

Likelihood Consensus and Its Application to Distributed Particle Filtering

Ondrej Hlinka, Student Member, IEEE, Ondrej Slučiak, Student Member, IEEE, Franz Hlawatsch, Fellow, IEEE, Petar M. Djurić, Fellow, IEEE, and Markus Rupp, Senior Member, IEEE

Abstract—We consider distributed state estimation in a wireless sensor network without a fusion center. Each sensor performs a global estimation task—based on the past and current measurements of *all* sensors—using only local processing and local communications with its neighbors. In this estimation task, the joint (all-sensors) likelihood function (JLF) plays a central role as it epitomizes the measurements of all sensors. We propose a distributed method for computing, at each sensor, an approximation of the JLF by means of consensus algorithms. This “likelihood consensus” method is applicable if the local likelihood functions of the various sensors (viewed as conditional probability density functions of the local measurements) belong to the exponential family of distributions. We then use the likelihood consensus method to implement a distributed particle filter and a distributed Gaussian particle filter. Each sensor runs a local particle filter, or a local Gaussian particle filter, that computes a global state estimate. The weight update in each local (Gaussian) particle filter employs the JLF, which is obtained through the likelihood consensus scheme. For the distributed Gaussian particle filter, the number of particles can be significantly reduced by means of an additional consensus scheme. Simulation results are presented to assess the performance of the proposed distributed particle filters for a multiple target tracking problem.

Index Terms—Wireless sensor network, distributed state estimation, sequential Bayesian estimation, consensus algorithm, distributed particle filter, distributed Gaussian particle filter, target tracking.

I. INTRODUCTION

Distributed estimation in wireless sensor networks has received significant attention recently (e.g., [1]–[3]). Applications include machine and structural health monitoring, pollution source localization, habitat monitoring, and target tracking. Typically, a wireless sensor network is composed of battery-powered sensing/processing nodes—briefly called “sensors” hereafter—which possess limited sensing, computation, and communication capabilities.

Copyright (©) 2012 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Ondrej Hlinka, Ondrej Slučiak, Franz Hlawatsch, and Markus Rupp are with the Institute of Telecommunications, Vienna University of Technology, A-1040 Vienna, Austria (e-mail: {ondrej.hlinka, ondrej.sluciak, franz.hlawatsch, markus.rupp}@nt.tuwien.ac.at). Petar M. Djurić is with the Department of Electrical and Computer Engineering, Stony Brook University, Stony Brook, NY, USA (e-mail: djuric@ece.sunysb.edu). This work was supported by the FWF under Awards S10603 and S10611 within the National Research Network SISE, by the NSF under Award CCF-1018323, and by the ONR under Award N00014-09-1-1154. Parts of this work were previously presented at the 44th Asilomar Conf. Sig., Syst., Comp., Pacific Grove, CA, Nov. 2010 and at IEEE ICASSP 2011, Prague, Czech Republic, May 2011. The simulation source files are available online at <http://ieeexplore.ieee.org>.

Centralized estimation techniques transmit sensor data to a possibly distant fusion center [1]. This may require energy-intensive communications over large distances or complex multi-hop routing protocols, which results in poor scalability. Centralized techniques are also less robust, and less suitable if the estimation results have to be available at the sensors (e.g., in sensor-actuator networks [4]). Furthermore, the fusion center must be aware of the measurement models and, possibly, additional parameters of all sensors. By contrast, decentralized estimation techniques without a fusion center use in-network processing and neighbor-to-neighbor communications to achieve low energy consumption as well as high robustness and scalability. The sensors do not require knowledge of the network topology, and no routing protocols are needed.

There are two basic categories of decentralized estimation techniques. In the first, information is transmitted in a sequential manner from sensor to sensor [5]–[7]. In the second, each sensor diffuses its local information in an iterative process using broadcasts to a set of neighboring sensors (e.g., [8]). This second category is more robust but involves an increased communication overhead. It includes consensus-based estimation techniques, which use distributed algorithms for reaching a consensus (on a sum, average, maximum, etc.) in the network [9], [10]. Examples are gossip algorithms [10], consensus algorithms [11], and combined approaches [12].

In this paper, we consider a decentralized wireless sensor network architecture without a fusion center and use consensus algorithms to perform a *global* estimation task through *local* processing and communications, in a way such that the final global estimate is available locally at each sensor. (“Global” estimation means that the measurements of *all* sensors are processed by each sensor.) This can be based on the joint (all-sensors) likelihood function, abbreviated JLF, which epitomizes the measurements of all sensors. The JLF is then required to be known by all sensors. For example, a global particle filter (PF) [13]–[15] that processes all sensor measurements relies on the pointwise evaluation of the JLF to perform its weight update.

The main contribution of this paper is a distributed method for calculating the JLF or an approximation of the JLF at each sensor. Generalizing our previous work in [16], [17], this method is suited to sensors with local likelihood functions that are members of the exponential family of distributions. A consensus algorithm—calculating sums—is used for a decentralized, iterative computation of a sufficient statistic that

describes the (approximate) JLF as a function of the state to be estimated. Consequently, we refer to our method as *likelihood consensus* (LC). The LC scheme requires communications only between neighboring sensors and operates without routing protocols. We furthermore propose an application of our LC method in a distributed PF scheme and in a distributed Gaussian PF scheme. Each sensor runs a local PF (or a local Gaussian PF [18]) that computes a global state estimate incorporating all sensor measurements. At any given PF recursion, each local (Gaussian) PF draws a set of particles and updates their weights based on an evaluation of the JLF at these particles. For the distributed Gaussian PF, the number of particles employed by each local Gaussian PF can be significantly reduced by means of a second consensus scheme.

Alternative consensus-based distributed PF schemes have been proposed in [19]–[24]. The method described in [19] uses one consensus algorithm per particle to calculate products of local particle weights. To reduce the communication requirements, the number of particles is kept small by an adaptation of the proposal distribution. Nevertheless, the number of consensus algorithms required can be significantly higher than in our approach. Furthermore, the random number generators of the individual sensors must be synchronized. On the other hand, since no approximation of the JLF is required, the performance can be closer to that of a centralized PF. The consensus-based distributed PFs proposed in [20] and [21] rely on local PFs that update their weights using only the *local likelihood functions* instead of the JLF. Gaussian or Gaussian mixture approximations of local posteriors are then computed, and a consensus algorithm is used to fuse these approximations. However, this fusion rule is suboptimal and leads to a performance loss. In [22], a novel gossiping approach implementing an approximation of the optimal fusion rule is employed to construct a Gaussian approximation of the global posterior. However, again only local likelihood functions are used by the local PFs, and the estimation performance is worse than in our approach. In [23], a distributed unscented PF is proposed that uses local measurements for proposal adaptation and an optimal consensus-based fusion rule to compute global estimates from local estimates. The distributed PF proposed in [24] operates across clusters of sensors and uses a modified maximum consensus algorithm to aggregate the local posterior distributions from all clusters.

Distributed PFs that do not rely on consensus algorithms have been presented in [25]–[27]. In these methods, a path through the sensor network is adaptively determined by means of a decentralized sensor scheduling algorithm. Parametric representations of partial likelihood functions or of partial posteriors are transmitted along this path. The last sensor in the path obtains the complete global information and is thus able to compute a global estimate. In general, these methods are not as robust to sensor failure as the consensus-based methods. However, in certain applications, their communication requirements may be much lower.

This paper is organized as follows. In Section II, we describe the system model and review sequential Bayesian estimation. To prepare the ground for the LC method, an

approximation of the exponential class of distributions is discussed in Section III. The LC method is presented in Section IV. In Section V, we consider the special case of additive Gaussian measurement noise. The application of LC to distributed particle filtering and distributed Gaussian particle filtering is considered in Section VI and VII, respectively. Finally, in Section VIII, the proposed distributed PFs are applied to multiple target tracking, and simulation results are presented.

II. SYSTEM MODEL AND SEQUENTIAL BAYESIAN ESTIMATION

We consider a wireless sensor network consisting of K sensors. At a given discrete time n , each sensor estimates a global M -dimensional state $\mathbf{x}_n = (x_{n,1} \cdots x_{n,M})^\top \in \mathbb{R}^M$ based on all sensor measurements. The state evolves according to the state-transition probability density function (pdf) $f(\mathbf{x}_n|\mathbf{x}_{n-1})$. At time n , the k th sensor ($k \in \{1, \dots, K\}$) acquires an $N_{n,k}$ -dimensional measurement $\mathbf{z}_{n,k} \in \mathbb{R}^{N_{n,k}}$. The relationship between $\mathbf{z}_{n,k}$ and \mathbf{x}_n is described by the *local likelihood function*¹ $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$, and the relationship between the all-sensors measurement vector $\mathbf{z}_n \triangleq (\mathbf{z}_{n,1}^\top \cdots \mathbf{z}_{n,K}^\top)^\top$ and \mathbf{x}_n is described by the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$. All $\mathbf{z}_{n,k}$ are assumed conditionally independent given \mathbf{x}_n , so that the JLF is the product of all local likelihood functions, i.e.,

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

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 .

In the sequel, we will use the following assumptions. First, the current state \mathbf{x}_n is conditionally independent of all past measurements, $\mathbf{z}_{1:n-1}$, given the previous state \mathbf{x}_{n-1} , i.e.,

$$f(\mathbf{x}_n|\mathbf{x}_{n-1}, \mathbf{z}_{1:n-1}) = f(\mathbf{x}_n|\mathbf{x}_{n-1}). \quad (2)$$

Second, the current measurement \mathbf{z}_n is conditionally independent of all past measurements, $\mathbf{z}_{1:n-1}$, given the current state \mathbf{x}_n , i.e.,

$$f(\mathbf{z}_n|\mathbf{x}_n, \mathbf{z}_{1:n-1}) = f(\mathbf{z}_n|\mathbf{x}_n). \quad (3)$$

Finally, sensor k knows the state-transition pdf $f(\mathbf{x}_n|\mathbf{x}_{n-1})$ and its own local likelihood function $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ as well as the pdf $f(\mathbf{x}_0)$ of the initial state \mathbf{x}_0 , but it does not know the local likelihood functions of the other sensors, i.e., $f(\mathbf{z}_{n,k'}|\mathbf{x}_n)$ for $k' \neq k$.

We briefly review sequential Bayesian state estimation [28], which will be considered as a motivating application of the LC method. At time n , each sensor estimates the current state \mathbf{x}_n from the measurements of all sensors up to time n , $\mathbf{z}_{1:n}$. For this task, we will use the minimum mean-square error (MMSE) estimator [29],

$$\hat{\mathbf{x}}_n^{\text{MMSE}} \triangleq \text{E}\{\mathbf{x}_n|\mathbf{z}_{1:n}\} = \int \mathbf{x}_n f(\mathbf{x}_n|\mathbf{z}_{1:n}) d\mathbf{x}_n, \quad (4)$$

¹The notation $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ suggests that \mathbf{x}_n is a random vector. However, for the LC method to be presented in Section IV, \mathbf{x}_n is also allowed to be deterministic, in which case the notation $f(\mathbf{z}_{n,k}; \mathbf{x}_n)$ would be more appropriate.

which is implemented at each sensor. Here, a major problem—even in a centralized scenario—is to calculate the posterior pdf $f(\mathbf{x}_n|\mathbf{z}_{1:n})$. Using (2) and (3), the current posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ can be obtained sequentially from the previous posterior $f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1})$ and the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ by means of the following temporal recursion [28]:

$$f(\mathbf{x}_n|\mathbf{z}_{1:n}) = \frac{f(\mathbf{z}_n|\mathbf{x}_n) \int f(\mathbf{x}_n|\mathbf{x}_{n-1}) f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1}) d\mathbf{x}_{n-1}}{f(\mathbf{z}_n|\mathbf{z}_{1:n-1})} \quad (5)$$

However, for nonlinear/non-Gaussian cases, the computational complexity of sequential MMSE state estimation as given by (4) and (5) is typically prohibitive. A computationally feasible approximation is provided by the PF [14], [15], [28]. In a PF, the (non-Gaussian) posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ is represented by a set of samples (or particles) $\mathbf{x}_n^{(j)}$, $j = 1, \dots, J$ and corresponding weights $w_n^{(j)}$.

As can be seen from (4) and (5), obtaining the global estimate $\hat{\mathbf{x}}_n^{\text{MMSE}}$ at each sensor presupposes that each sensor knows the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ as a function of the state \mathbf{x}_n (\mathbf{z}_n is observed and thus fixed, and $f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1})$ used in (5) was calculated by each sensor at the previous time $n-1$). In particular, a PF approximation of $\hat{\mathbf{x}}_n^{\text{MMSE}}$ relies on the pointwise evaluation of the JLF at the particles $\mathbf{x}_n^{(j)}$ —i.e., on the evaluation of $f(\mathbf{z}_n|\mathbf{x}_n^{(j)})$ —to obtain the weights $w_n^{(j)}$. Since each sensor knows only its local likelihood function $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$, we need a distributed method for calculating the JLF at each sensor. Such a method is proposed in Section IV.

It is important to note that, although we consider distributed sequential Bayesian estimation and distributed particle filtering as a motivating application, the proposed method can also be used for other distributed statistical inference tasks that require the pointwise evaluation of the JLF at the individual sensors.

III. APPROXIMATION OF THE JOINT LIKELIHOOD FUNCTION

The LC method can always be used if the local likelihood functions (viewed as conditional pdfs of the local measurements) belong to the exponential family of distributions. Typically, it requires an approximation of the local likelihood functions, and consequently of the JLF, which is discussed in the following. In Section IV-B, we will consider a class of JLFs for which an approximation is not needed.

A. Exponential Family

In this paper, except in Section IV-B, we assume that the local likelihood function of each sensor (viewed as the conditional pdf of $\mathbf{z}_{n,k}$) belongs to the exponential family of distributions [30], i.e.,

$$f(\mathbf{z}_{n,k}|\mathbf{x}_n) = c_{n,k}(\mathbf{z}_{n,k}) \exp(\mathbf{a}_{n,k}^\top(\mathbf{x}_n) \mathbf{b}_{n,k}(\mathbf{z}_{n,k}) - d_{n,k}(\mathbf{x}_n)), \quad k = 1, \dots, K, \quad (6)$$

with some 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 $q \in \mathbb{N}$. We furthermore assume that sensor k knows its own functions $c_{n,k}(\cdot)$, $\mathbf{a}_{n,k}(\cdot)$, $\mathbf{b}_{n,k}(\cdot)$, and $d_{n,k}(\cdot)$, but not $c_{n,k'}(\cdot)$,

$\mathbf{a}_{n,k'}(\cdot)$, $\mathbf{b}_{n,k'}(\cdot)$, and $d_{n,k'}(\cdot)$ for $k' \neq k$. Using (1), the JLF is obtained as

$$\begin{aligned} f(\mathbf{z}_n|\mathbf{x}_n) &= \prod_{k=1}^K c_{n,k}(\mathbf{z}_{n,k}) \exp(\mathbf{a}_{n,k}^\top(\mathbf{x}_n) \mathbf{b}_{n,k}(\mathbf{z}_{n,k}) - d_{n,k}(\mathbf{x}_n)) \\ &= C_n(\mathbf{z}_n) \exp(S_n(\mathbf{z}_n, \mathbf{x}_n)), \end{aligned} \quad (7)$$

where

$$C_n(\mathbf{z}_n) \triangleq \prod_{k=1}^K c_{n,k}(\mathbf{z}_{n,k}) \quad (9)$$

and

$$S_n(\mathbf{z}_n, \mathbf{x}_n) \triangleq \sum_{k=1}^K [\mathbf{a}_{n,k}^\top(\mathbf{x}_n) \mathbf{b}_{n,k}(\mathbf{z}_{n,k}) - d_{n,k}(\mathbf{x}_n)]. \quad (10)$$

Note that the JLF (viewed as the conditional pdf of \mathbf{z}_n) also belongs to the exponential family. The normalization factor $C_n(\mathbf{z}_n)$ does not depend on the state \mathbf{x}_n and is hence typically irrelevant; we will ignore it for now and consider it only at the end of Section IV-A. Thus, according to (8), for global inference based on the all-sensors measurement vector \mathbf{z}_n , each sensor needs to know $S_n(\mathbf{z}_n, \mathbf{x}_n)$ as a function of \mathbf{x}_n , for the observed (fixed) \mathbf{z}_n . However, calculation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$ at a given sensor according to (10) presupposes that the sensor knows the measurements $\mathbf{z}_{n,k}$ and the functions $\mathbf{a}_{n,k}(\cdot)$, $\mathbf{b}_{n,k}(\cdot)$, and $d_{n,k}(\cdot)$ of *all* sensors, i.e., for all k . Transmitting the necessary information from each sensor to each other sensor may be infeasible.

B. Approximation of the Exponential Family

A powerful approach to diffusing local information through a wireless sensor network is given by iterative consensus algorithms, which require only communications with neighboring sensors and are robust to failing communication links and changing network topologies [11]. Unfortunately, a consensus-based distributed calculation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$ is not possible in general because the terms of the sum in (10) depend on the unknown state \mathbf{x}_n . Therefore, we will use an approximation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$ that involves a set of coefficients not dependent on \mathbf{x}_n . This approximation is induced by the following approximations of the functions $\mathbf{a}_{n,k}(\mathbf{x}_n)$ and $d_{n,k}(\mathbf{x}_n)$ in terms of given basis functions $\{\varphi_{n,r}(\mathbf{x}_n)\}_{r=1}^{R_a}$ and $\{\psi_{n,r}(\mathbf{x}_n)\}_{r=1}^{R_d}$, respectively:

$$\mathbf{a}_{n,k}(\mathbf{x}_n) \approx \tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n) \triangleq \sum_{r=1}^{R_a} \alpha_{n,k,r} \varphi_{n,r}(\mathbf{x}_n) \quad (11)$$

$$d_{n,k}(\mathbf{x}_n) \approx \tilde{d}_{n,k}(\mathbf{x}_n) \triangleq \sum_{r=1}^{R_d} \gamma_{n,k,r} \psi_{n,r}(\mathbf{x}_n). \quad (12)$$

Here, $\alpha_{n,k,r} \in \mathbb{R}^q$ and $\gamma_{n,k,r} \in \mathbb{R}$ are expansion coefficients that do not depend on \mathbf{x}_n . (For simplicity, the $\alpha_{n,k,r}$ are referred to as coefficients, even though they are vector-valued.) The basis functions $\varphi_{n,r}(\mathbf{x}_n)$ and $\psi_{n,r}(\mathbf{x}_n)$ do not depend on k , i.e., the same basis functions are used by all sensors. They are allowed to depend on n , even though time-independent basis functions may often be sufficient. We assume that sensor k knows the basis functions $\varphi_{n,r}(\mathbf{x}_n)$ and $\psi_{n,r}(\mathbf{x}_n)$, as well

as the coefficients $\alpha_{n,k,r}$ and $\gamma_{n,k,r}$ corresponding to its own functions $\mathbf{a}_{n,k}(\mathbf{x}_n)$ and $d_{n,k}(\mathbf{x}_n)$, respectively; however, it does not know the coefficients of other sensors, $\alpha_{n,k',r}$ and $\gamma_{n,k',r}$ with $k' \neq k$. The coefficients $\alpha_{n,k,r}$ and $\gamma_{n,k,r}$ can either be precomputed, or each sensor can calculate them locally. A method for calculating these coefficients will be reviewed in Section III-C.

Substituting $\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n)$ for $\mathbf{a}_{n,k}(\mathbf{x}_n)$ and $\tilde{d}_{n,k}(\mathbf{x}_n)$ for $d_{n,k}(\mathbf{x}_n)$ in (10), we obtain the following approximation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$:

$$\begin{aligned} \tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n) &\triangleq \sum_{k=1}^K [\tilde{\mathbf{a}}_{n,k}^\top(\mathbf{x}_n) \mathbf{b}_{n,k}(\mathbf{z}_{n,k}) - \tilde{d}_{n,k}(\mathbf{x}_n)] \quad (13) \\ &= \sum_{k=1}^K \left[\left(\sum_{r=1}^{R_a} \alpha_{n,k,r}^\top \varphi_{n,r}(\mathbf{x}_n) \right) \mathbf{b}_{n,k}(\mathbf{z}_{n,k}) \right. \\ &\quad \left. - \sum_{r=1}^{R_d} \gamma_{n,k,r} \psi_{n,r}(\mathbf{x}_n) \right]. \end{aligned}$$

By changing the order of summation, we obtain further

$$\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{r=1}^{R_a} A_{n,r}(\mathbf{z}_n) \varphi_{n,r}(\mathbf{x}_n) - \sum_{r=1}^{R_d} \Gamma_{n,r} \psi_{n,r}(\mathbf{x}_n), \quad (14)$$

with

$$A_{n,r}(\mathbf{z}_n) \triangleq \sum_{k=1}^K \alpha_{n,k,r}^\top \mathbf{b}_{n,k}(\mathbf{z}_{n,k}), \quad \Gamma_{n,r} \triangleq \sum_{k=1}^K \gamma_{n,k,r}. \quad (15)$$

Finally, substituting $\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n)$ from (14) for $S_n(\mathbf{z}_n, \mathbf{x}_n)$ in (8), an approximation of the JLF is obtained as

$$\begin{aligned} \tilde{f}(\mathbf{z}_n | \mathbf{x}_n) &\propto \exp(\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n)) \\ &= \exp\left(\sum_{r=1}^{R_a} A_{n,r}(\mathbf{z}_n) \varphi_{n,r}(\mathbf{x}_n) - \sum_{r=1}^{R_d} \Gamma_{n,r} \psi_{n,r}(\mathbf{x}_n) \right). \quad (16) \end{aligned}$$

This shows that a sensor that knows $A_{n,r}(\mathbf{z}_n)$ and $\Gamma_{n,r}$ can evaluate an approximation of the JLF (up to a \mathbf{z}_n -dependent but \mathbf{x}_n -independent normalization factor) for all values of \mathbf{x}_n . In fact, the vector of all coefficients $A_{n,r}(\mathbf{z}_n)$ and $\Gamma_{n,r}$, $\tilde{\mathbf{t}}_n(\mathbf{z}_n) \triangleq (A_{n,1}(\mathbf{z}_n) \cdots A_{n,R_a}(\mathbf{z}_n) \Gamma_{n,1} \cdots \Gamma_{n,R_d})^\top$, can be viewed as a *sufficient statistic* [29] that epitomizes the total measurement \mathbf{z}_n within the limits of our approximation. Because of expression (16), this sufficient statistic fully describes the approximate JLF $\tilde{f}(\mathbf{z}_n | \mathbf{x}_n)$ as a function of \mathbf{x}_n .

The expressions (14) and (15) allow a distributed calculation of $\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n)$ and, in turn, of $\tilde{f}(\mathbf{z}_n | \mathbf{x}_n)$ by means of consensus algorithms, due to the following key facts. (i) The coefficients $A_{n,r}(\mathbf{z}_n)$ and $\Gamma_{n,r}$ do not depend on the state \mathbf{x}_n but contain the information of all sensors (the sensor measurements $\mathbf{z}_{n,k}$ and approximation coefficients $\alpha_{n,k,r}$ and $\gamma_{n,k,r}$ for all k). (ii) The state \mathbf{x}_n enters into $\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n)$ only via the functions $\varphi_{n,r}(\cdot)$ and $\psi_{n,r}(\cdot)$, which are sensor-independent and known to each sensor. (iii) According to (15), the coefficients $A_{n,r}(\mathbf{z}_n)$ and $\Gamma_{n,r}$ are sums in which each term contains only local information of a single sensor. These facts form the basis of the LC method, which will be presented in Section IV-A.

Examples of basis functions $\varphi_{n,r}(\cdot)$ and $\psi_{n,r}(\cdot)$ are monomials (see the polynomial expansion discussed below), orthogonal polynomials, and Fourier basis functions. The choice of the basis functions affects the accuracy, computational complexity, and communication requirements of the LC method.

Example—polynomial approximation. A simple example of a basis expansion approximation (11) is given by the polynomial approximation

$$\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n) = \sum_{\mathbf{r}=0}^{R_p} \alpha_{n,k,\mathbf{r}} \prod_{m=1}^M x_{n,m}^{r_m}, \quad (17)$$

where $\mathbf{r} \triangleq (r_1 \cdots r_M) \in \{0, \dots, R_p\}^M$; R_p is the degree of the multivariate vector-valued polynomial $\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n)$; $\sum_{\mathbf{r}=0}^{R_p}$ is short for $\sum_{r_1=0}^{R_p} \cdots \sum_{r_M=0}^{R_p}$ with the constraint $\sum_{m=1}^M r_m \leq R_p$; and $\alpha_{n,k,\mathbf{r}} \in \mathbb{R}^q$ is the coefficient vector associated with the basis function (monomial) $\varphi_{n,\mathbf{r}}(\mathbf{x}_n) = \prod_{m=1}^M x_{n,m}^{r_m}$ (here, $x_{n,m}$ denotes the m th entry of \mathbf{x}_n). We can rewrite (17) in the form of (11) by a suitable index mapping $(r_1 \cdots r_M) \in \{0, \dots, R_p\}^M \leftrightarrow r \in \{1, \dots, R_a\}$, where $R_a = \binom{R_p+M}{R_p}$. An analogous polynomial basis expansion can be used for $\tilde{d}_{n,k}(\mathbf{x}_n)$ in (12). The polynomial basis expansion will be further considered in Section V-B.

C. Least Squares Approximation

A convenient method for calculating the approximations $\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n)$ in (11) and $\tilde{d}_{n,k}(\mathbf{x}_n)$ in (12) is given by least squares (LS) fitting [31]–[33]. We first discuss the calculation of the coefficients $\{\alpha_{n,k,r}\}_{r=1}^{R_a}$ of $\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n)$ at time n and sensor k . Consider J data pairs $\{(\mathbf{x}_{n,k}^{(j)}, \mathbf{a}_{n,k}(\mathbf{x}_{n,k}^{(j)}))\}_{j=1}^J$, where the state points $\mathbf{x}_{n,k}^{(j)}$ are chosen to “cover” those regions of the \mathbf{x}_n space \mathbb{R}^M where the JLF is expected to be evaluated when estimating \mathbf{x}_n . In particular, in the distributed PF application to be considered in Sections VI and VII, the $\mathbf{x}_{n,k}^{(j)}$ will be the predicted particles. With LS fitting, the coefficients $\alpha_{n,k,r}$ are calculated such that the sum of the squared approximation errors at the state points $\mathbf{x}_{n,k}^{(j)}$, i.e., $\sum_{j=1}^J \|\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_{n,k}^{(j)}) - \mathbf{a}_{n,k}(\mathbf{x}_{n,k}^{(j)})\|^2$, is minimized.

To describe the solution to this minimization problem, we define the coefficient matrix $\mathbf{Y}_{n,k} \triangleq (\alpha_{n,k,1} \cdots \alpha_{n,k,R_a})^\top \in \mathbb{R}^{R_a \times q}$, whose rows are the coefficient vectors $\{\alpha_{n,k,r}\}_{r=1}^{R_a}$. Furthermore, let

$$\begin{aligned} \Phi_{n,k} &\triangleq \begin{pmatrix} \varphi_{n,1}(\mathbf{x}_{n,k}^{(1)}) & \cdots & \varphi_{n,R_a}(\mathbf{x}_{n,k}^{(1)}) \\ \vdots & & \vdots \\ \varphi_{n,1}(\mathbf{x}_{n,k}^{(J)}) & \cdots & \varphi_{n,R_a}(\mathbf{x}_{n,k}^{(J)}) \end{pmatrix} \in \mathbb{R}^{J \times R_a}, \\ \mathbf{A}_{n,k} &\triangleq (\mathbf{a}_{n,k}(\mathbf{x}_{n,k}^{(1)}) \cdots \mathbf{a}_{n,k}(\mathbf{x}_{n,k}^{(J)}))^\top \in \mathbb{R}^{J \times q}. \end{aligned}$$

Then the LS solution for the coefficients $\{\alpha_{n,k,r}\}_{r=1}^{R_a}$ is given by [31]

$$\mathbf{Y}_{n,k} = (\Phi_{n,k}^\top \Phi_{n,k})^{-1} \Phi_{n,k}^\top \mathbf{A}_{n,k}.$$

Here, we assume that $J \geq R_a$ and that the columns of $\Phi_{n,k}$ are linearly independent, so that $\Phi_{n,k}^\top \Phi_{n,k}$ is nonsingular. Note that $J \geq R_a$ means that the number of state points $\mathbf{x}_{n,k}^{(j)}$ is not

smaller than the number of basis functions $\varphi_{n,r}(\mathbf{x}_n)$, for any given n and k .

Similarly, the LS solution for the coefficients $\{\gamma_{n,k,r}\}_{r=1}^{R_d}$ of $\tilde{d}_{n,k}(\mathbf{x}_n)$ in (12) is obtained as $\gamma_{n,k} = (\Psi_{n,k}^\top \Psi_{n,k})^{-1} \Psi_{n,k}^\top \mathbf{d}_{n,k}$, where $\gamma_{n,k} \triangleq (\gamma_{n,k,1} \cdots \gamma_{n,k,R_d})^\top \in \mathbb{R}^{R_d}$, $\Psi_{n,k} \in \mathbb{R}^{J \times R_d}$ is defined like $\Phi_{n,k}$ but with $\{\varphi_{n,r}(\cdot)\}_{r=1}^{R_d}$ replaced by $\{\psi_{n,r}(\cdot)\}_{r=1}^{R_d}$, and $\mathbf{d}_{n,k} \triangleq (d_{n,k}(\mathbf{x}_{n,k}^{(1)}) \cdots d_{n,k}(\mathbf{x}_{n,k}^{(J)}))^\top \in \mathbb{R}^J$. Here, we assume that $J \geq R_d$ and that the columns of $\Psi_{n,k}$ are linearly independent. To summarize, the number of state points $\mathbf{x}_{n,k}^{(j)}$ must satisfy $J \geq \max\{R_a, R_d\}$ for any given n and k .

IV. LIKELIHOOD CONSENSUS

We now present the LC algorithm for local likelihood functions belonging to the exponential family, using the approximation of the JLF discussed in Section III. Subsequently, we will consider a class of JLFs for which an approximation is not needed.

A. Distributed Calculation of the Approximate JLF – The LC Algorithm

Based on the sum expressions (15), the sufficient statistic $\tilde{\mathbf{t}}_n(\mathbf{z}_n) = (A_{n,1}(\mathbf{z}_n) \cdots A_{n,R_a}(\mathbf{z}_n) \Gamma_{n,1} \cdots \Gamma_{n,R_d})^\top$ can be computed at each sensor by means of a distributed, iterative consensus algorithm that requires only communications between neighboring sensors. Here, we use a *linear* consensus algorithm [11] for simplicity; however, other consensus algorithms (e.g., [34]) as well as gossip algorithms (e.g., [10]) could be used as well. In what follows, the superscript (i) denotes the iteration index and $\mathcal{N}_k \subseteq \{1, \dots, K\} \setminus \{k\}$ denotes a fixed set of sensors that are neighbors of sensor k . For simplicity, we only discuss the calculation of $A_{n,r}(\mathbf{z}_n)$, since the same principles apply to the calculation of $\Gamma_{n,r}$ in a straightforward manner. We explain the operations performed by a fixed sensor k ; note that such operations are performed by all sensors simultaneously.

At time n , to compute $A_{n,r}(\mathbf{z}_n) = \sum_{k'=1}^K \alpha_{n,k',r}^\top \times \mathbf{b}_{n,k'}(\mathbf{z}_{n,k'})$, sensor k first initializes its local “state” as $\zeta_k^{(0)} \triangleq \alpha_{n,k,r}^\top \mathbf{b}_{n,k}(\mathbf{z}_{n,k})$. This involves only the quantities $\mathbf{z}_{n,k}$, $\alpha_{n,k,r}$, and $\mathbf{b}_{n,k}(\cdot)$, all of which are available at sensor k ; thus, no communication is required at this initialization stage. Then, at the i th iteration of the consensus algorithm ($i \in \{1, 2, \dots\}$), the following two steps are performed by sensor k :

- Using the previous local state $\zeta_k^{(i-1)}$ and the previous neighbor states $\zeta_{k'}^{(i-1)}$, $k' \in \mathcal{N}_k$ (which were received by sensor k at the previous iteration), the local state of sensor k is updated according to

$$\zeta_k^{(i)} = \omega_{k,k}^{(i)} \zeta_k^{(i-1)} + \sum_{k' \in \mathcal{N}_k} \omega_{k,k'}^{(i)} \zeta_{k'}^{(i-1)}.$$

The choice of the weights $\omega_{k,k'}^{(i)}$ is discussed in [35], [36]. Here, we use the Metropolis weights [36]

$$\omega_{k,k'}^{(i)} \equiv \omega_{k,k'} = \begin{cases} \frac{1}{1 + \max\{|\mathcal{N}_k|, |\mathcal{N}_{k'}|\}}, & k' \neq k, \\ 1 - \sum_{k'' \in \mathcal{N}_k} \omega_{k,k''}, & k' = k, \end{cases}$$

where $|\mathcal{N}_k|$ denotes the number of neighbors of sensor k . (We note that knowledge at sensor k of $|\mathcal{N}_k|$ and $|\mathcal{N}_{k'}|$, $k' \in \mathcal{N}_k$ is not required by certain other choices of the weights [36].)

- The new local state $\zeta_k^{(i)}$ is broadcast to all neighbors $k' \in \mathcal{N}_k$.

These two steps are repeated in an iterative manner until a desired degree of convergence is reached.

If the communication graph of the sensor network is connected, the state $\zeta_k^{(i)}$ of each sensor k converges to the average $\frac{1}{K} \sum_{k'=1}^K \alpha_{n,k',r}^\top \mathbf{b}_{n,k'}(\mathbf{z}_{n,k'}) = \frac{1}{K} A_{n,r}(\mathbf{z}_n)$ as $i \rightarrow \infty$ [11]. Therefore, after convergence, the states $\zeta_k^{(i \rightarrow \infty)}$ of all sensors are equal and hence a consensus on the value of $\frac{1}{K} A_{n,r}(\mathbf{z}_n)$ is achieved. For a finite number i_{\max} of iterations, the states $\zeta_k^{(i_{\max})}$ will be (slightly) different for different sensors k and also from the desired value $\frac{1}{K} A_{n,r}(\mathbf{z}_n)$. In what follows, we assume that i_{\max} is sufficiently large so that $K \zeta_k^{(i_{\max})} \approx A_{n,r}(\mathbf{z}_n)$ with sufficient accuracy, for all k . (In the simulations presented in Section VIII, $i_{\max} \in \{7, 8, 9, 10\}$, which arguably does not imply impractical communication requirements.) Note that in order to calculate the coefficient $A_{n,r}(\mathbf{z}_n)$ from $\zeta_k^{(i_{\max})}$, each sensor needs to know 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., [37]).

The consensus-based calculations of all $A_{n,r}(\mathbf{z}_n)$, $r = 1, \dots, R_a$ and all $\Gamma_{n,r}$, $r = 1, \dots, R_d$ are executed simultaneously, and their iterations are synchronized. These consensus algorithms taken together form the LC algorithm, which is stated in what follows.

ALGORITHM 1: LIKELIHOOD CONSENSUS (LC)

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

- 1) Calculate the coefficients $\{\alpha_{n,k,r}\}_{r=1}^{R_a}$ and $\{\gamma_{n,k,r}\}_{r=1}^{R_d}$ of the approximations (11) and (12).
- 2) *Consensus algorithm*— $A_{n,r}(\mathbf{z}_n)$: For each $r = 1, \dots, R_a$:
 - a) Initialize the local state as $\zeta_k^{(0)} = \alpha_{n,k,r}^\top \mathbf{b}_{n,k}(\mathbf{z}_{n,k})$.
 - b) For $i = 1, 2, \dots, i_{\max}$ (here, i_{\max} is a predetermined iteration count or determined by the condition that $|\zeta_k^{(i)} - \zeta_k^{(i-1)}|$ falls below a given threshold):
 - Update the local state according to $\zeta_k^{(i)} = \omega_{k,k}^{(i)} \zeta_k^{(i-1)} + \sum_{k' \in \mathcal{N}_k} \omega_{k,k'}^{(i)} \zeta_{k'}^{(i-1)}$.
 - Broadcast the new state $\zeta_k^{(i)}$ to all neighbors $k' \in \mathcal{N}_k$.
 - c) Calculate $\tilde{A}_{n,r}(\mathbf{z}_n) \triangleq K \zeta_k^{(i_{\max})}$.
- 3) *Consensus algorithm*— $\Gamma_{n,r}$: For each $r = 1, \dots, R_d$:
 - a) Initialize the local state as $\zeta_k^{(0)} = \gamma_{n,k,r}$.
 - b) Same as 2b).
 - c) Calculate $\tilde{\Gamma}_{n,r} \triangleq K \zeta_k^{(i_{\max})}$.

Finally, by substituting $\tilde{A}_{n,r}(\mathbf{z}_n)$ for $A_{n,r}(\mathbf{z}_n)$ and $\tilde{\Gamma}_{n,r}$ for $\Gamma_{n,r}$ in (16), sensor k is able to obtain a consensus approximation of the approximate JLF $\tilde{f}(\mathbf{z}_n | \mathbf{x}_n)$ for any given value of \mathbf{x}_n .

Because one consensus algorithm has to be executed for each $A_{n,r}(\mathbf{z}_n)$, $r = 1, \dots, R_a$ and $\Gamma_{n,r}$, $r = 1, \dots, R_d$, the number of consensus algorithms that are executed simultaneously is $N_c = R_a + R_d$. This is also the number of real numbers broadcast by each sensor in each iteration of the LC algorithm. It is important to note that R_a and R_d do not depend on the dimensions $N_{n,k}$ of the measurement vectors $\mathbf{z}_{n,k}$, and thus the communication requirements of LC do not depend on the $N_{n,k}$. This is particularly advantageous in the case of high-dimensional measurements. However, R_a and R_d usually grow with the dimension M of the state vector \mathbf{x}_n . In particular, if the MD basis $\{\varphi_{n,r}(\mathbf{x}_n)\}_{r=1}^{R_a}$ is constructed as the M -fold tensor product of a 1D basis $\{\tilde{\varphi}_{n,\tilde{r}}(x)\}_{\tilde{r}=1}^{\tilde{R}_a}$, then $R_a = \tilde{R}_a^M$, and similarly for the MD basis $\{\psi_{n,r}(\mathbf{x}_n)\}_{r=1}^{R_d}$.

So far, we have disregarded the normalization factor $C_n(\mathbf{z}_n)$ occurring in (8). If this factor is required at each sensor, it can also be computed by a consensus algorithm. From (9),

$$\log C_n(\mathbf{z}_n) = \sum_{k=1}^K \log c_{n,k}(\mathbf{z}_{n,k}).$$

Since this is a sum and $c_{n,k}(\mathbf{z}_{n,k})$ is known to each sensor, a consensus algorithm can again be used for a distributed calculation of $\log C_n(\mathbf{z}_n)$.

B. Distributed Calculation of the Exact JLF

The basis expansion approximations (11) and (12) can be avoided if the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ has a special structure. In that case, the *exact* JLF can be computed in a distributed way, up to errors that are only due to the limited number of consensus iterations performed. We note that the special structure considered now is compatible with the exponential family structure considered so far, but it does not presuppose that structure.

Let $\mathbf{t}_n(\mathbf{z}_n) = (t_{n,1}(\mathbf{z}_n) \cdots t_{n,P}(\mathbf{z}_n))^\top$ be a sufficient statistic for the estimation problem corresponding to $f(\mathbf{z}_n|\mathbf{x}_n)$. According to the Neyman-Fisher factorization theorem [29], $f(\mathbf{z}_n|\mathbf{x}_n)$ can then be written as

$$f(\mathbf{z}_n|\mathbf{x}_n) = f_1(\mathbf{z}_n) f_2(\mathbf{t}_n(\mathbf{z}_n), \mathbf{x}_n). \quad (18)$$

Typically, the factor $f_1(\mathbf{z}_n)$ can be disregarded since it does not depend on \mathbf{x}_n . Thus, $\mathbf{t}_n(\mathbf{z}_n)$ epitomizes the total measurement \mathbf{z}_n , in that a sensor that knows $\mathbf{t}_n(\mathbf{z}_n)$ and $f_2(\cdot, \cdot)$ 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 form

$$t_{n,p}(\mathbf{z}_n) = \sum_{k=1}^K \eta_{n,k,p}(\mathbf{z}_{n,k}), \quad p = 1, \dots, P, \quad (19)$$

with arbitrary functions $\eta_{n,k,p}(\cdot)$, and that sensor k knows its own functions $\eta_{n,k,p}(\cdot)$ but not $\eta_{n,k',p}(\cdot)$, $k' \neq k$. Based on the sum expression (19), we can then use consensus algorithms as described in Section IV-A, with obvious modifications, to calculate $\mathbf{t}_n(\mathbf{z}_n)$ and, thus, the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ in a distributed manner.

Clearly, an example where exact calculation of the JLF is possible is the case where $f(\mathbf{z}_n|\mathbf{x}_n)$ belongs to the exponential

family (7), with functions $\mathbf{a}_{n,k}(\mathbf{x}_n)$ and $d_{n,k}(\mathbf{x}_n)$ that can be exactly represented using expansions of the form (11) and (12), i.e., $\mathbf{a}_{n,k}(\mathbf{x}_n) = \sum_{r=1}^{R_a} \alpha_{n,k,r} \varphi_{n,r}(\mathbf{x}_n)$ and $d_{n,k}(\mathbf{x}_n) = \sum_{r=1}^{R_d} \gamma_{n,k,r} \psi_{n,r}(\mathbf{x}_n)$. This is a special case of (18) and (19), with (cf. (16))

$$f_2(\mathbf{t}_n(\mathbf{z}_n), \mathbf{x}_n) = \exp\left(\sum_{p=1}^P t_{n,p}(\mathbf{z}_n) \rho_{n,p}(\mathbf{x}_n)\right),$$

where $P = R_a + R_d$ and

$$t_{n,p}(\mathbf{z}_n) = \begin{cases} A_{n,p}(\mathbf{z}_n) = \sum_{k=1}^K \alpha_{n,k,p}^\top \mathbf{b}_{n,k}(\mathbf{z}_{n,k}), & p = 1, \dots, R_a, \\ -\Gamma_{n,p-R_a} = -\sum_{k=1}^K \gamma_{n,k,p-R_a}, & p = R_a+1, \dots, P, \end{cases}$$

$$\rho_{n,p}(\mathbf{x}_n) = \begin{cases} \varphi_{n,p}(\mathbf{x}_n), & p = 1, \dots, R_a, \\ \psi_{n,p-R_a}(\mathbf{x}_n), & p = R_a+1, \dots, P. \end{cases}$$

Equivalently, $t_{n,p}(\mathbf{z}_n)$ is of the form (19), with (cf. (15))

$$\eta_{n,k,p}(\mathbf{z}_{n,k}) = \begin{cases} \alpha_{n,k,p}^\top \mathbf{b}_{n,k}(\mathbf{z}_{n,k}), & p = 1, \dots, R_a, \\ -\gamma_{n,k,p-R_a}, & p = R_a+1, \dots, P. \end{cases}$$

V. SPECIAL CASE: GAUSSIAN MEASUREMENT NOISE

In this section, we consider the important special case of (generally nonlinear) measurement functions and independent additive Gaussian measurement noises at the various sensors. We will also develop the application of the polynomial approximation that was briefly introduced in Section III-B.

A. Measurement Model

The dependence of the sensor measurements $\mathbf{z}_{n,k}$ on the state \mathbf{x}_n is described by the local likelihood functions $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$. Let us now assume, more specifically, that the measurements are modeled as

$$\mathbf{z}_{n,k} = \mathbf{h}_{n,k}(\mathbf{x}_n) + \mathbf{v}_{n,k}, \quad k = 1, \dots, K, \quad (20)$$

where $\mathbf{h}_{n,k}(\cdot)$ is the *measurement function* of sensor k and $\mathbf{v}_{n,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{n,k})$ is zero-mean Gaussian measurement noise that is independent of $\mathbf{x}_{n'}$ for all n' . We furthermore assume that $\mathbf{v}_{n,k}$ and $\mathbf{v}_{n',k'}$ are independent unless $(n,k) = (n',k')$. Under these assumptions, the $\mathbf{z}_{n,k}$ are conditionally independent given \mathbf{x}_n , i.e., (1) holds. The local likelihood function of sensor k is here given by

$$f(\mathbf{z}_{n,k}|\mathbf{x}_n) = \bar{c}_{n,k} \exp\left(-\frac{1}{2} [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]^\top \mathbf{Q}_{n,k}^{-1} [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]\right), \quad (21)$$

with $\bar{c}_{n,k} \triangleq [(2\pi)^{N_{n,k}} \det\{\mathbf{Q}_{n,k}\}]^{-1/2}$. Furthermore, using (1), the JLF is obtained as

$$f(\mathbf{z}_n|\mathbf{x}_n) = \bar{c}_n \exp\left(-\frac{1}{2} \sum_{k=1}^K [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]^\top \mathbf{Q}_{n,k}^{-1} [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]\right), \quad (22)$$

with $\bar{c}_n = \prod_{k=1}^K \bar{c}_{n,k}$.

The local likelihood function $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ in (21) is a special case of the exponential family (6), with

$$\mathbf{a}_{n,k}(\mathbf{x}_n) = \mathbf{h}_{n,k}(\mathbf{x}_n), \quad (23)$$

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

$$c_{n,k}(\mathbf{z}_{n,k}) = \bar{c}_{n,k} \exp\left(-\frac{1}{2} \mathbf{z}_{n,k}^\top \mathbf{Q}_{n,k}^{-1} \mathbf{z}_{n,k}\right),$$

$$d_{n,k}(\mathbf{x}_n) = \frac{1}{2} \mathbf{h}_{n,k}^\top(\mathbf{x}_n) \mathbf{Q}_{n,k}^{-1} \mathbf{h}_{n,k}(\mathbf{x}_n). \quad (24)$$

Consequently (see (10)),

$$S_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{k=1}^K \mathbf{h}_{n,k}^\top(\mathbf{x}_n) \mathbf{Q}_{n,k}^{-1} \left[\mathbf{z}_{n,k} - \frac{1}{2} \mathbf{h}_{n,k}(\mathbf{x}_n) \right]. \quad (25)$$

We now approximate $\mathbf{a}_{n,k}(\mathbf{x}_n)$ and $d_{n,k}(\mathbf{x}_n)$ by truncated basis expansions $\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n)$ and $\tilde{d}_{n,k}(\mathbf{x}_n)$ of the form (11) and (12), respectively. According to (23), approximating $\mathbf{a}_{n,k}(\mathbf{x}_n)$ is equivalent to approximating the sensor measurement function $\mathbf{h}_{n,k}(\mathbf{x}_n)$ (which is also the mean of $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ in (21)). Thus,

$$\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n) = \tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n) = \sum_{r=1}^{R_a} \alpha_{n,k,r} \varphi_{n,r}(\mathbf{x}_n). \quad (26)$$

Furthermore, an approximation of $d_{n,k}(\mathbf{x}_n)$ of the form (12) can be obtained in an indirect way by substituting in (24) the above approximation $\tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n)$ for $\mathbf{h}_{n,k}(\mathbf{x}_n)$; this yields

$$\begin{aligned} \tilde{d}_{n,k}(\mathbf{x}_n) &= \frac{1}{2} \tilde{\mathbf{h}}_{n,k}^\top(\mathbf{x}_n) \mathbf{Q}_{n,k}^{-1} \tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n) \\ &= \frac{1}{2} \sum_{r_1=1}^{R_a} \sum_{r_2=1}^{R_a} \alpha_{n,k,r_1}^\top \mathbf{Q}_{n,k}^{-1} \alpha_{n,k,r_2} \varphi_{n,r_1}(\mathbf{x}_n) \varphi_{n,r_2}(\mathbf{x}_n). \end{aligned} \quad (27)$$

(28)

Using a suitable index mapping $(r_1, r_2) \in \{1, \dots, R_a\} \times \{1, \dots, R_a\} \leftrightarrow r \in \{1, \dots, R_d\}$, we can write (28) in the form (12):

$$\tilde{d}_{n,k}(\mathbf{x}_n) = \sum_{r=1}^{R_d} \gamma_{n,k,r} \psi_{n,r}(\mathbf{x}_n),$$

with $R_d = R_a^2$, $\gamma_{n,k,r} = \frac{1}{2} \alpha_{n,k,r_1}^\top \mathbf{Q}_{n,k}^{-1} \alpha_{n,k,r_2}$, and $\psi_{n,r}(\mathbf{x}_n) = \varphi_{n,r_1}(\mathbf{x}_n) \varphi_{n,r_2}(\mathbf{x}_n)$. It is easily verified that with this special basis expansion approximation of $d_{n,k}(\mathbf{x}_n)$, the resulting approximate JLF can be written as

$$\begin{aligned} &\tilde{f}(\mathbf{z}_n|\mathbf{x}_n) \\ &= \bar{c}_n \exp\left(-\frac{1}{2} \sum_{k=1}^K [\mathbf{z}_{n,k} - \tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n)]^\top \mathbf{Q}_{n,k}^{-1} [\mathbf{z}_{n,k} - \tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n)]\right), \end{aligned}$$

which is (22) with $\mathbf{h}_{n,k}(\mathbf{x}_n)$ replaced by $\tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n)$. This means that only the mean of $f(\mathbf{z}_n|\mathbf{x}_n)$ is changed by this approximation.

In the additive Gaussian noise setting considered, the LC method operates almost as in the general case. The only difference is in Step 1 of Algorithm 1: instead of calculating the coefficients $\gamma_{n,k,r}$ directly, using, e.g., a separate LS fitting, we obtain them in an indirect way as described above. Hence, the computational complexity is reduced. Note that in general,

the directly and indirectly obtained coefficients $\gamma_{n,k,r}$ will be different. Furthermore, if the indirectly obtained coefficients are used, the approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ is a valid pdf in the sense that $\int \tilde{f}(\mathbf{z}_n|\mathbf{x}_n) d\mathbf{z}_n = 1$ holds exactly, not only approximately. The number of consensus algorithms that are executed is $N_c = R_a + R_d = R_a + R_a^2$. Again, this does not depend on the dimensions $N_{n,k}$ of the measurement vectors $\mathbf{z}_{n,k}$ since R_a does not depend on $N_{n,k}$.

B. Polynomial Approximation

The polynomial approximation was introduced in Section III-B. We will now apply it to the case of Gaussian measurement noise studied above. Using (17), we obtain for (26)

$$\tilde{\mathbf{a}}_{n,k}(\mathbf{x}_n) = \tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n) = \sum_{\mathbf{r}=0}^{R_p} \alpha_{n,k,\mathbf{r}} \prod_{m=1}^M x_{n,m}^{r_m}. \quad (29)$$

Inserting this into (27) yields

$$\tilde{d}_{n,k}(\mathbf{x}_n) = \sum_{\mathbf{r}=0}^{2R_p} \gamma_{n,k,\mathbf{r}} \prod_{m=1}^M x_{n,m}^{r_m}, \quad (30)$$

with

$$\gamma_{n,k,\mathbf{r}} = \frac{1}{2} \sum_{\substack{\mathbf{r}'=0 \\ \mathbf{r}'+\mathbf{r}''=\mathbf{r}}}^{R_p} \sum_{\mathbf{r}''=0}^{R_p} \alpha_{n,k,\mathbf{r}'}^\top \mathbf{Q}_{n,k}^{-1} \alpha_{n,k,\mathbf{r}''}. \quad (31)$$

Next, inserting expressions (29) and (30) into (13), we obtain

$$\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{k=1}^K \sum_{\mathbf{r}=0}^{2R_p} \beta_{n,k,\mathbf{r}}(\mathbf{z}_{n,k}) \prod_{m=1}^M x_{n,m}^{r_m}, \quad (32)$$

with

$$\beta_{n,k,\mathbf{r}}(\mathbf{z}_{n,k}) = \begin{cases} \alpha_{n,k,\mathbf{r}}^\top \mathbf{b}_{n,k}(\mathbf{z}_{n,k}) - \gamma_{n,k,\mathbf{r}}, & \mathbf{r} \in \mathcal{R}_1 \\ -\gamma_{n,k,\mathbf{r}}, & \mathbf{r} \in \mathcal{R}_2, \end{cases} \quad (33)$$

where \mathcal{R}_1 is the set of all $\mathbf{r} = (r_1 \dots r_M) \in \{0, \dots, R_p\}^M$ such that $\sum_{m=1}^M r_m \leq R_p$ and \mathcal{R}_2 is the set of all $\mathbf{r} \in \{0, \dots, 2R_p\}^M \setminus \mathcal{R}_1$ such that $\sum_{m=1}^M r_m \leq 2R_p$. Finally, changing the order of summation in (32) gives

$$\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{\mathbf{r}=0}^{2R_p} B_{n,\mathbf{r}}(\mathbf{z}_n) \prod_{m=1}^M x_{n,m}^{r_m}, \quad (34)$$

with

$$B_{n,\mathbf{r}}(\mathbf{z}_n) = \sum_{k=1}^K \beta_{n,k,\mathbf{r}}(\mathbf{z}_{n,k}). \quad (35)$$

It should be noted that (34) is a special case of (14). The coefficients $B_{n,\mathbf{r}}(\mathbf{z}_n)$ can again be calculated using a consensus algorithm. For each time n , the number of coefficients $B_{n,\mathbf{r}}(\mathbf{z}_n)$, and hence the number of consensus algorithms that have to be executed in parallel, is given by $N_c = \binom{2R_p+M}{2R_p} - 1$. Here, the subtraction of 1 is due to the fact that the coefficient $B_{n,\mathbf{r}=0}(\mathbf{z}_n)$ need not be calculated: according to (34), $B_{n,\mathbf{0}}(\mathbf{z}_n)$ corresponds to a JLF factor that does not depend on \mathbf{x}_n and is hence irrelevant.

VI. DISTRIBUTED PARTICLE FILTERING

In this section, we show how the LC method can be applied to obtain a distributed PF. By way of preparation, we first review a standard centralized PF [14], [15], [28].

A. Review of Centralized Particle Filtering

The centralized PF is implemented at a fusion center that knows the all-sensors measurement vector \mathbf{z}_n and the functional form of the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$. The PF maintains a set of samples (or particles) $\{\mathbf{x}_n^{(j)}\}_{j=1}^J$ and associated weights $\{w_n^{(j)}\}_{j=1}^J$, which establish the following approximative sample representation of the posterior pdf $f(\mathbf{x}_n|\mathbf{z}_{1:n})$:

$$f_\delta(\mathbf{x}_n|\mathbf{z}_{1:n}) \triangleq \sum_{j=1}^J w_n^{(j)} \delta(\mathbf{x}_n - \mathbf{x}_n^{(j)}).$$

The MMSE estimate in (4) can then be approximated by the mean of $f_\delta(\mathbf{x}_n|\mathbf{z}_{1:n})$, which is equivalent to a weighted sample mean:

$$\hat{\mathbf{x}}_n \triangleq \int \mathbf{x}_n f_\delta(\mathbf{x}_n|\mathbf{z}_{1:n}) d\mathbf{x}_n = \sum_{j=1}^J w_n^{(j)} \mathbf{x}_n^{(j)}. \quad (36)$$

At each time step n , when the new measurement vector \mathbf{z}_n becomes available, new particles and weights are calculated by a PF algorithm that is based on the recursion (5).

Many PF algorithms have been proposed [13]–[15], [28]. Here, we consider a sequential importance resampling filter [13], [15], which performs the following steps. For initialization ($n = 0$), J particles $\mathbf{x}_0^{(j)}$ are sampled from a prior distribution $f(\mathbf{x}_0)$, and the weights are set to $w_0^{(j)} \equiv 1/J$. Then, three steps—resampling, sampling, and weight update—are repeated for every n . In the *resampling step*, J resampled particles $\bar{\mathbf{x}}_{n-1}^{(j)}$ are obtained by sampling with replacement from the set of previous particles $\{\mathbf{x}_{n-1}^{(j')}\}_{j'=1}^J$, where the probability of sampling $\mathbf{x}_{n-1}^{(j')}$ is $w_{n-1}^{(j')}$. In the *sampling step*, for each resampled particle $\bar{\mathbf{x}}_{n-1}^{(j)}$, a new, “predicted” particle $\mathbf{x}_n^{(j)}$ is sampled from $f(\mathbf{x}_n|\bar{\mathbf{x}}_{n-1}^{(j)})$, i.e., from the state-transition pdf $f(\mathbf{x}_n|\mathbf{x}_{n-1})$ evaluated at $\mathbf{x}_{n-1} = \bar{\mathbf{x}}_{n-1}^{(j)}$. In the *weight update step*, the weight associated with each particle $\mathbf{x}_n^{(j)}$ is calculated as

$$w_n^{(j)} = \frac{f(\mathbf{z}_n|\mathbf{x}_n^{(j)})}{\sum_{j'=1}^J f(\mathbf{z}_n|\mathbf{x}_n^{(j')})}. \quad (37)$$

Finally, the state estimate $\hat{\mathbf{x}}_n$ is calculated from $\{\mathbf{x}_n^{(j)}, w_n^{(j)}\}_{j=1}^J$ according to (36).

B. Distributed Particle Filtering Using LC

Next, we develop a distributed implementation of the sequential importance resampling filter reviewed above, in which each sensor acts similarly to the fusion center of the centralized PF. More specifically, sensor k tracks a particle representation of the global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ using a *local PF*. For each n , it obtains a state estimate $\hat{\mathbf{x}}_{n,k}$ that is based on $\mathbf{z}_{1:n}$, i.e., the past and current measurements of *all* sensors.

This requires each sensor to know the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ as a function of the state \mathbf{x}_n , because the weight update in (37) requires the pointwise evaluation of the JLF. Therefore, an approximation of the JLF is provided to each sensor in a distributed way by means of the LC method. No routing of measurements or other sensor-local data is needed; each sensor merely broadcasts information to neighboring sensors. The algorithm is stated as follows.

ALGORITHM 2: LC-BASED DISTRIBUTED PF (LC-DPF)

At time n , the local PF at sensor k performs the following steps, which are identical for all k . (Note that these steps are essentially analogous to those of the centralized PF of Section VI-A, except that an approximation of the JLF is used.)

- 1) At the previous time $n-1$, sensor k calculated J particles $\mathbf{x}_{n-1,k}^{(j)}$ and weights $w_{n-1,k}^{(j)}$, which together represent the previous global posterior $f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1})$. The first step at time n is a resampling of $\{(\mathbf{x}_{n-1,k}^{(j)}, w_{n-1,k}^{(j)})\}_{j=1}^J$, which produces J resampled particles $\bar{\mathbf{x}}_{n-1,k}^{(j)}$. Here, the $\bar{\mathbf{x}}_{n-1,k}^{(j)}$ are obtained by sampling with replacement from the set $\{\mathbf{x}_{n-1,k}^{(j')}\}_{j'=1}^J$, where $\mathbf{x}_{n-1,k}^{(j')}$ is sampled with probability $w_{n-1,k}^{(j')}$.
- 2) For each $\bar{\mathbf{x}}_{n-1,k}^{(j)}$, a new, “predicted” particle $\mathbf{x}_{n,k}^{(j)}$ is sampled from $f(\mathbf{x}_n|\mathbf{x}_{n-1})|_{\mathbf{x}_{n-1}=\bar{\mathbf{x}}_{n-1,k}^{(j)}}$.
- 3) An approximation $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ of the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ is computed by means of LC as described in Section IV-A. This step requires communications with neighboring sensors. The local approximation at sensor k can be calculated by means of LS fitting as described in Section III-C, using the predicted particles $\{\mathbf{x}_{n,k}^{(j)}\}_{j=1}^J$.
- 4) The weights associated with the predicted particles $\mathbf{x}_{n,k}^{(j)}$ obtained in Step 2 are calculated according to

$$w_{n,k}^{(j)} = \frac{\tilde{f}(\mathbf{z}_n|\mathbf{x}_{n,k}^{(j)})}{\sum_{j'=1}^J \tilde{f}(\mathbf{z}_n|\mathbf{x}_{n,k}^{(j')})}, \quad j = 1, \dots, J. \quad (38)$$

This involves the approximate JLF $\tilde{f}(\mathbf{z}_n|\mathbf{x}_n)$ calculated in Step 3, which is evaluated at all predicted particles $\mathbf{x}_{n,k}^{(j)}$.

- 5) From $\{(\mathbf{x}_{n,k}^{(j)}, w_{n,k}^{(j)})\}_{j=1}^J$, an approximation of the global MMSE state estimate (4) is computed according to (36), i.e.,

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

The recursion defined by Steps 1–5 is initialized at $n = 0$ by J particles $\mathbf{x}_{0,k}^{(j)}$ sampled (at each sensor) from a suitable prior pdf $f(\mathbf{x}_0)$, and by equal weights $w_{0,k}^{(j)} \equiv 1/J$.

Through the above recursion, each sensor obtains a global quasi-MMSE state estimate that involves the past and current measurements of all sensors. Because of the use of LC, this is achieved without communicating between distant sensors or employing complex routing protocols. Also, no particles, local state estimates, or measurements need to be communicated between sensors. The local PF algorithms running at different sensors are identical. Therefore, any differences between the state estimates $\hat{\mathbf{x}}_{n,k}$ at different sensors k are only due to the random sampling of the particles (using nonsynchronized

local random generators) and errors caused by insufficiently converged consensus algorithms.

C. Communication Requirements

We now discuss the communication requirements of our LC-based distributed PF (LC-DPF). For comparison, we also consider the centralized PF (CPF) of Section VI-A, in which all sensor measurements are transmitted to a fusion center (FC), and a straightforward distributed PF implementation (S-DPF) in which the measurements of each sensor are transmitted to all other sensors. Note that with the S-DPF, each sensor performs exactly the same PF operations as does the FC in the CPF scheme.

For the CPF, communicating all sensor measurements at time n to the FC requires the transmission of a total of $\sum_{k=1}^K H_k N_{n,k}$ real numbers within the sensor network [25]. Here, H_k denotes the number of communication hops from sensor k to the FC, and $N_{n,k}$ is the dimension of $\mathbf{z}_{n,k}$. Additional information needs to be transmitted to the FC if the FC does not possess prior knowledge of the JLF. Finally, if the state estimate calculated at the FC is required to be available at the sensors, additional MH' real numbers need to be transmitted at each time n . Here, H' denotes the number of communication hops needed to disseminate the state estimate throughout the network. A problem of the CPF using multihop transmission is that all data pass through a small subset of sensors surrounding the FC, which can lead to fast depletion of the batteries of these sensors.

With the S-DPF, disseminating the measurements of all sensors at time n to all other sensors requires the transmission of $\sum_{k=1}^K H_k'' N_{n,k}$ real numbers [25], where H_k'' is the number of communication hops required to disseminate the measurement of sensor k throughout the network. Again, additional information needs to be transmitted if the JLF is not known to all sensors.

Finally, the proposed LC-DPF requires the transmission of KIN_c real numbers at each time n , where I is the number of consensus iterations performed by each consensus algorithm and $N_c = R_a + R_d$ is the number of consensus algorithms executed in parallel (see Section IV-A). In contrast to the CPF and S-DPF, this number of transmissions does not depend on the measurement dimensions $N_{n,k}$; this makes the LC-DPF particularly attractive in the case of high-dimensional measurements. Another advantage of the LC-DPF is that no additional communications are needed (e.g., to transmit local likelihood functions between sensors). Furthermore, the LC-DPF does not require multihop transmissions or routing protocols since each sensor simply broadcasts information to its neighbors. This makes the LC-DPF particularly suited to wireless sensor networks with dynamic network topologies (e.g., moving sensors or a time-varying number of active sensors): in contrast to the CPF and S-DPF, there is no need to rebuild routing tables each time the network topology changes.

On the other hand, the computational complexity of the LC-DPF is higher than that of the S-DPF because the approximation described in Section III needs to be computed at each sensor. Overall, the LC-DPF performs more local computations than the S-DPF in order to reduce communications; this

is especially true for high-dimensional measurements and/or high-dimensional parametrizations of the local likelihood functions. Since the energy consumption of local computations is typically much smaller than that of communication, the total energy consumption is reduced and thus the operation lifetime is extended. This advantage of the LC-DPF comes at the cost of a certain performance loss (compared to the CPF or S-DPF) due to the approximate JLF used by the local PFs. This will be analyzed experimentally in Section VIII.

VII. DISTRIBUTED GAUSSIAN PARTICLE FILTERING

Next, we propose two distributed versions of the *Gaussian PF* (GPF). The GPF was introduced in [18] as a simplified version of the PF using a Gaussian approximation of the posterior $f(\mathbf{x}_n | \mathbf{z}_{1:n})$. The mean and covariance of this Gaussian approximation are derived from a weighted particle set. The particles and their weights are computed in a similar way as described in Section VI, with the difference that no resampling is required. This results in a reduced complexity and allows for a parallel implementation [38].

A. Distributed Gaussian Particle Filtering Using LC

In the proposed distributed GPF schemes, sensor k uses a local GPF to track the mean vector $\boldsymbol{\mu}_{n,k}$ and covariance matrix $\mathbf{C}_{n,k}$ of a local Gaussian approximation $\mathcal{N}(\boldsymbol{\mu}_{n,k}, \mathbf{C}_{n,k})$ of the global posterior $f(\mathbf{x}_n | \mathbf{z}_{1:n})$. The state estimate $\hat{\mathbf{x}}_{n,k}$ of sensor k at time n is defined to be the current mean, i.e., $\hat{\mathbf{x}}_{n,k} = \boldsymbol{\mu}_{n,k}$. The calculation of this estimate is based on the past and current measurements of all sensors, $\mathbf{z}_{1:n}$. As with the distributed PF described in Section VI-B (Algorithm 2), these measurements are epitomized by an approximation of the JLF, which is provided to each sensor by means of LC. A statement of the algorithm follows.

ALGORITHM 3: LC-BASED DISTRIBUTED GPF (LC-DGPF)

At time n , the local GPF at sensor k performs the following steps, which are identical for all k .

- 1) J particles $\{\bar{\mathbf{x}}_{n-1,k}^{(j)}\}_{j=1}^J$ are sampled from the previous local Gaussian approximation $\mathcal{N}(\boldsymbol{\mu}_{n-1,k}, \mathbf{C}_{n-1,k})$, where $\boldsymbol{\mu}_{n-1,k}$ and $\mathbf{C}_{n-1,k}$ were calculated at time $n-1$. Note that this sampling step replaces the resampling step of the distributed PF of Section VI-B (Step 1 in Algorithm 2).
- 2)–4) These steps are identical to the corresponding steps of the distributed PF of Section VI-B (Algorithm 2); they involve LC (Step 3) and result in a set of “predicted” particles and corresponding weights, $\{(\mathbf{x}_{n,k}^{(j)}, w_{n,k}^{(j)})\}_{j=1}^J$.
- 5) From $\{(\mathbf{x}_{n,k}^{(j)}, w_{n,k}^{(j)})\}_{j=1}^J$, the mean $\boldsymbol{\mu}_{n,k}$ and covariance $\mathbf{C}_{n,k}$ of the Gaussian approximation $\mathcal{N}(\boldsymbol{\mu}_{n,k}, \mathbf{C}_{n,k})$ of the current posterior $f(\mathbf{x}_n | \mathbf{z}_{1:n})$ are calculated as

$$\begin{aligned} \boldsymbol{\mu}_{n,k} &= \sum_{j=1}^J w_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)} \\ \mathbf{C}_{n,k} &= \sum_{j=1}^J w_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)\top} - \boldsymbol{\mu}_{n,k} \boldsymbol{\mu}_{n,k}^\top. \end{aligned} \quad (39)$$

The state estimate $\hat{\mathbf{x}}_{n,k}$ (approximating $\hat{\mathbf{x}}_n^{\text{MMSE}}$ in (4)) is taken to be the posterior mean $\boldsymbol{\mu}_{n,k}$.

The recursion defined by Steps 1–5 is initialized as in Algorithm 2.

B. Reduced-Complexity Method

We next present a reduced-complexity variant of the LC-DGPF described above, in which each of the K local GPFs uses only $J' \triangleq J/K$ particles. Here, J' is chosen such that J' is an integer and $J' \geq \max\{R_a, R_d\}$ (cf. Section III-C). The sets of J' particles of all local GPFs are effectively combined via a second stage of consensus algorithms, such that a “virtual global GPF” with $J = KJ'$ particles is obtained. In other words, J particles—which, in the LC-DGPF, were used by each individual sensor separately—are “distributed” over the K sensors. As a consequence, the computational complexity of the local GPFs is substantially reduced while the estimation accuracy remains effectively unchanged. This advantage comes at the cost of some increase in local communications due to the additional consensus algorithms.

This reduced-complexity method is similar to a parallel GPF implementation proposed in [38], which uses multiple processing units—corresponding to our sensors—collocated with a central unit. However, instead of a central unit, we employ distributed consensus algorithms to combine the partial estimates (means) and partial covariances calculated at the individual sensors. Another difference from [38] is the use of an approximate JLF that is obtained in a distributed way by means of LC. The algorithm is stated as follows.

ALGORITHM 4: REDUCED-COMPLEXITY LC-DGPF (R-LC-DGPF)

At time n , the local GPF at sensor k first performs Steps 1–3 of the LC-DGPF algorithm described in Section VII-A (Algorithm 3), however using $J' = J/K$ rather than J particles. The remaining steps, described in the following, are modified versions of Steps 4 and 5 of Algorithm 3, as well as an additional consensus step.

- 4) Nonnormalized weights are calculated as (cf. (38))

$$\tilde{w}_{n,k}^{(j)} = \tilde{f}(\mathbf{z}_n | \mathbf{x}_{n,k}^{(j)}), \quad j = 1, \dots, J'.$$

This requires evaluation of the approximate JLF $\tilde{f}(\mathbf{z}_n | \mathbf{x}_n)$, which was calculated in Step 3 using LC, at the J' predicted particles $\{\mathbf{x}_{n,k}^{(j)}\}_{j=1}^{J'}$ drawn in Step 2. Furthermore, the sum of the J' nonnormalized weights is computed:

$$\tilde{W}_{n,k} = \sum_{j=1}^{J'} \tilde{w}_{n,k}^{(j)}.$$

- 5) From the weighted particles $\{(\mathbf{x}_{n,k}^{(j)}, \tilde{w}_{n,k}^{(j)})\}_{j=1}^{J'}$, a partial nonnormalized mean and a partial nonnormalized correlation are calculated as

$$\boldsymbol{\mu}'_{n,k} = \sum_{j=1}^{J'} \tilde{w}_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)}, \quad \mathbf{R}'_{n,k} = \sum_{j=1}^{J'} \tilde{w}_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)\top}, \quad (40)$$

respectively. Note that Steps 4 and 5 are carried out locally at sensor k .

- 6) The partial means and correlations from all sensors are combined to obtain the global mean and covariance:

$$\boldsymbol{\mu}_n = \frac{1}{W_n} \sum_{k=1}^K \boldsymbol{\mu}'_{n,k}, \quad \mathbf{C}_n = \frac{1}{W_n} \sum_{k=1}^K \mathbf{R}'_{n,k} - \boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top, \quad (41)$$

where

$$W_n = \sum_{k=1}^K \tilde{W}_{n,k} \quad (42)$$

is the global sum of all particle weights. The sums over all sensors in (41) and (42) are computed in a distributed manner by means of consensus algorithms. The normalization by W_n and subtraction of $\boldsymbol{\mu}_n \boldsymbol{\mu}_n^\top$ in (41) are performed locally at each sensor after convergence of these consensus algorithms. The state estimate $\hat{\mathbf{x}}_n$ is taken to be $\boldsymbol{\mu}_n$.

As a result of this algorithm, all sensors obtain identical $\hat{\mathbf{x}}_n = \boldsymbol{\mu}_n$ and \mathbf{C}_n provided that the consensus algorithms are sufficiently converged. Therefore, we omit the subscript k indicating the sensor dependence (cf. (39)), i.e., we write $\hat{\mathbf{x}}_n = \boldsymbol{\mu}_n$ instead of $\hat{\mathbf{x}}_{n,k} = \boldsymbol{\mu}_{n,k}$ and \mathbf{C}_n instead of $\mathbf{C}_{n,k}$ for all k .

It is easily seen from (40)–(42) that $\boldsymbol{\mu}_n$ and \mathbf{C}_n are actually the result of an averaging (summation) over J particles (note that $J' = J/K$ particles are sampled independently at each of the K sensors). Therefore, under the assumption that the consensus algorithms used to calculate the sums over all sensors in (41) and (42) are converged, $\boldsymbol{\mu}_n$ and \mathbf{C}_n should ideally be effectively equal to the corresponding quantities obtained by the LC-DGPF. However, a certain performance degradation is caused by the fact that the LS fitting performed at each sensor (see Section III-C) is now based on only $J' = J/K$ predicted particles $\mathbf{x}_{n,k}^{(j)}$, and hence the resulting approximate local likelihood functions and, in turn, the approximate JLF will be less accurate. In Section VIII, we will show by means of simulations that this degradation is very small.

C. Computational Complexity and Communication Requirements

We compare the computational complexity and communication requirements of the LC-DGPF and of its reduced-complexity variant discussed above (abbreviated R-LC-DGPF). We will disregard Steps 2 and 3 of the LC component (Algorithm 1), because their complexity and communication requirements are identical for the LC-DGPF and R-LC-DGPF; furthermore, their complexity is typically² much lower than that of the remaining steps (local GPF algorithm and LS approximation).

The complexity of the local GPF algorithm and of the LS approximation in the LC scheme (Step 1 of Algorithm 1) depends linearly on the number of particles [31], [38]. Thus, reducing the number of particles at each sensor from J to $J' = J/K$ reduces this complexity by a factor of K . It follows that the R-LC-DGPF is significantly less complex than

²The complexity of Steps 2 and 3 of Algorithm 1 is linear in the number of consensus algorithms and in the number of consensus iterations; these numbers depend on the specific application and setting.

the LC-DGPF. (The complexity of the additional consensus algorithms required by the R-LC-DGPF is typically negligible compared to the other operations.) The additional communication requirements of the R-LC-DGPF relative to the LC-DGPF are determined primarily by the speed of convergence (i.e., number of iterations I) of the additional consensus algorithms, which depends mainly on the second smallest eigenvalue of the Laplacian of the communication graph [39], and by the state dimension M . More specifically, the additional number of real numbers transmitted in the entire sensor network at each time n is KIN'_c , where $N'_c = M + M(M+1)/2 + 1$ is the number of additional consensus algorithms, i.e., of (scalar) consensus algorithms needed to calculate the mean vector and covariance matrix in (41) as well as the total weight in (42). Since N'_c is of order M^2 , the R-LC-DGPF has a disadvantage for high-dimensional states.

The reduced operation count of the R-LC-DGPF relative to the LC-DGPF can be exploited in two alternative ways, which represent a tradeoff between latency and power consumption. First, the processing time can be reduced; this results in a smaller latency of the R-LC-DGPF relative to the LC-DGPF, provided that the delays caused by the additional consensus algorithms are not too large. Thus, the R-LC-DGPF may be more suitable for real-time applications; however, the power consumption is higher due to the increased communications. Alternatively, if latency is not an issue, the processor's clock frequency can be reduced. The processing time can then be made equal to that of the LC-DGPF, while the processor's power consumption is reduced due to the lower clock frequency [40]. Thereby, the overall power consumption of the R-LC-DGPF is smaller relative to the LC-DGPF, provided that the additional power consumption due to the increased communications is not too large. However, the total latency is increased by the delays caused by the additional consensus algorithms.

VIII. NUMERICAL STUDY

We will now apply the proposed LC-based distributed PF algorithms to the problem of tracking multiple targets using acoustic amplitude sensors. We will compare the performance of our methods with that of the centralized PF and state-of-the-art distributed PFs.

A. Acoustic-Amplitude-Based Multiple Target Tracking

We consider P targets (P assumed known) moving independently in the x - y plane. The p th target, $p \in \{1, \dots, P\}$, is represented by the state vector $\mathbf{x}_n^{(p)} \triangleq (x_n^{(p)} \ y_n^{(p)} \ \dot{x}_n^{(p)} \ \dot{y}_n^{(p)})^\top$ containing the target's 2D position and 2D velocity. The overall state vector is defined as $\mathbf{x}_n \triangleq (\mathbf{x}_n^{(1)\top} \dots \mathbf{x}_n^{(P)\top})^\top$. Each vector $\mathbf{x}_n^{(p)}$ evolves independently of the other vectors $\mathbf{x}_n^{(p')}$ according to $\mathbf{x}_n^{(p)} = \mathbf{G}_p \mathbf{x}_{n-1}^{(p)} + \mathbf{W}_p \mathbf{u}_n^{(p)}$. Here, $\mathbf{u}_n^{(p)} \sim \mathcal{N}(\mathbf{0}_2, \sigma_u^2 \mathbf{I}_2)$ is Gaussian driving noise, with $\mathbf{u}_n^{(p)}$ and $\mathbf{u}_n^{(p')}$ independent unless $(n, p) = (n', p')$, and $\mathbf{G}_p \in \mathbb{R}^{4 \times 4}$ and $\mathbf{W}_p \in \mathbb{R}^{4 \times 2}$ are system matrices that will be specified in Section VIII-B. This model is commonly used in target tracking applications [18], [41]–[43]. It follows that the overall

state vector \mathbf{x}_n evolves according to

$$\mathbf{x}_n = \mathbf{G} \mathbf{x}_{n-1} + \mathbf{W} \mathbf{u}_n, \quad n = 1, 2, \dots,$$

where $\mathbf{G} \triangleq \text{diag}\{\mathbf{G}_1, \dots, \mathbf{G}_P\}$, $\mathbf{W} \triangleq \text{diag}\{\mathbf{W}_1, \dots, \mathbf{W}_P\}$, and $\mathbf{u}_n \triangleq (\mathbf{u}_n^{(1)\top} \dots \mathbf{u}_n^{(P)\top})^\top \sim \mathcal{N}(\mathbf{0}_{2P}, \sigma_u^2 \mathbf{I}_{2P})$.

Each target p emits a sound with a (root mean-squared) amplitude A_p that is assumed constant and known. At the position of sensor k , denoted $\boldsymbol{\xi}_{n,k}$, the sound amplitude due to target p is modeled as $A_p / \|\boldsymbol{\rho}_n^{(p)} - \boldsymbol{\xi}_{n,k}\|^\kappa$, where $\boldsymbol{\rho}_n^{(p)} \triangleq (x_n^{(p)} \ y_n^{(p)})^\top$ is the position of target p and κ is the path loss exponent [41], [44], [45]. The (scalar) measurement $z_{n,k}$ obtained by sensor k at time n is then given by

$$z_{n,k} = h_{n,k}(\mathbf{x}_n) + v_{n,k},$$

$$\text{with } h_{n,k}(\mathbf{x}_n) = \sum_{p=1}^P \frac{A_p}{\|\boldsymbol{\rho}_n^{(p)} - \boldsymbol{\xi}_{n,k}\|^\kappa}, \quad (43)$$

where $v_{n,k} \sim \mathcal{N}(0, \sigma_v^2)$ are zero-mean Gaussian measurement noise variables of equal variance σ_v^2 . 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 $(n, k) = (n', k')$. Note that this measurement model is a special instance of (20), and that $z_{n,k}$ does not depend on the velocities $\dot{x}_n^{(p)}$ and $\dot{y}_n^{(p)}$. The local likelihood functions and the JLF are respectively given by (cf. (21), (22))

$$f(z_{n,k} | \mathbf{x}_n) = \frac{1}{\sqrt{2\pi\sigma_v^2}} \exp\left(-\frac{1}{2\sigma_v^2} [z_{n,k} - h_{n,k}(\mathbf{x}_n)]^2\right) \quad (44)$$

$$f(\mathbf{z}_n | \mathbf{x}_n) = \frac{1}{\sqrt{(2\pi\sigma_v^2)^K}} \exp\left(-\frac{1}{2\sigma_v^2} \sum_{k=1}^K [z_{n,k} - h_{n,k}(\mathbf{x}_n)]^2\right),$$

and hence (cf. (25))

$$S_n(\mathbf{z}_n, \mathbf{x}_n) = \frac{1}{\sigma_v^2} \sum_{k=1}^K h_{n,k}(\mathbf{x}_n) \left[z_{n,k} - \frac{1}{2} h_{n,k}(\mathbf{x}_n) \right],$$

with $h_{n,k}(\mathbf{x}_n)$ given by (43).

In general, the sensor positions $\boldsymbol{\xi}_{n,k}$ are allowed to change with time n . (However, we used static sensors for simplicity.) Each sensor is supposed to know its own position but not the positions of the other sensors. The sensor positions (which are contained in the local likelihood functions) are implicitly fused by the LC method in the process of calculating the JLF; they need not be explicitly transmitted between the sensors. Therefore, the LC method and our LC-based distributed (G)PFs are well suited for dynamic sensor networks.

B. Simulation Setting

In our simulations, the number of targets is $P = 2$ unless stated otherwise. The system matrices \mathbf{G}_p and \mathbf{W}_p are identical for the two targets and given by [18]

$$\mathbf{G}_p = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad \mathbf{W}_p = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \\ 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad p = 1, 2.$$

The variance of the driving noises $\mathbf{u}_n^{(p)}$ is given by $\sigma_u^2 = 0.00035$. Each of the two targets emits a sound of equal

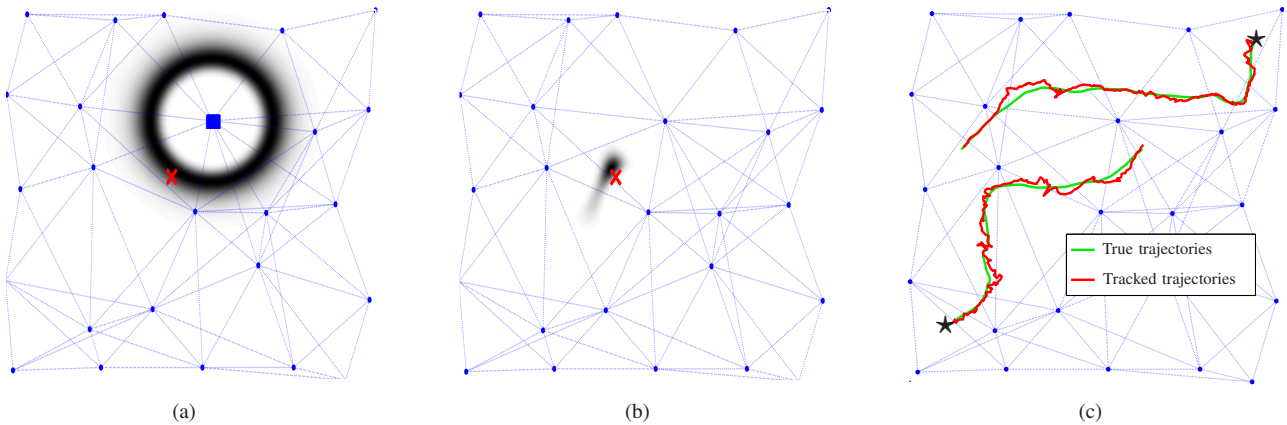


Fig. 1. Example of a sensor network and communication topology, along with (a) a local likelihood function for one target, (b) a JLF for one target, and (c) a realization of the trajectories of two targets and the corresponding trajectories tracked by the LC-DPF. In (a), the square indicates the sensor for which the local likelihood is depicted. In (a) and (b), darker shading represents higher likelihood values and the cross indicates the position of the target. In (c), the stars indicate the start points of the target trajectories.

amplitude $A_p = 10$. The initial prior pdf $f(\mathbf{x}_0^{(p)}) = \mathcal{N}(\boldsymbol{\mu}_0^{(p)}, \mathbf{C}_0)$ is different for the two targets, with $\boldsymbol{\mu}_0^{(1)} = (36 \ 36 \ -0.05 \ -0.05)^\top$ for target 1, $\boldsymbol{\mu}_0^{(2)} = (4 \ 4 \ 0.05 \ 0.05)^\top$ for target 2, and $\mathbf{C}_0 = \text{diag}\{1, 1, 0.001, 0.001\}$ for both targets.

The network consists of $K=25$ acoustic amplitude sensors that are deployed on a jittered grid within a rectangular region of size $40\text{m} \times 40\text{m}$. Each sensor communicates with other sensors within a range of 18m . The measurement noise variance is $\sigma_v^2 = 0.05$ and the path loss exponent is $\kappa = 1$.

For LC, we approximate the measurement function $h_{n,k}(\mathbf{x}_n)$ in (43) by a polynomial (see (29)) of degree $R_p=2$. This results in the following approximation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$ (cf. (32)):

$$\begin{aligned} \tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n) &= \sum_{k=1}^{25} \sum_{\mathbf{r}=\mathbf{0}}^4 \beta_{n,k,\mathbf{r}}(z_{n,k}) (x_n^{(1)})^{r_1} (y_n^{(1)})^{r_2} (x_n^{(2)})^{r_3} (y_n^{(2)})^{r_4}. \end{aligned}$$

To obtain the approximation coefficients $\alpha_{n,k,\mathbf{r}}$ needed for calculating the $\beta_{n,k,\mathbf{r}}(z_{n,k})$ according to (33) and (31), we use LS fitting as described in Section III-C. The sums over all sensors in (35) are computed by average consensus algorithms using Metropolis weights [36]. There are $N_c = \binom{4+4}{4} - 1 = 69$ consensus algorithms that are executed in parallel, each using $I = 8$ iterations unless noted otherwise. The same remarks apply to the sums in (41) and (42), which are required by the R-LC-DGPF. The number of additional consensus algorithms employed by the R-LC-DGPF is $N'_c = 8 + 8 \cdot 9/2 + 1 = 45$.

We compare the LC-DPF, LC-DGPF, R-LC-DGPF, CPF, and a centralized GPF (CGPF), which, similarly to the CPF, processes all sensor measurements at an FC. In addition, we consider the state-of-the-art consensus-based distributed PFs proposed (i) by Gu et al. in [21] (abbreviated GSHL-DPF), (ii) by Oreshkin and Coates in [22] (OC-DPF), and (iii) by Farahmand et al. in [19] (FRG-DPF). Unless stated otherwise, the number of particles at each sensor was $J = 5000$ for the LC-DPF, LC-DGPF, GSHL-DPF, and OC-DPF; $J = 2000$ for the FRG-DPF (this reduction is made possible by the adapted

proposal distribution); and $J' = 5000/25 = 200$ for the R-LC-DGPF. The PF at the FC of the CPF and CGPF employed 5000 particles. In the FRG-DPF [19], the rejection probability used for proposal adaptation was set to $\beta_k = 0.02$, and the oversampling factor was chosen as $L = 10$.

As a performance measure, we use the n -dependent root-mean-square error of the targets' position estimate $\hat{\boldsymbol{\rho}}_{n,k}$, denoted RMSE_n , which is computed as the square root of the average of $\|\hat{\boldsymbol{\rho}}_{n,k}^{(p)} - \boldsymbol{\rho}_n^{(p)}\|^2$ over the two targets $p = 1, 2$, all sensors $k = 1, \dots, 25$, and 5000 simulation runs. Here, $\boldsymbol{\rho}_n^{(p)}$ denotes the position of target p and $\hat{\boldsymbol{\rho}}_{n,k}^{(p)}$ denotes the corresponding estimate at sensor k . We also compute the *average RMSE* (ARMSE) as the square root of the average of RMSE_n^2 over all 200 simulated time instants n . Finally, we assess the error variation across the sensors k by the standard deviation σ_{ARMSE} of a k -dependent error defined as the square root of the average of $\|\hat{\boldsymbol{\rho}}_{n,k}^{(p)} - \boldsymbol{\rho}_n^{(p)}\|^2$ over the two targets $p = 1, 2$, all 200 time instants n , and 5000 simulation runs.

C. Simulation Results

Fig. 1 shows an example of a sensor network and communication topology. For the case of a single target ($P = 1$), examples of the local likelihood function and of the JLF are visualized in Fig. 1(a) and (b), respectively. The local likelihood function is circularly symmetric because the measurement function $h_{n,k}(\mathbf{x}_n)$ in (43) depends only on the distance between the target and the sensor. We can also see that the JLF is unimodal, which is an expected result since the JLF is the product of the local likelihood functions of all $K = 25$ sensors (see (1)), all having circularly symmetric shapes as shown in Fig. 1(a) but different locations due to the different local measurements and the different distances between target and sensor (see (44)). Furthermore, we note that the nonlinearity of the local measurement functions $h_{n,k}(\mathbf{x}_n)$ results in a non-Gaussian posterior (not shown in Fig. 1). For the case of two targets as described in Section VIII-B, Fig. 1(c) shows a realization of the target trajectories and the corresponding tracked trajectories that were obtained at one

	ARMSE [m]	Track loss adjusted ARMSE [m]	σ_{ARMSE} [m]	Track loss adjusted σ_{ARMSE} [m]	Track loss percentage [%]	Communication requirements
LC-DPF	0.6225	0.5424	0.0860	0.0222	0.95	13800
LC-DGPF	0.6187	0.5387	0.0889	0.0205	0.7	13800
R-LC-DGPF	0.5531	0.5204	0.0005	0.0005	0.46	22800
GSHL-DPF [21]	1.3022	1.2841	0.0032	0.0032	0.74	8800
OC-DPF [22]	0.9992	0.8399	0.0022	0.0024	1.1	8800
FRG-DPF [19]	0.5553	0.5335	0	0	0.2	400000
CPF	0.4975	0.4975	–	–	0	770
CGPF	0.5156	0.5086	–	–	0.18	770

TABLE I

ESTIMATION PERFORMANCE AND COMMUNICATION REQUIREMENTS OF THE PROPOSED CONSENSUS-BASED DISTRIBUTED PFs (LC-DPF, LC-DGPF, AND R-LC-DGPF), OF STATE-OF-THE-ART CONSENSUS-BASED DISTRIBUTED PFs (GSHL-DPF, OC-DPF, AND FRG-DPF), AND OF CENTRALIZED PFs (CPF AND CGPF).

specific sensor by means of the LC-DPF. It can be seen that the target is tracked fairly well. Other sensors obtained similar results.

Table I summarizes the estimation performance (ARMSE, track loss adjusted ARMSE, σ_{ARMSE} , track loss adjusted σ_{ARMSE} , and track loss percentage) and the communication requirements of the proposed consensus-based distributed PFs (LC-DPF, LC-DGPF, and R-LC-DGPF), of the state-of-the-art consensus-based distributed PFs (GSHL-DPF, OC-DPF, and FRG-DPF), and of the centralized methods (CPF and CGPF). The “track loss percentage” is defined as the percentage of simulation runs during which the estimation error at time $n = 200$ exceeded 5m, which is half the average inter-sensor distance. Such simulation runs were excluded in the calculation of the “track loss adjusted” RMSE_n, ARMSE, and σ_{ARMSE} . However, Table I presents also the ARMSE and σ_{ARMSE} computed using all the simulation runs (including those with lost tracks). The “communication requirements” are defined as the total number of real numbers transmitted (over one hop between neighboring sensors) at one time instant within the entire network. For the centralized methods (CPF and CGPF), we used multi-hop routing of measurements and sensor locations from every sensor to the FC (located in one of the corners of the network). Furthermore, the estimates calculated at the FC are disseminated throughout the network, such that every sensor obtains the centralized estimate.

It is seen from Table I that the track loss adjusted ARMSEs of the proposed distributed PFs are quite similar and that they are close to those of the centralized methods; they are slightly higher than that of FRG-DPF, slightly lower than that of OC-DPF, and about half that of GSHL-DPF. For FRG-DPF, σ_{ARMSE} is zero, since max and min consensus algorithms are employed to ensure identical results at each sensor. Furthermore, σ_{ARMSE} is higher for LC-DPF and LC-DGPF than for R-LC-DGPF, GSHL-DPF, and OC-DPF. This is because R-LC-DGPF, GSHL-DPF, and OC-DPF employ a consensus step whereby Gaussian approximations of the partial/local posterior pdfs are combined to obtain a global posterior, thus achieving a tighter coupling between the sensors. By contrast, the local PFs of LC-DPF and LC-DGPF operate completely independently; only the JLF is computed in a distributed way using the LC scheme. Note, however, that the ARMSE and track loss

adjusted ARMSE of LC-DPF and LC-DGPF are lower than for GSHL-DPF and OC-DPF. Finally, the track loss percentages of the proposed distributed PFs are below 1% and similar to those of GSHL-DPF, OC-DPF, and FRG-DPF. As a consequence, the ARMSEs are generally very close to the track loss adjusted ARMSEs.

The communication requirements of the distributed PFs are seen to be much higher than those of the centralized methods. This is due to our low-dimensional (scalar) measurements and the fact that each local likelihood function is parametrized only by the sensor location, i.e., three real numbers must be transmitted in one hop. For high-dimensional measurements and/or a different parametrization of the local likelihood functions, resulting in about 190 or more real numbers to be transmitted in one hop, the opposite will be true. Note that even when the consensus-based methods require more communications, they may be preferable over centralized methods because they are more robust (no possibility of FC failure), they require no routing protocols, and each sensor obtains an approximation of the global posterior (in the centralized schemes, each sensor obtains from the FC only the state estimate). It is furthermore seen that the communication requirements of the proposed distributed PFs are higher than those of GSHL-DPF and OC-DPF but much lower than those of FRG-DPF. Note, however, that the communication requirements of FRG-DPF depend on the number of particles and thus could be reduced by using fewer particles, whereas those of the other methods do not depend on the number of particles. (A setting with a lower number of particles will be considered later.) Finally, among the proposed distributed PFs, the communication requirements of R-LC-DGPF are higher by about 65% than those of LC-DPF and LC-DGPF.

In Fig. 2, we compare the RMSE_n and track loss adjusted RMSE_n of the proposed LC-DGPF with that of CGPF and the state-of-the-art distributed PFs (GSHL-DPF, OC-DPF, FRG-DPF). In terms of track loss adjusted RMSE_n (Fig. 2(b)), LC-DGPF outperforms GSHL-DPF and OC-DPF, and it performs almost as well as FRG-DPF and CGPF. The increase in RMSE_n over time in Fig. 2(a) is caused by the lost tracks.

In Fig. 3, we compare the RMSE_n and track loss adjusted RMSE_n of LC-DPF (using eight consensus iterations) with that of CPF. As a performance benchmark, we also show

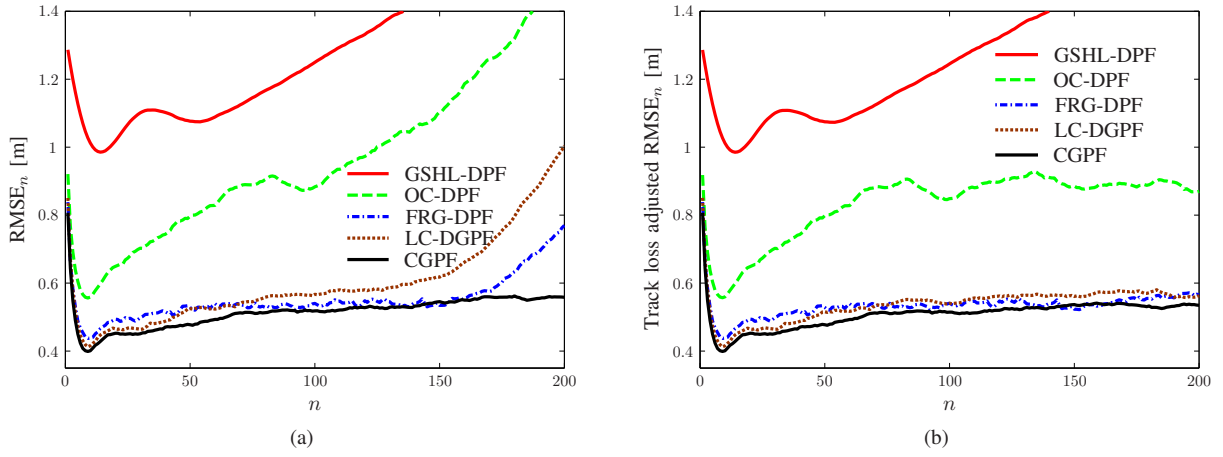


Fig. 2. (a) $RMSE_n$ and (b) track loss adjusted $RMSE_n$ versus time n for the proposed LC-DGPF, for the CGPF, and for state-of-the-art distributed PFs (GSHL-DPF, OC-DPF, and FRG-DPF). All distributed PFs use eight consensus iterations.

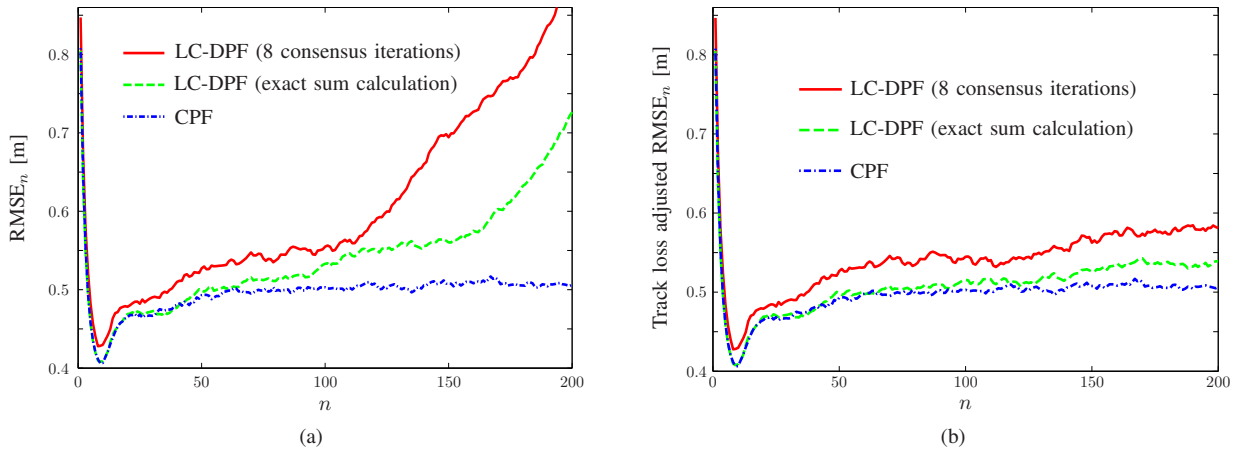


Fig. 3. (a) $RMSE_n$ and (b) track loss adjusted $RMSE_n$ versus time n for the CPF, for the proposed LC-DPF using eight consensus iterations, and for an impractical LC-DPF variant with exact sum calculation.

the results obtained by an impractical variant of LC-DPF in which the consensus algorithm is replaced by an exact, direct calculation of the sums in (35). The performance degradation of LC-DPF with exact sum calculation relative to CPF is caused by the LS approximation of the sensor measurement functions. The additional performance degradation of LC-DPF with eight consensus iterations relative to LC-DPF with exact sum calculation is due to the insufficiently converged consensus algorithms; it can be reduced by using more consensus iterations. In terms of the track loss adjusted $RMSE_n$, both performance degradations are seen to be quite moderate. The track loss percentages were 0.95% for LC-DPF, 0.29% for LC-DPF with exact sum calculation, and 0% for CPF.

Fig. 4 shows the track loss adjusted ARMSE of the proposed LC-DGPF and R-LC-DGPF versus the number I of consensus iterations. Here, R-LC-DGPF uses I consensus iterations in each one of its two consensus stages (i.e., I iterations to compute the sums in (35) and I iterations each to compute the sums in (41) and (42)). As a performance benchmark, the figure also shows the results for impractical variants of LC-DGPF and R-LC-DGPF using exact, direct calculation of the sums (35), (41), and (42). It is seen that the

performance of the impractical direct calculation is essentially achieved for I about 7 in the case of R-LC-DGPF and for I about 10 in the case of LC-DGPF. Somewhat surprisingly, R-LC-DGPF outperforms LC-DGPF for up to 10 consensus iterations, i.e., the additional consensus algorithms used to calculate the sums in (41) and (42) result in a better performance of R-LC-DGPF, in spite of the significantly reduced number of particles (200 instead of 5000). However, as the number of consensus iterations increases, both methods approach the performance of the respective “exact sum calculation” variant and LC-DGPF slightly outperforms R-LC-DGPF. This behavior can be explained as follows. The LC with a small number of consensus iterations is not completely converged, which means that the local information is not yet completely diffused throughout the network and the resulting approximate JLF does not yet contain the complete global information. The additional consensus stage of R-LC-DGPF then helps to further diffuse the local information.

Finally, we consider a setting where each sensor in the distributed PF methods (LC-DPF, LC-DGPF, GSHL-DPF, OC-DPF, and FRG-DPF) as well as the FC in CPF and CGPF use only $J = 400$ particles, and consequently R-LC-DGPF uses

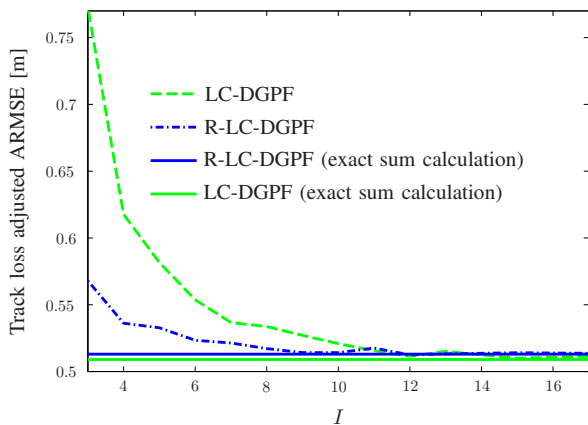


Fig. 4. Track loss adjusted ARMSE of the LC-DGPF and R-LC-DGPF versus the number I of consensus iterations, along with the track loss adjusted ARMSE of the impractical LC-DGPF and R-LC-DGPF variants with exact sum calculation. (R-LC-DGPF uses I consensus iterations for each sum calculation.)

only $J' = 400/25 = 16$ particles per sensor. This reduction of the number of particles results in reduced communication requirements of FRG-DPF but not of the other methods as their communication requirements are independent of the number of particles. Table II summarizes the simulation results we obtained. A comparison with Table I shows that, as expected, the performance of all methods is degraded. Furthermore, the high ARMSE and track loss percentage values of LC-DPF, LC-DGPF, and OC-DPF can be viewed as signs of divergence. In the case of LC-DPF and LC-DGPF, high σ_{ARMSE} values indicate significant differences between the local particle representations of the global posterior; these differences reduce the effectiveness of the LS approximation in the LC scheme. In the case of OC-DPF, the divergence is due to the peaky functions (powers of local likelihood functions) used in the weight update, which cause most of the particles to be located in regions of low likelihood. FRG-DPF performs well due to its use of adapted proposal distributions; its communication requirements are now closer to those of the other methods but still higher. R-LC-DGPF is seen to perform even slightly better with, at the same time, lower communication costs. As mentioned before, the additional consensus algorithms used by R-LC-DGPF lead to very similar particle representations of the local PFs across the network, with particles located in almost identical regions of the state space; this is evidenced by the low value of σ_{ARMSE} . Therefore, all sensors perform the LS approximation of their local likelihood functions in almost the same state space region, which moreover is the region where the particles of *all* sensors are located. Combining the local approximations using the LC scheme, we thus obtain a JLF approximation that is most accurate in that state space region. This explains the good tracking performance of R-LC-DGPF.

IX. CONCLUSION

For global estimation tasks in wireless sensor networks, the joint (all-sensors) likelihood function (JLF) plays a central role because it epitomizes the measurements of all sensors. We

proposed a distributed, consensus-based method for computing the JLF. This “likelihood consensus” method uses iterative consensus algorithms to compute, at each sensor, an approximation of the JLF as a function of the state to be estimated. Our method is applicable if the local likelihood functions of the various sensors (viewed as conditional probability density functions of the local measurements) belong to the exponential family of distributions. This includes the case of additive Gaussian measurement noises. The employed consensus algorithms require only local communications between neighboring sensors and operate without complex routing protocols.

We demonstrated the use of the likelihood consensus method for distributed particle filtering and distributed Gaussian particle filtering. At each sensor, a local particle filter computes a global state estimate that reflects the measurements of all sensors. The approximate JLF provided by the likelihood consensus method is used for updating the particle weights of each local particle filter. A second stage of consensus algorithms can be employed to significantly reduce the complexity of the distributed Gaussian particle filter. We applied the proposed distributed particle filters to a multiple target tracking problem and demonstrated experimentally that their performance is close to that of the centralized particle filters. Compared to three state-of-the-art distributed particle filtering schemes, our methods typically achieve a comparable or better estimation performance, while the communication requirements are somewhat higher in two cases and much lower in one case.

We finally note that the proposed distributed Gaussian particle filter can be extended to a consensus-based, distributed implementation of the Gaussian sum particle filter proposed in [46]. Furthermore, an extension of the likelihood consensus method to general local likelihood functions (i.e., not necessarily belonging to the exponential family) has been presented in [47].

REFERENCES

- [1] F. Zhao and L. J. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. Amsterdam, The Netherlands: Morgan Kaufmann, 2004.
- [2] S. Haykin and K. J. R. Liu, *Handbook on Array Processing and Sensor Networks*. Hoboken, NJ: Wiley, 2009.
- [3] G. Ferrari, *Sensor Networks: Where Theory Meets Practice*. Heidelberg, Germany: Springer, 2010.
- [4] A. Nayak and I. Stojmenović, *Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication*. Hoboken, NJ: Wiley, 2010.
- [5] T. Zhao and A. Nehorai, “Information-driven distributed maximum likelihood estimation based on Gauss-Newton method in wireless sensor networks,” *IEEE Trans. Signal Process.*, vol. 55, pp. 4669–4682, Sep. 2007.
- [6] T. Zhao and A. Nehorai, “Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks,” *IEEE Trans. Signal Process.*, vol. 55, pp. 1511–1524, Apr. 2007.
- [7] O. Hlinka and F. Hlawatsch, “Time-space-sequential algorithms for distributed Bayesian state estimation in serial sensor networks,” in *Proc. IEEE ICASSP-09*, Taipei, Taiwan, pp. 2057–2060, Apr. 2009.
- [8] R. G. Baraniuk, V. Delouille, and R. Neelamani, “Robust distributed estimation using the embedded subgraphs algorithm,” *IEEE Trans. Signal Process.*, vol. 54, pp. 2998–3010, Aug. 2006.
- [9] R. Olfati-Saber, “Distributed Kalman filter with embedded consensus filters,” in *Proc. 44th IEEE Conf. Decis. Contr.*, Seville, Spain, pp. 8179–8184, Dec. 2005.

	ARMSE [m]	Track loss adjusted ARMSE [m]	σ_{ARMSE} [m]	Track loss adjusted σ_{ARMSE} [m]	Track loss percentage [%]	Communication requirements
LC-DPF	3.3137	0.9651	0.1882	0.1523	26.34	13800
LC-DGPF	1.8692	0.6704	0.1074	0.0512	5.68	13800
R-LC-DGPF	0.7600	0.6556	0.0005	0.0004	0.71	22800
GSHL-DPF [21]	1.3371	1.3200	0.0032	0.0032	0.42	8800
OC-DPF [22]	2.9276	1.7746	0.0008	0.0015	28.50	8800
FRG-DPF [19]	0.8587	0.7162	0	0	1.01	80000
CPF	1.0552	0.7300	–	–	2.48	770
CGPF	0.6702	0.5879	–	–	0.44	770

TABLE II

SAME COMPARISON AS IN TABLE I, BUT USING ONLY $J = 400$ PARTICLES PER SENSOR FOR LC-DPF, LC-DGPF, GSHL-DPF, OC-DPF, AND FRG-DPF; $J' = 16$ PARTICLES PER SENSOR FOR R-LC-DGPF; AND 400 PARTICLES AT THE FUSION CENTER FOR CPF AND CGPF.

- [10] 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.
- [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] T. C. Aysal, M. E. Yildiz, A. D. Sarwate, and A. Scaglione, "Broadcast gossip algorithms for consensus," *IEEE Trans. Signal Process.*, vol. 57, pp. 2748–2761, Jul. 2009.
- [13] 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.
- [14] A. Doucet, N. De Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York, NY: Springer, 2001.
- [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] O. Hlinka, O. Slučiak, F. Hlawatsch, P. M. Djurić, and M. Rupp, "Likelihood consensus: Principles and application to distributed particle filtering," in *Proc. 44th Asilomar Conf. Sig., Syst., Comp.*, Pacific Grove, CA, pp. 349–353, Nov. 2010.
- [17] O. Hlinka, O. Slučiak, F. Hlawatsch, P. M. Djurić, and M. Rupp, "Distributed Gaussian particle filtering using likelihood consensus," in *Proc. IEEE ICASSP-11*, Prague, Czech Republic, pp. 3756–3759, May 2011.
- [18] J. H. Kotecha and P. M. Djurić, "Gaussian particle filtering," *IEEE Trans. Signal Process.*, vol. 51, pp. 2592–2601, Oct. 2003.
- [19] 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.
- [20] D. Gu, "Distributed particle filter for target tracking," in *Proc. IEEE ICRA-07*, Rome, Italy, pp. 3856–3861, Apr. 2007.
- [21] D. Gu, J. Sun, Z. Hu, and H. Li, "Consensus based distributed particle filter in sensor networks," in *Proc. IEEE ICIA-08*, Zhangjiajie, P. R. China, pp. 302–307, June 2008.
- [22] B. N. Oreshkin and M. J. Coates, "Asynchronous distributed particle filter via decentralized evaluation of Gaussian products," in *Proc. ISIF Int. Conf. Inform. Fusion*, Edinburgh, UK, pp. 1–8, Jul. 2010.
- [23] A. Mohammadi and A. Asif, "Consensus-based distributed unscented particle filter," in *Proc. IEEE SSP-11*, Nice, France, pp. 237–240, Jun. 2011.
- [24] F. Xaver, G. Matz, P. Gerstoft, and C. Mecklenbräuker, "Localization of acoustic sources using a decentralized particle filter," *EURASIP J. Wireless Commun. Networking*, vol. 2011, pp. 1–14, Sep. 2011.
- [25] M. Coates, "Distributed particle filters for sensor networks," in *Proc. ACM/IEEE IPSN-04*, Berkeley, CA, pp. 99–107, Apr. 2004.
- [26] X. Sheng, Y. H. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network," in *Proc. ACM/IEEE IPSN-05*, Los Angeles, CA, pp. 181–188, Apr. 2005.
- [27] O. Hlinka, P. M. Djurić, and F. Hlawatsch, "Time-space-sequential distributed particle filtering with low-rate communications," in *Proc. 43th Asilomar Conf. Sig., Syst., Comp.*, Pacific Grove, CA, pp. 196–200, Nov. 2009.
- [28] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Boston, MA: Artech House, 2004.
- [29] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ: Prentice-Hall, 1993.
- [30] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.
- [31] Å. Björck, *Numerical Methods for Least Squares Problems*. Philadelphia, PA: SIAM, 1996.
- [32] P. R. Bevington and D. K. Robinson, *Data Reduction and Error Analysis for the Physical Sciences*. New York, NY: McGraw-Hill, 2003.
- [33] G. Allaire and S. M. Kaber, *Numerical Linear Algebra*. New York, NY: Springer, 2008.
- [34] L. Georgopoulos and M. Hasler, "Nonlinear average consensus," in *Proc. NOLTA-09*, Sapporo, Japan, pp. 10–14, Oct. 2009.
- [35] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Contr. Letters*, vol. 53, no. 1, pp. 65–78, 2004.
- [36] L. Xiao, S. Boyd, and S. Lall, "A scheme for robust distributed sensor fusion based on average consensus," in *Proc. ACM/IEEE IPSN-05*, Los Angeles, CA, pp. 63–70, Apr. 2005.
- [37] D. Mosk-Aoyama and D. Shah, "Fast distributed algorithms for computing separable functions," *IEEE Trans. Inf. Theory*, vol. 54, pp. 2997–3007, Jul. 2008.
- [38] M. Bolić, A. Athalye, S. Hong, and P. M. Djurić, "Study of algorithmic and architectural characteristics of Gaussian particle filters," *J. Sig. Process. Syst.*, vol. 61, pp. 205–218, Nov. 2009.
- [39] M. Fiedler, "Algebraic connectivity of graphs," *Czech. Math. J.*, vol. 23, no. 2, pp. 298–305, 1973.
- [40] A. Wang and A. Chandrakasan, "Energy-efficient DSPs for wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 19, pp. 68–78, July 2002.
- [41] P. M. Djurić, M. Vemula, and M. F. Bugallo, "Target tracking by particle filtering in binary sensor networks," *IEEE Trans. Signal Process.*, vol. 56, pp. 2229–2238, Jun. 2008.
- [42] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. Hoboken, NJ: Wiley, 2001.
- [43] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Trans. Signal Process.*, vol. 50, pp. 425–437, Feb. 2002.
- [44] X. Sheng and Y. Hu, "Maximum likelihood multiple-source localization using acoustic energy measurements with wireless sensor networks," *IEEE Trans. Signal Process.*, vol. 53, pp. 44–53, Jan. 2004.
- [45] J. Liu, J. Reich, and F. Zhao, "Collaborative in-network processing for target tracking," *EURASIP J. Appl. Sig. Process.*, vol. 2003, pp. 378–391, Jan. 2003.
- [46] J. H. Kotecha and P. M. Djurić, "Gaussian sum particle filtering," *IEEE Trans. Signal Process.*, vol. 51, pp. 2602–2612, Oct. 2003.
- [47] 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.



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



Petar M. Djurić (S'86–M'90–SM'99–F'06) received his B.S. and M.S. degrees in electrical engineering from the University of Belgrade in 1981 and 1986, respectively, and his Ph.D. degree in electrical engineering from the University of Rhode Island in 1990.

From 1981 to 1986, he was a Research Associate with the Institute of Nuclear Sciences, Vinča, Belgrade. Since 1990, he has been with Stony Brook University, where he is currently a Professor in the Department of Electrical and Computer Engineering.

He works in the area of statistical signal processing, and his primary interests are in the theory of signal modeling, detection, and estimation and application of the theory to a wide range of disciplines.

Prof. Djurić has been invited to lecture at many universities in the United States and overseas. In 2007, he received the IEEE SIGNAL PROCESSING MAGAZINE Best Paper Award, and in 2008, he was elected Chair of Excellence of Universidad Carlos III de Madrid-Banco de Santander. During 2008–2009, he was Distinguished Lecturer of the IEEE Signal Processing Society. In 2012, he received the EURASIP Technical Achievement Award. He has served on numerous committees for the IEEE, and currently he is a Member-at-Large of the Board of Governors of the Signal Processing Society.



Ondrej Slučtiak (S'10) received the Ing. (M. Eng.) degree in electrical engineering from the Slovak University of Technology, Bratislava, Slovakia. Since 2008, he has been a Research Assistant with the Institute of Telecommunications, Vienna University of Technology, Vienna, Austria, where he is working toward the Ph.D. degree. His research interests include signal processing for wireless sensor networks, distributed consensus algorithms, and their convergence analysis.



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/ENSEEIH, Toulouse, France and IRCCyN, Nantes, France. He (co)authored a book, a review paper that appeared in the IEEE SIGNAL PROCESSING MAGAZINE, about 190 refereed scientific papers and book chapters, and three patents. He coedited three books. His research interests include signal processing for wireless communications and sensor networks, 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.



Markus Rupp (M'03–SM'06) received the Dipl.-Ing. degree at the University of Saarbrücken, Germany, in 1988 and the Dr.-Ing. degree at the Technische Universität Darmstadt, Germany, in 1993, where he worked with E. Hänslér on designing new algorithms for acoustical and electrical echo compensation. From November 1993 until July 1995, he had a postdoctoral position at the University of Santa Barbara, Santa Barbara, CA, with S. Mitra, where he worked with A. H. Sayed on a robustness description of adaptive filters with impact on neural

networks and active noise control. From October 1995 until August 2001, he was a member of Technical Staff in the Wireless Technology Research Department of Bell Labs, Crawford Hill, NJ, where he worked on various topics related to adaptive equalization and rapid implementation for IS-136, 802.11 and UMTS, including the first MIMO prototype for UMTS. Since October 2001, he has been a Full Professor for Digital Signal Processing in Mobile Communications at the Vienna University of Technology, Vienna, Austria, where he founded the Christian-Doppler Laboratory for Design Methodology of Signal Processing Algorithms in 2002 at the Institute for Communications and Radio-Frequency Engineering. He served as Dean during 2005–2007. He authored and coauthored more than 400 scientific papers and patents on adaptive filtering, wireless communications, and rapid prototyping, as well as automatic design methods.

Prof. Rupp was Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING during 2002–2005, and is currently Associate Editor of the *EURASIP Journal of Advances in Signal Processing* and the *EURASIP Journal on Embedded Systems*. He has been an elected AdCom member of EURASIP since 2004 and served as President of EURASIP during 2009–2010.