggest that the differences in DPF performance for different choices of the basis functions tend to be rather small if the least squares approximation of Section V-A is used (see Table II in Section VIII-C). This is because the approximation error is minimized locally at those state values where the JLF is actually evaluated by the local particle filters.

The polynomial basis approximation—using monomials as basis functions $\varphi_{n,r}(\cdot)$—was considered in [6]. Here, we discuss the Fourier basis approximation as an alternative. We assume n-independent basis functions $\varphi_r(\cdot)$ for simplicity. Consider first a 1D Fourier basis $\big\{\tilde{\varphi}_{\tilde{r}}(x)\big\}_{\tilde{r}=1}^{2\tilde{R}+1}$ defined for a scalar state variable x, with basis functions given as

$$\tilde{\varphi}_{\tilde{r}}(x) = \begin{cases} 1\,, & \tilde{r} = 1\,, \\ \cos\big(\frac{2\pi}{d_a}(\tilde{r}-1)\,x\big)\,, & \tilde{r} = 2,\ldots,\tilde{R}+1\,, \\ \sin\big(\frac{2\pi}{d_a}(\tilde{r}-1-\tilde{R})\,x\big)\,, & \tilde{r} = \tilde{R}+2,\ldots,2\tilde{R}+1\,. \end{cases}$$

Here, d_a denotes the length of the interval within which estimation of the 1D state x is being envisaged (e.g., in the target tracking application considered in Section VIII, d_a is the side length of the square region within which the target location can be tracked). A basis for our MD parameter $\mathbf{x}_n \in \mathbb{R}^M$ can then be constructed as the MD product basis $\{\varphi_{\tilde{\mathbf{r}}}(\mathbf{x}_n)\}_{\tilde{\mathbf{r}}\in\{1,\ldots,2\tilde{R}+1\}^M}$ induced by the 1D Fourier basis $\{\tilde{\varphi}_{\tilde{r}}(x)\}_{\tilde{r}=1}^{2\tilde{R}+1}$. The basis functions are thus given by

$$\varphi_{\tilde{\mathbf{r}}}(\mathbf{x}_n) = \prod_{i=1}^{M} \tilde{\varphi}_{\tilde{r}_i}(x_{n,i}), \quad \tilde{\mathbf{r}} \in \{1,\ldots,2\tilde{R}+1\}^M,$$

where $\tilde{r}_i \triangleq (\tilde{\mathbf{r}})_i$ and $x_{n,i} \triangleq (\mathbf{x}_n)_i$. Using an index transformation that maps the MD index $\tilde{\mathbf{r}} = (\tilde{r}_1 \cdots \tilde{r}_M) \in \{1,\ldots,2\tilde{R}+1\}^M$ onto the 1D index $r \in \{1,\ldots,R\}$, with $R = (2\tilde{R}+1)^M$, we can reindex the basis $\{\varphi_{\tilde{\mathbf{r}}}(\mathbf{x}_n)\}_{\tilde{\mathbf{r}}\in\{1,\ldots,2\tilde{R}+1\}^M}$ as $\{\varphi_r(\mathbf{x}_n)\}_{r=1}^R$, i.e., in the form used in (17).

VI. Likelihood Consensus for the General JLF

In Algorithm 2, we state the LC algorithm for distributed approximate calculation of a general JLF (cf. Section IV-B). This algorithm relies on the basis expansion approximation described in (17)–(21). The distributed calculation of the JLF for the cases considered in Sections IV-A and IV-C is analogous with obvious modifications.

Algorithm 2: Likelihood Consensus (LC)

At each time n, the following steps are performed by sensor k (analogous steps are performed by all sensors simultaneously).

1) Using the local measurement $\mathbf{z}_{n,k}$, the coefficients $\{\lambda_{n,k,r}(\mathbf{z}_{n,k})\}_{r=1}^R$ of the approximation (17) are computed locally by means of the least squares approximation reviewed in Section V-A.
2) The sums of the coefficients $\lambda_{n,k,r}(\mathbf{z}_{n,k})$ of all sensors k, $t_{n,r}(\mathbf{z}_n) = \sum_{k=1}^K \lambda_{n,k,r}(\mathbf{z}_{n,k})$ for $r \in \{1,\ldots,R\}$ (cf. (19)), are calculated in a distributed way using an average consensus algorithm [11], [22]. (Alternatively, a gossip algorithm [23] can be used.) If the communication graph of the sensor network is connected, then, after a sufficient number I of consensus iterations, all sensors obtain $\frac{1}{K} t_{n,r}(\mathbf{z}_n)$ and, hence, $t_{n,r}(\mathbf{z}_n)$ with sufficient accuracy. We note that one instance of the consensus algorithm is executed for each $r \in \{1,\ldots,R\}$; all instances are executed in parallel.
3) Using (20) and the approximate $t_{r,n}(\mathbf{z}_n)$ obtained in Step 2, each sensor is able to evaluate (up to small errors due to insufficiently converged consensus algorithms) the approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ for any value of $\mathbf{x}_n$.

Regarding Step 2 of Algorithm 2, we note that if the network's communication graph is connected, then for $I \to \infty$, each sensor k would obtain the exact average $\frac{1}{K} t_{n,r}(\mathbf{z}_n) = \frac{1}{K}\sum_{k'=1}^K \lambda_{n,k',r}(\mathbf{z}_{n,k'})$ [11]. After only a finite number I of iterations, the computed values are, in general, (slightly) different from the desired value $\frac{1}{K} t_{n,r}(\mathbf{z}_n)$ and also different at different sensors k. For application of the LC in our DPF (Algorithm 1), this may lead to some performance degradation of the DPF but does not represent a fundamental problem. Furthermore, since $t_{n,r}(\mathbf{z}_n)$ and not $\frac{1}{K} t_{n,r}(\mathbf{z}_n)$ is needed for JLF evaluation using (20), it is clear that each sensor has to know the total number of sensors, K. This information may be provided to each sensor beforehand, or some distributed algorithm for counting the number of sensors may be employed (e.g., [24]).

In the LC algorithm, the number of consensus algorithms executed in parallel is $N_c = R$. This is also the number of real values broadcast by each sensor to its neighbors in each consensus iteration. The total number of real values broadcast by each sensor at time n is then given by

$$C_{\mathrm{LC}} = N_c I = RI\,, \tag{27}$$

where I is the number of iterations performed by each consensus algorithm. The fact that the communication requirements of LC do not depend on the dimensions $N_{n,k}$ of the measurement vectors $\mathbf{z}_{n,k}$ is a major advantage in the case of high-dimensional measurements (e.g., in camera networks). On the other hand, R usually grows with the dimension M of the state vector $\mathbf{x}_n$ (see Section V-B).

The most complex part of the LC algorithm is the least squares approximation in Step 2. The complexity of the least squares approximation grows linearly with the number of particles J and cubically with the number of basis functions R.

VII. Distributed Proposal Density Adaptation

The performance of a particle filter is better when the proposal density $q(\mathbf{x}_n; \mathbf{z}_n)$ is more similar to the posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ [15]. A simple and common approach is to use the state-transition pdf $f(\mathbf{x}_n|\mathbf{x}_{n-1})$ as proposal density. However, since the measurements $\mathbf{z}_{1:n}$ are not taken into account, many of the particles may be located in regions with small likelihood and, thus, small posterior, which degrades the performance of the particle filter. To incorporate $\mathbf{z}_{1:n}$ into the proposal density and obtain particles in regions of large posterior, *proposal adaptation* can be used. An adapted proposal density typically yields improved performance and/or allows the number of particles to be reduced [12]. Because optimal proposal adaptation [25] is usually difficult, suboptimal adaptation methods are commonly employed. These methods use a parametric representation of the proposal density (e.g., a Gaussian or Gaussian mixture pdf), whose parameters are calculated by means of a simpler nonlinear filter.

In a DPF, it is desirable that the local particle filters use a *globally* adapted proposal density that incorporates the measurements of all sensors. Therefore, we next present a consensus-based distributed scheme for global proposal adaptation. Because the adapted proposal densities used by the local particle filters reflect the measurements of all sensors, our scheme is especially suitable for DPFs where the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$—involving the all-sensors measurement vector $\mathbf{z}_n$—is employed at each sensor to calculate the weights of the local particle filter. We consider, in particular, the LC-based

DPF described in Section III (cf. Steps 2 and 5 of Algorithm 1). However, the proposal density adaptation scheme can also be used in other DPFs.

A. Distributed Calculation of a Global Proposal Density

Our distributed proposal density adaptation scheme can be summarized as follows. First, a "predistorted" local posterior is calculated locally at each sensor. Next, a Gaussian approximation of the global posterior[1] is obtained by fusing all predistorted local posteriors via a consensus-based distributed fusion rule. This approximate global posterior, which takes into account the measurements of all sensors, is used as a proposal density by the local PFs. Our overall approach is inspired by that in [2] but employs a different predistortion that enables the use of a Gaussian filter for calculating the predistorted local posteriors. Furthermore, in [2], the approximate global posterior is not used for proposal density adaptation but directly for filtering.

To develop the method, we first note that the global posterior can be written, up to a normalization factor, as

$$\begin{aligned} f(\mathbf{x}_n|\mathbf{z}_{1:n}) &= f(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_n) \\ &\propto f(\mathbf{z}_n|\mathbf{x}_n, \mathbf{z}_{1:n-1})\, f(\mathbf{x}_n|\mathbf{z}_{1:n-1}) \\ &= f(\mathbf{z}_n|\mathbf{x}_n)\, f(\mathbf{x}_n|\mathbf{z}_{1:n-1})\,, \end{aligned}$$

where Bayes' rule and (5) have been used. With (3), we obtain further

$$f(\mathbf{x}_n|\mathbf{z}_{1:n}) \propto \left[\prod_{k=1}^{K} f(\mathbf{z}_{n,k}|\mathbf{x}_n)\right] f(\mathbf{x}_n|\mathbf{z}_{1:n-1})\,. \tag{28}$$

Let us now suppose that each sensor k calculates the following predistorted, nonnormalized "local pseudoposterior":

$$\tilde{f}(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_{n,k}) \triangleq f(\mathbf{z}_{n,k}|\mathbf{x}_n) \left(f(\mathbf{x}_n|\mathbf{z}_{1:n-1})\right)^{1/K}. \tag{29}$$

We then obtain for the product of all local pseudoposteriors

$$\prod_{k=1}^{K} \tilde{f}(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_{n,k}) = \left[\prod_{k=1}^{K} f(\mathbf{z}_{n,k}|\mathbf{x}_n)\right] f(\mathbf{x}_n|\mathbf{z}_{1:n-1}) \tag{30}$$

$$\propto f(\mathbf{x}_n|\mathbf{z}_{1:n})\,, \tag{31}$$

where (28) has been used. Thus, the product of all local pseudoposteriors equals the global posterior up to a factor.

The global posterior reflects all the sensor measurements and could be employed as the global proposal density. However, for a simple distributed computation of (30), we use Gaussian approximations of the local pseudoposteriors and of the global posterior, i.e.,

$$\tilde{f}(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_{n,k}) \approx \mathcal{N}(\mathbf{x}_n; \tilde{\boldsymbol{\mu}}_{n,k}, \tilde{\mathbf{C}}_{n,k}) \tag{32}$$

and (cf. (9))

$$f(\mathbf{x}_n|\mathbf{z}_{1:n}) \approx q(\mathbf{x}_n; \mathbf{z}_n) = \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_n, \mathbf{C}_n)\,. \tag{33}$$

[1]Note that this approximate global posterior is different from the global posterior obtained in Step 5 of Algorithm 1. However, both approximate the true global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$.

Inserting the approximations (32) and (33) into the relation $f(\mathbf{x}_n|\mathbf{z}_{1:n}) \propto \prod_{k=1}^{K} \tilde{f}(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_{n,k})$ in (31), we obtain

$$\mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_n, \mathbf{C}_n) \propto \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}_n; \tilde{\boldsymbol{\mu}}_{n,k}, \tilde{\mathbf{C}}_{n,k})\,.$$

This relation shows how the global mean $\boldsymbol{\mu}_n$ and the global covariance $\mathbf{C}_n$—which are used in the global proposal density $q(\mathbf{x}_n; \mathbf{z}_n) = \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_n, \mathbf{C}_n)$—depend on the local means $\tilde{\boldsymbol{\mu}}_{n,k}$ and local covariances $\tilde{\mathbf{C}}_{n,k}$. More specifically, using the rules for a product of Gaussian densities [2], [26], we obtain the following explicit expressions:

$$\boldsymbol{\mu}_n = \mathbf{C}_n \sum_{k=1}^{K} \tilde{\mathbf{C}}_{n,k}^{-1} \tilde{\boldsymbol{\mu}}_{n,k}\,, \quad \mathbf{C}_n = \left(\sum_{k=1}^{K} \tilde{\mathbf{C}}_{n,k}^{-1}\right)^{-1}. \tag{34}$$

The sums over all sensors k in these expressions can be computed in a distributed manner using consensus algorithms.

The calculation of the Gaussian approximations $\mathcal{N}(\mathbf{x}_n; \tilde{\boldsymbol{\mu}}_{n,k}, \tilde{\mathbf{C}}_{n,k})$ of the local pseudoposteriors $\tilde{f}(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_{n,k})$ at the individual sensors (see (32)) is based on the following two observations. First, (29) can be interpreted as the update step of a Bayesian filter using the predistorted predicted posterior $\left(f(\mathbf{x}_n|\mathbf{z}_{1:n-1})\right)^{1/K}$ instead of the true predicted posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n-1})$. Second, each sensor calculated a Gaussian approximation of the predicted posterior,

$$f(\mathbf{x}_n|\mathbf{z}_{1:n-1}) \approx \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}'_{n,k}, \mathbf{C}'_{n,k})\,, \tag{35}$$

in Step 1 of Algorithm 1 (see (7) and (8)), and hence $\boldsymbol{\mu}'_{n,k}$ and $\mathbf{C}'_{n,k}$ are available at sensor k. It can be easily verified that the Gaussian approximation (35) entails

$$\left(f(\mathbf{x}_n|\mathbf{z}_{1:n-1})\right)^{1/K} \approx \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}'_{n,k}, K\mathbf{C}'_{n,k})\,. \tag{36}$$

According to (32) and (36), Gaussian approximations are used for both $\tilde{f}(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_{n,k})$ and $\left(f(\mathbf{x}_n|\mathbf{z}_{1:n-1})\right)^{1/K}$. Within these approximations, the update in (29) can be written as

$$\mathcal{N}(\mathbf{x}_n; \tilde{\boldsymbol{\mu}}_{n,k}, \tilde{\mathbf{C}}_{n,k}) = f(\mathbf{z}_{n,k}|\mathbf{x}_n)\, \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}'_{n,k}, K\mathbf{C}'_{n,k})\,.$$

We propose to perform this update by means of the update step of a Gaussian filter, with input mean $\boldsymbol{\mu}'_{n,k}$, input covariance $K\mathbf{C}'_{n,k}$, and measurement $\mathbf{z}_{n,k}$. As a result of this Gaussian filter update step, which is performed locally at each sensor, $\tilde{\boldsymbol{\mu}}_{n,k}$ and $\tilde{\mathbf{C}}_{n,k}$ are obtained. Examples of a Gaussian filter include the extended Kalman filter [27], the unscented Kalman filter [28], [29], the cubature Kalman filter [30], and the filters described in [31]. For the simulations presented in Section VIII, we use an unscented Kalman filter because it achieves better estimation performance than an extended Kalman filter with a computational complexity that is much lower than that of a particle filter. The unscented Kalman filter represents random variables using a minimal set of deterministically chosen sample points (so-called sigma points). These sigma points are propagated through the nonlinear system model (1) and (2) and used to calculate approximations of the posterior mean and covariance. It has been shown [32] that these approximations are at least as accurate as those resulting from a linearization of the system model.

In Algorithm 3, we list the operations of the proposed scheme.

ALGORITHM 3: DISTRIBUTED PROPOSAL DENSITY ADAPTATION

At time n, sensor k performs the following steps.

1) The mean $\tilde{\boldsymbol{\mu}}_{n,k}$ and covariance $\tilde{\mathbf{C}}_{n,k}$ of the Gaussian approximation (32) of the local pseudoposterior, $\mathcal{N}(\mathbf{x}_n; \tilde{\boldsymbol{\mu}}_{n,k}, \tilde{\mathbf{C}}_{n,k}) \approx \tilde{f}(\mathbf{x}_n|\mathbf{z}_{1:n-1}, \mathbf{z}_{n,k})$, are computed. This is done locally by performing a Gaussian filter update step with the following inputs: mean $\boldsymbol{\mu}'_{n,k}$, covariance $K\mathbf{C}'_{n,k}$, and measurement $\mathbf{z}_{n,k}$. Here, $\boldsymbol{\mu}'_{n,k}$ and $\mathbf{C}'_{n,k}$ were obtained locally according to (7) and (8), respectively. This Gaussian filter update step approximates (29) and produces $\tilde{\boldsymbol{\mu}}_{n,k}$ and $\tilde{\mathbf{C}}_{n,k}$.
2) The terms $\tilde{\mathbf{C}}_{n,k}^{-1}\tilde{\boldsymbol{\mu}}_{n,k}$ and $\tilde{\mathbf{C}}_{n,k}^{-1}$ of the sums in (34) are computed locally. This requires the inversion of the $M\times M$ matrix $\tilde{\mathbf{C}}_{n,k}$.
3) The sums over all sensors k in (34) are obtained by means of consensus algorithms. This step requires communication with neighboring sensors.
4) After convergence of the consensus algorithms, each sensor obtains $\sum_{k=1}^{K}\tilde{\mathbf{C}}_{n,k}^{-1}\tilde{\boldsymbol{\mu}}_{n,k}$ and $\sum_{k=1}^{K}\tilde{\mathbf{C}}_{n,k}^{-1}$. From these quantities, $\boldsymbol{\mu}_n$ and $\mathbf{C}_n$ are determined locally according to (34); this requires one matrix inversion and one matrix-vector multiplication. Each sensor k is now able to sample from the global proposal density $q(\mathbf{x}_n;\mathbf{z}_n) = \mathcal{N}(\mathbf{x}_n;\boldsymbol{\mu}_n,\mathbf{C}_n)$ (cf. (33)), as required in Step 3 of Algorithm 1, and to evaluate $q(\mathbf{x}_n;\mathbf{z}_n)$ at arbitrary values of $\mathbf{x}_n$, as required in Step 5 of Algorithm 1 (see (10)).

B. Discussion

As a result of Algorithm 3, all the sensors obtain (almost) identical $\boldsymbol{\mu}_n$ and $\mathbf{C}_n$ provided that the consensus algorithms are sufficiently converged. Therefore, we omit the subscript k indicating the sensor dependence, i.e., we write $\boldsymbol{\mu}_n$ instead of $\boldsymbol{\mu}_{n,k}$ and $\mathbf{C}_n$ instead of $\mathbf{C}_{n,k}$, for all k. We note that differences between the means and covariances obtained at the individual sensors (caused by insufficiently converged consensus algorithms) may lead to a certain performance degradation but do not represent a fundamental problem.

In Step 3, one consensus algorithm is executed for each entry of the M-dimensional vector $\tilde{\mathbf{C}}_{n,k}^{-1}\tilde{\boldsymbol{\mu}}_{n,k}$ and for each entry in the upper triangular part of the symmetric $M\times M$ matrix $\tilde{\mathbf{C}}_{n,k}^{-1}$. Thus, $M+M(M+1)/2$ consensus algorithms are executed in parallel, and hence, at time n, a total of

$$C_{\text{PA}} = \left[M + \frac{M(M+1)}{2}\right] I = \frac{(M^2+3M)I}{2} \tag{37}$$

real values are broadcast by each sensor to its neighbors. Here, I denotes the number of consensus iterations.

Using Algorithm 3 within Algorithm 1 requires C_{PA} additional transmissions per sensor per time step. On the other hand, the DPF performance is significantly improved, since the particles are sampled in regions of large likelihood. Alternatively, a performance similar to that without proposal density adaptation can be achieved with fewer particles. These benefits are verified in Section VIII-C. Furthermore, the least squares approximation (see Section V-A) tends to become more accurate because proposal density adaptation typically reduces the size of the state-space region that is occupied by the particles, and thus the local log-likelihood function needs to be approximated in a smaller region. This is especially important in the initial phase of estimation. Without proposal density adaptation, a broad, uninformative prior has to be used, which can lead to poor performance since the local log-likelihood functions have to be approximated in a large state-space region.

Furthermore, proposal density adaptation introduces an additional coupling between the sensors. Because essentially the same proposal density is used at each sensor, the sampled particles lie in practically identical state-space regions. As a consequence, one obtains accurate least squares approximations of the local log-likelihood functions in practically identical state-space regions. This tends to improve the accuracy of the resulting JLF approximation.

The increase in computational complexity of the LC-based DPF (Algorithm 1) due to proposal density adaptation is small, especially when an unscented Kalman filter is used in Step 1 of Algorithm 3. The complexity of the unscented Kalman filter grows cubically with the state dimension M [29], and it is typically almost negligible compared to the complexity of the local particle filter. The complexities of the matrix inversion and matrix-vector multiplications performed in Steps 2 and 4 are cubic and quadratic in M, respectively.

VIII. NUMERICAL STUDY

A. Target Tracking

We consider the problem of tracking a target that moves in the x-y plane, using sensors that measure the power of a signal emitted by the target. The target is represented by the state vector $\mathbf{x}_n \triangleq (x_n\ y_n\ \nu_n\ \theta_n)^\top$ containing the target's 2D location $\boldsymbol{\rho}(\mathbf{x}_n) \triangleq (x_n\ y_n)^\top$ as well as the magnitude ν_n and bearing θ_n of the target's 2D velocity. Thus, $M=4$. The state $\mathbf{x}_n$ evolves according to (cf. (1))

$$\begin{aligned} x_n &= x_{n-1} + u_{1,n} + (\nu_{n-1}+u_{3,n})\cos(\theta_{n-1}+u_{4,n}) \\ y_n &= y_{n-1} + u_{2,n} + (\nu_{n-1}+u_{3,n})\sin(\theta_{n-1}+u_{4,n}) \\ \nu_n &= \nu_{n-1} + u_{3,n} \\ \theta_n &= \theta_{n-1} + u_{4,n}\,, \end{aligned} \tag{38}$$

where $\mathbf{u}_n = (u_{1,n}\ u_{2,n}\ u_{3,n}\ u_{4,n})^\top \sim \mathcal{N}(\mathbf{0}_4, \mathbf{C}_u)$ is Gaussian driving noise, with $\mathbf{u}_n$ and $\mathbf{u}_{n'}$ independent unless $n=n'$. A similar state-transition model was used, e.g., in [29].

The target emits an acoustic or radio signal with a known, constant transmit power A. The (scalar) measurement $z_{n,k}$ obtained by sensor k at time n is modeled as

$$z_{n,k} = m_k(\mathbf{x}_n) + v_{n,k}\,, \quad \text{with } m_k(\mathbf{x}_n) = \frac{A}{\|\boldsymbol{\rho}(\mathbf{x}_n) - \boldsymbol{\xi}_k\|^\kappa}\,, \tag{39}$$

where $\boldsymbol{\xi}_k$ is the location of sensor k, κ is the path loss exponent, and $v_{n,k}$ is measurement noise. We assume that $v_{n,k}$ is independent of $\mathbf{x}_{n'}$ for all n', and that $v_{n,k}$ and $v_{n',k'}$

are independent unless $(n,k)=(n',k')$. Because the $v_{n,k}$ are independent across the sensors, the factorization (3) holds and the JLF is obtained as

$$f(\mathbf{z}_n|\mathbf{x}_n)=\prod_{k=1}^{K} f(z_{n,k}|\mathbf{x}_n)=\prod_{k=1}^{K} f(v_{n,k})\big|_{v_{n,k}=z_{n,k}-m_k(\mathbf{x}_n)}.$$

B. Simulation Setting

The covariance matrix of the driving noise $\mathbf{u}_n$ in (38) is $\mathbf{C}_u=\mathrm{diag}\{0.1,0.1,0.01,0.01\}$. The transmit power of the target is $A=10$. The state prior is $f(\mathbf{x}_0)=\mathcal{N}(\mathbf{x}_0;\boldsymbol{\mu}_0,\mathbf{C}_0)$, with $\boldsymbol{\mu}_0=(4\ 4\ 1\ \pi/4)^\top$ and $\mathbf{C}_0=\mathbf{I}_4$. The network consists of $K=25$ sensors that are deployed on a jittered grid within a region of size $40\,\mathrm{m}\times 40\,\mathrm{m}$. Each sensor communicates with other sensors within a range of $18\,\mathrm{m}$. Unless noted otherwise, the measurement noise pdf is a two-component Gaussian mixture $f(v_{n,k})=(1-\epsilon)\mathcal{N}(v_{n,k};0,\sigma_1^2)+\epsilon\mathcal{N}(v_{n,k};0,\sigma_2^2)$ with $\sigma_1^2=5\cdot 10^{-6}$, $\sigma_2^2=5\cdot 10^{-5}$, and $\epsilon=0.11$; note that the resulting local likelihood function and JLF do not belong to the exponential family. The path loss exponent is $\kappa=2$. These values of σ_1^2, σ_2^2, ϵ, and κ yield peaky likelihood functions, which is a case highlighting the performance gains of proposal density adaptation.

For LC, we use a basis expansion approximation either of the local log-likelihood functions $\log f(z_{n,k}|\mathbf{x}_n)$ as described in Section IV-B—this is used unless noted otherwise—or of the functions $m_k(\mathbf{x}_n)$ as described at the end of Section IV-C. We employ either a polynomial basis—used unless noted otherwise—or a Fourier basis (see Section V-B). For the polynomial basis, unless noted otherwise, the degree of the polynomial is $R_p=6$, corresponding to $R=\binom{R_p+\tilde{M}}{R_p}=28$ basis functions (here, $\tilde{M}=2$, because $m_k(\mathbf{x}_n)$ in (39) depends only on the target location $\boldsymbol{\rho}(\mathbf{x}_n)$). For the Fourier basis, $\tilde{R}=2$, corresponding to $R=(2\tilde{R}+1)^{\tilde{M}}=25$ basis functions.

We compare the performance of the proposed LC-based DPF algorithm using distributed proposal density adaptation—i.e., Algorithms 1–3 (abbreviated LC-DPF)—with that of the following alternative methods: a distributed unscented Kalman filter (DUKF), the state-of-the-art DPF proposed in [3] (DPF-1), the state-of-the-art DPF proposed in [5] (DPF-2), and a centralized particle filter (CPF). In Step 1 of Algorithm 3, the update step of an unscented Kalman filter is used. The measurement update step in DUKF is performed in a distributed fashion and is analogous to our proposal density adaptation scheme; however, the prediction step is performed using sigma points instead of particles. The proposal density adaptation of DPF-1 uses all the sensor measurements [3] whereas that of DPF-2 is only based on the local measurement of each sensor [5]. CPF performs the same algorithmic steps as LC-DPF except that (i) it collects all the sensor measurements at a fusion center, and thus does not need to use the LC to compute the JLF, and (ii) it employs a centralized implementation of the proposal density adaptation of LC-DPF, without the use of consensus algorithms. We note that CPF is similar to the particle filter proposed in [13], however with the difference that it uses a Gaussian rather than Gaussian mixture approximation of the posterior pdf. Unless stated otherwise, the number of particles at each sensor (for LC-DPF, DPF-1, and DPF-2) and at the fusion center (for CPF) is $J=200$. In DPF-1 [3], the rejection probability used for proposal density adaptation is $\beta_k=0.02$, and the oversampling factor is $L=10$. The consensus algorithms employed in LC-DPF, DPF-1, and DPF-2 to calculate sums over all sensors use Metropolis weights [10]. Unless noted otherwise, each consensus algorithm performs $I=10$ iterations.

As a performance measure, we consider the estimated n-dependent root-mean-square error of the target location estimate, denoted RMSE_n, which is calculated as the square root of the average of $\|\hat{\boldsymbol{\rho}}_{n,k}-\boldsymbol{\rho}(\mathbf{x}_n)\|^2$ over all sensors $k=1,\ldots,25$ and over 1000 simulation runs. Here, $\hat{\boldsymbol{\rho}}_{n,k}$ denotes the estimate at sensor k of the target location $\boldsymbol{\rho}(\mathbf{x}_n)$. Furthermore, we calculate the *(time-)averaged* RMSE (ARMSE) as the square root of the average of RMSE_n^2 over $n=21,\ldots,200$. Note that we disregard the first 20 time instants in order to portray the steady-state performance. We also assess the error variation across the sensors k by the standard deviation σ_{ARMSE} of a k-dependent time-averaged RMSE that is calculated as the square root of the average of $\|\hat{\boldsymbol{\rho}}_{n,k}-\boldsymbol{\rho}(\mathbf{x}_n)\|^2$ over $n=21,\ldots,200$ and all 1000 simulation runs. In the calculation of RMSE_n, ARMSE, and σ_{ARMSE}, simulation runs producing "lost tracks" are excluded. These are simulation runs where the estimation error at the final time $n=200$ exceeds $5\,\mathrm{m}$ (i.e., half the average distance between two neighboring sensors). Excluding lost tracks is important because they tend to skew the RMSE results beyond interpretability [3], [33]. However, for a complete picture of DPF performance, we also report the percentage of lost tracks (PLT) in addition to the RMSE results.

As a measure of the communication requirements, we consider the number C of real values broadcast at each time instant n by sensor k to its neighbors; this is equal to the number of consensus algorithms executed in parallel times the number of consensus iterations, I. For LC-DPF, $C=C_{\mathrm{LC}}+C_{\mathrm{PA}}=\big[R+(M^2+3M)/2\big]I$ (cf. (27) and (37)). For DPF-1, the number of (average) consensus algorithms equals the number of particles, i.e., $J=200$; for DPF-2, it equals the number of entries of the posterior mean vector and in the upper triangular part of the posterior covariance matrix, i.e., $(M^2+3M)/2$. In addition, DPF-1 requires transmission of $2\tilde{M}I_m$ real values for proposal density adaptation and of $2JI_m$ real values for the max and min consensus algorithms that ensure identical particle weights across all sensors. Here, I_m is the number of iterations of the max or min consensus algorithms, which equals the diameter of the communication graph, i.e., $I_m=4$. For CPF, we report the average communication requirements per sensor (note that the communication requirements of CPF vary from sensor to sensor and also depend on the network topology). We assume that CPF uses multihop transmission of measurements from each sensor to the fusion center, which is located in one of the corners of the network.

C. Simulation Results

Table II summarizes the estimation performance (ARMSE, σ_{ARMSE}, and PLT) and the communication requirements (C)

	ARMSE [m]	σ_{ARMSE} [m]	PLT [%]	C
LC-DPF (Fourier basis)	0.1569	0.0259	0.2	390
LC-DPF (polyn. basis)	0.1272	0.0227	0.4	420
DUKF	1.7484	0.0070	90.7	140
DPF-1 [3]	0.1130	0	0.2	3616
DPF-2 [5]	0.4245	0.0084	43.8	140
CPF	0.0651	–	0.2	2.36

TABLE II
ESTIMATION PERFORMANCE (ARMSE, σ_{ARMSE}, PLT) AND COMMUNICATION REQUIREMENTS (C) OF LC-DPF (PROPOSED), DUKF, DPF-1, DPF-2, AND CPF.

of the proposed LC-DPF—using the Fourier or polynomial basis—and of DUKF, DPF-1, DPF-2, and CPF. It is seen that the ARMSE of LC-DPF is higher than that of CPF, close to that of DPF-1, and much lower than that of DPF-2 and DUKF. The PLTs of LC-DPF, DPF-1, and CPF are 0.4% or lower, whereas those of DPF-2 and DUKF are much higher (43.8% for DPF-2 and 90.7% for DUKF). For DPF-1, $\sigma_{\text{ARMSE}} = 0$ because identical estimates at each sensor are enforced by max and min consensus algorithms. Furthermore, σ_{ARMSE} for LC-DPF is higher than for DPF-2 and DUKF (however, this result should be interpreted in the light of the high PLTs of DPF-2 and DUKF). The communication requirements of LC-DPF are higher than those of DPF-2 and DUKF but much lower than those of DPF-1. Note, however, that the communication requirements of DPF-1 could be reduced by using fewer particles, whereas those of the other methods do not depend on the number of particles. The communication requirements of CPF are the lowest of all methods. This is due to the low dimension of the measurements in our simulation setting and may be very different when high-dimensional measurements have to be transmitted to the fusion center. Furthermore, we do not consider the overhead required for transmitting the local likelihood functions, the identities of the sensors producing the measurements, and data associated with the routing protocol, and we assume that there is no need for the fusion center to send information about the posterior back to the sensors. Note that even though the distributed methods require more communication, they may be preferable over CPF because they are more robust (no possibility of fusion center failure), they do not require a routing protocol, and each sensor obtains an approximation of the global posterior.

The following conclusions can be drawn from Table II:

1) For LC-DPF, the Fourier and polynomial basis approximations lead to similar estimation performance. This is due to the calculation of the expansion coefficients using least squares fitting at the particles $\mathbf{x}_{n,k}^{(j)}$ (see Section V-A). The polynomial basis leads to a slightly better ARMSE performance at the cost of somewhat higher communication requirements. In the subsequent simulations, we use the polynomial basis.
2) The ARMSE and PLT of DPF-2 are much higher than those of LC-DPF and DPF-1. This is because DPF-2 uses only locally adapted proposal densities, unlike the other DPFs which use all sensor measurements for proposal density adaptation. Local proposal density adaptation is seen to be insufficient for the simulated setting.
3) DUKF performs worst of all the simulated methods. This demonstrates the advantage of particle filters in nonlinear/non-Gaussian settings. Because of its high PLT and ARMSE, DUKF is not further considered in the following.

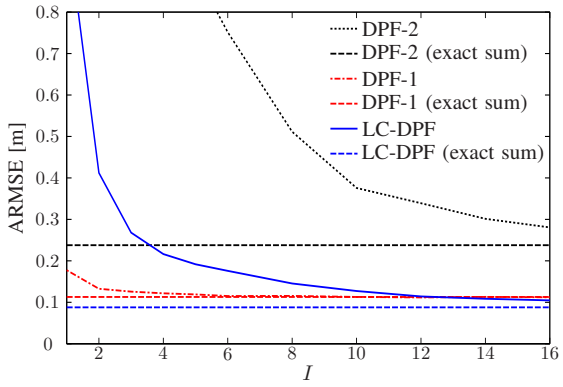

Fig. 1. ARMSE of LC-DPF, DPF-1, and DPF-2 and of the respective "exact sum calculation" variants versus the number I of (average) consensus iterations.

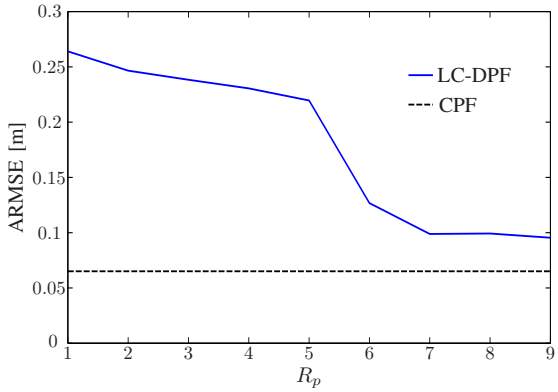

Fig. 2. ARMSE of LC-DPF versus the degree R_p of polynomial approximation. The ARMSE of CPF is provided as a reference.

Fig. 1 shows the ARMSE of LC-DPF, DPF-1, and DPF-2 versus the number I of consensus iterations. As a performance benchmark, the figure also shows the results for impractical LC-DPF, DPF-1, and DPF-2 variants in which the approximate sum calculations performed by the consensus algorithms are replaced by direct, exact sum calculations; note that this corresponds to an infinite number of consensus iterations. It is seen that as I increases, the performance of all DPFs approaches that of the respective "exact sum calculation" variant. While LC-DPF is outperformed by DPF-1 for I less than 12, it performs equally well as DPF-1 for $I \geq 12$, and it significantly outperforms DPF-2 for all I.

In Fig. 2, we show the ARMSE of LC-DPF versus the degree R_p of the polynomials used to approximate the local log-likelihood functions. For growing R_p, as can be expected, the ARMSE decreases (since the approximation of the JLF is improved) and gets closer to that of CPF. Note, however, that

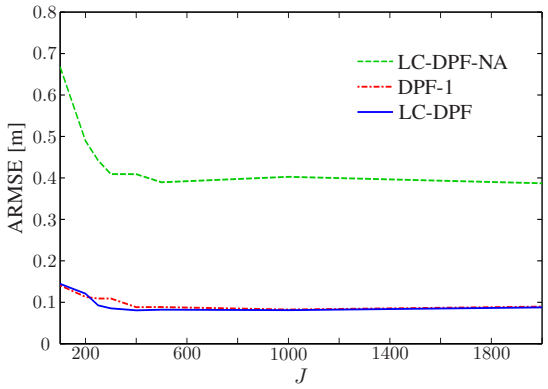

Fig. 3. ARMSE of LC-DPF, LC-DPF-NA, and DPF-1 versus the number J of particles.

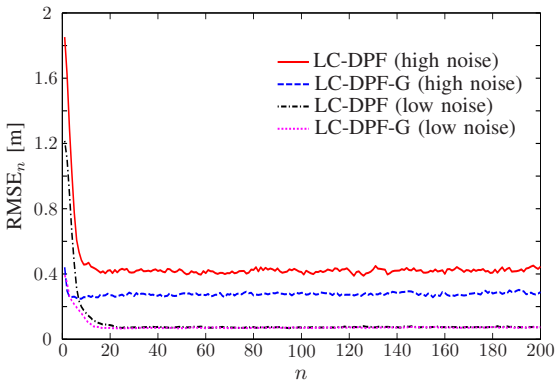

Fig. 5. RMSE$_n$ versus time n for LC-DPF and LC-DPF-G in the case of Gaussian measurement noises.

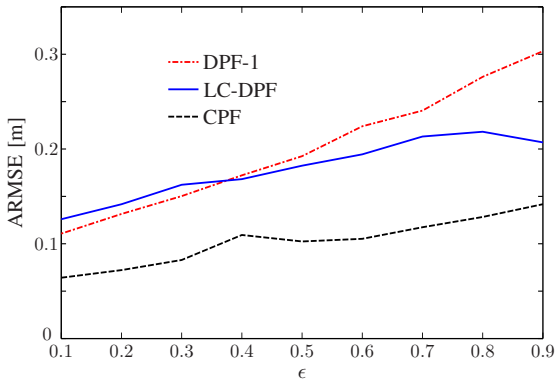

Fig. 4. ARMSE of LC-DPF, DPF-1, and CPF versus the mixing parameter ϵ of the Gaussian mixture measurement noise pdf.

the communication requirements of LC-DPF increase with growing R_p.

The dependence of the ARMSE on the number J of particles is depicted in Fig. 3 for LC-DPF, LC-DPF without proposal density adaptation (abbreviated as LC-DPF-NA), and DPF-1. The ARMSE of LC-DPF-NA is seen to be significantly higher than that of LC-DPF, even when J is large. This demonstrates the benefit of proposal density adaptation. The ARMSEs of LC-DPF and DPF-1 are almost equal and approximately constant for J above 400. However, it should be noted that the communication requirements of DPF-1 increase with increasing J, whereas those of LC-DPF are independent of J.

Fig. 4 depicts the dependence of the ARMSE of LC-DPF, DPF-1, and CPF on the mixing parameter ϵ of the measurement noise pdf $f(v_{n,k}) = (1-\epsilon)\mathcal{N}(v_{n,k}; 0, \sigma_1^2) + \epsilon\mathcal{N}(v_{n,k}; 0, \sigma_2^2)$. Since in our setting $\sigma_2^2 > \sigma_1^2$, by increasing ϵ, the overall variance of the measurement noise also increases. As may be expected, the ARMSE of all three algorithms increases with increasing ϵ. For $\epsilon > 0.4$, LC-DPF outperforms DPF-1.

Finally, a setting with additive Gaussian measurement noises is considered in Fig. 5. We show the time dependence of RMSE$_n$ for LC-DPF (which, as before, uses a basis expansion approximation of the local log-likelihood functions as described in Section IV-B) and for an LC-DPF version that exploits the additive Gaussian noise structure by using a basis expansion approximation of the functions $m_k(\mathbf{x}_n)$ as described at the end of Section IV-C. This LC-DPF version is abbreviated as LC-DPF-G. We consider two settings: a "low noise setting" with $\sigma_v^2 = 10^{-5}$ and (as before) $\mathbf{C}_u = \mathrm{diag}\{0.1, 0.1, 0.01, 0.01\}$, and a "high noise setting" with $\sigma_v^2 = 10^{-4}$ and $\mathbf{C}_u = \mathrm{diag}\{0.2, 0.2, 0.02, 0.02\}$. It is seen that the performance of LC-DPF and LC-DPF-G is very similar in the low noise setting whereas LC-DPF-G clearly outperforms LC-DPF in the high noise setting. This can be interpreted as follows. For higher noise variances, the particles are sampled in a larger state-space region and thus the least squares approximation is less accurate. In that case, exploiting the additive Gaussian noise structure by using the basis expansion approximation of Section IV-C yields a larger performance improvement.

IX. Conclusion

We presented a distributed particle filter algorithm with distributed proposal density adaptation for decentralized wireless sensor networks. At each sensor, a local particle filter computes a global state estimate that takes into account the past and present measurements of all sensors. This global estimation mode is based on a distributed approximate calculation of the joint (all-sensors) likelihood function via the likelihood consensus scheme. We extended a previously proposed version of the likelihood consensus, which was limited to local likelihood functions belonging to the exponential family, to arbitrary local likelihood functions. Furthermore, for an improvement of performance or a reduction of the required number of particles, we proposed a consensus-based distributed method for adapting the proposal densities used by the local particle filters. The resulting proposal densities are again global in that they take into account the measurements of all sensors.

The proposed distributed particle filter uses only local communication between neighboring sensors and does not require any routing protocols; moreover, no particles, local state estimates, or measurements are communicated between the sensors. The communication requirements are independent of the number of particles and of the dimension of the measurements, but they grow with the state dimension.

We applied the proposed distributed particle filter to a target tracking problem and demonstrated experimentally that its performance is close to that of a centralized particle filter with centralized proposal density adaptation. We also observed that our method achieves a better estimation performance than the distributed particle filter presented in [5], at the cost of somewhat higher communication requirements. Furthermore, for a low number of consensus iterations, our method is outperformed by the distributed particle filter presented in [3]; on the other hand, the communication requirements of our method are much lower than those of the method in [3]. Our simulation results also show that our proposal density adaptation scheme can yield large performance gains.

An extension of the likelihood consensus method to spatially correlated measurement noises was presented in [34]. This extension is limited to Gaussian measurement noises and does not include proposal density adaptation. An extension to statistically dependent non-Gaussian measurement noises and to distributed particle filters with proposal density adaptation remains a topic for future research.

REFERENCES

[1] O. Hlinka, F. Hlawatsch, and P. M. Djurić, "Distributed particle filtering in agent networks: A survey, classification, and comparison," *IEEE Signal Process. Mag.*, vol. 30, pp. 61–81, Jan. 2013.

[2] B. N. Oreshkin and M. J. Coates, "Asynchronous distributed particle filter via decentralized evaluation of Gaussian products," in *Proc. FUSION-10*, Edinburgh, UK, Jul. 2010.

[3] S. Farahmand, S. I. Roumeliotis, and G. B. Giannakis, "Set-membership constrained particle filter: Distributed adaptation for sensor networks," *IEEE Trans. Signal Process.*, vol. 59, pp. 4122–4138, Sep. 2011.

[4] D. Üstebay, M. Coates, and M. Rabbat, "Distributed auxiliary particle filters using selective gossip," in *Proc. IEEE ICASSP-11*, Prague, Czech Republic, pp. 3296–3299, May 2011.

[5] A. Mohammadi and A. Asif, "Consensus-based distributed unscented particle filter," in *Proc. IEEE SSP-11*, Nice, France, pp. 237–240, Jun. 2011.

[6] O. Hlinka, O. Slučiak, F. Hlawatsch, P. M. Djurić, and M. Rupp, "Likelihood consensus and its application to distributed particle filtering," *IEEE Trans. Signal Process.*, vol. 60, pp. 4334–4349, Aug. 2012.

[7] O. Hlinka, F. Hlawatsch, and P. M. Djurić, "Likelihood consensus-based distributed particle filtering with distributed proposal density adaptation," in *Proc. IEEE ICASSP-12*, Kyoto, Japan, pp. 3869–3872, Mar. 2012.

[8] C. J. Bordin and M. G. S. Bruno, "Consensus-based distributed particle filtering algorithms for cooperative blind equalization in receiver networks," in *Proc. IEEE ICASSP-11*, Prague, Czech Republic, pp. 3968–3971, May 2011.

[9] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Contr. Letters*, vol. 53, no. 1, pp. 65–78, 2004.

[10] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. IEEE/ACM IPSN-05*, Los Angeles, CA, pp. 63–70, Apr. 2005.

[11] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, pp. 215–233, Jan. 2007.

[12] M. Šimandl and O. Straka, "Sampling densities of particle filter: A survey and comparison," in *Proc. IEEE ACC-07*, New York, NY, pp. 4437–4442, Jul. 2007.

[13] R. van der Merwe and E. Wan, "Gaussian mixture sigma-point particle filters for sequential probabilistic inference in dynamic state-space models," in *Proc. IEEE ICASSP-03*, Hong Kong, P. R. China, pp. 701–704, Apr. 2003.

[14] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ: Prentice Hall, 1993.

[15] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, pp. 174–188, Feb. 2002.

[16] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proc. F Radar Signal Process.*, vol. 140, pp. 107–113, Apr. 1993.

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

[18] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Míguez, "Particle filtering," *IEEE Signal Process. Mag.*, vol. 20, pp. 19–38, Sep. 2003.

[19] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *J. Am. Stat. Assoc.*, vol. 94, pp. 590–599, Jun. 1999.

[20] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.

[21] Å. Björck, *Numerical Methods for Least Squares Problems*. Philadelphia, PA: SIAM, 1996.

[22] L. Georgopoulos and M. Hasler, "Nonlinear average consensus," in *Proc. NOLTA-09*, Sapporo, Japan, pp. 10–14, Oct. 2009.

[23] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, pp. 1847–1864, Nov. 2010.

[24] D. Mosk-Aoyama and D. Shah, "Fast distributed algorithms for computing separable functions," *IEEE Trans. Inf. Theory*, vol. 54, pp. 2997–3007, Jul. 2008.

[25] O. Cappé, S. J. Godsill, and E. Moulines, "An overview of existing methods and recent advances in sequential Monte Carlo," *Proc. IEEE*, vol. 95, pp. 899–924, May 2007.

[26] M. J. F. Gales and S. S. Airey, "Product of Gaussians for speech recognition," *Computer Speech & Language*, vol. 20, pp. 22–40, Jan. 2006.

[27] H. Tanizaki, *Nonlinear Filters: Estimation and Applications*. Berlin, Germany: Springer, 1996.

[28] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proc. IEEE*, vol. 92, pp. 401–422, Mar. 2004.

[29] R. van der Merwe, *Sigma-point Kalman filters for probabilistic inference in dynamic state-space models*. PhD thesis, OGI School of Science and Eng., Oregon Health and Science University, Hillsboro, OR, Apr. 2004.

[30] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Autom. Contr.*, vol. 54, pp. 1254–1269, Jun. 2009.

[31] K. Ito and K. Xiong, "Gaussian filters for nonlinear filtering problems," *IEEE Trans. Autom. Contr.*, vol. 45, pp. 910–927, May 2000.

[32] E. A. Wan and R. van der Merwe, "The unscented Kalman filter," in *Kalman Filtering and Neural Networks* (S. Haykin, ed.), pp. 221–280, New York, NY: Wiley, 2001.

[33] P. Willett, R. Niu, and Y. Bar-Shalom, "Integration of Bayes detection with target tracking," *IEEE Trans. Signal Process.*, vol. 49, pp. 17–29, Jan. 2001.

[34] O. Hlinka and F. Hlawatsch, "Distributed particle filtering in the presence of mutually correlated sensor noises," in *Proc. IEEE ICASSP-13*, Vancouver, BC, Canada, pp. 6269–6273, May 2013.

Ondrej Hlinka (S'09–M'13) received the Ing. (M.Eng.) degree in electrical engineering from the Slovak University of Technology, Bratislava, Slovakia in 2008 and the Dr. techn. (Ph.D.) degree in electrical engineering from the Vienna University of Technology, Vienna, Austria in 2012. Since 2008, he has been a Research Assistant with the Institute of Telecommunications, Vienna University of Technology. His research interests include distributed signal processing for agent networks, statistical signal processing, and sequential Monte Carlo methods.

Franz Hlawatsch (S'85–M'88–SM'00–F'12) received the Diplom-Ingenieur, Dr. techn., and Univ.-Dozent (habilitation) degrees in electrical engineering/signal processing from Vienna University of Technology, Vienna, Austria in 1983, 1988, and 1996, respectively. Since 1983, he has been with the Institute of Telecommunications, Vienna University of Technology, where he is currently an Associate Professor. During 1991–1992, as a recipient of an Erwin Schrödinger Fellowship, he spent a sabbatical year with the Department of Electrical Engineering, University of Rhode Island, Kingston, RI, USA. In 1999, 2000, and 2001, he held one-month Visiting Professor positions with INP/ENSEEIHT, Toulouse, France and IRCCyN, Nantes, France. He (co)authored a book, three review papers that appeared in the IEEE SIGNAL PROCESSING MAGAZINE, about 200 refereed scientific papers and book chapters, and three patents. He coedited three books. His research interests include signal processing for sensor networks and wireless communications, statistical signal processing, and compressive signal processing.

Prof. Hlawatsch was Technical Program Co-Chair of EUSIPCO 2004 and served on the technical committees of numerous IEEE conferences. He was an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 2003 to 2007 and for the IEEE TRANSACTIONS ON INFORMATION THEORY from 2008 to 2011. From 2004 to 2009, he was a member of the IEEE SPCOM Technical Committee. He is coauthor of papers that won an IEEE Signal Processing Society Young Author Best Paper Award and a Best Student Paper Award at IEEE ICASSP 2011.

Petar M. Djurić (S'86–M'90–SM'99–F'06) received the B.S. and M.S. degrees in electrical engineering from the University of Belgrade, Belgrade, Yugoslavia, in 1981 and 1986, respectively, and the Ph.D. degree in electrical engineering from the University of Rhode Island, Kingston, RI, USA, in 1990. He is currently a Professor with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY, USA, where he joined in 1990. From 1981 to 1986, he was a Research Associate with the Vinča Institute of Nuclear Sciences, Belgrade. His research has been in the area of signal and information processing with primary interests in the theory of signal modeling, detection, and estimation; Monte Carlo-based methods; distributed signal processing; and applications of the theory in a wide range of disciplines. He has been invited to lecture at many universities in the United States and overseas. Prof. Djurić was a recipient of the IEEE Signal Processing Magazine Best Paper Award in 2007 and the EURASIP Technical Achievement Award in 2012. In 2008, he was the Chair of Excellence of Universidad Carlos III de Madrid-Banco de Santander. From 2008 to 2009, he was a Distinguished Lecturer of the IEEE Signal Processing Society. He has been on numerous committees of the IEEE Signal Processing Society and of many professional conferences and workshops.