in Proc. 44th Asilomar Conf. Signals, Systems, Computers,
Pacific Grove, CA, Nov. 2010

Likelihood Consensus: Principles and Application to Distributed Particle Filtering

Ondrej Hlinka[a], *Ondrej Slučiak*[a], *Franz Hlawatsch*[a], *Petar M. Djurić*[b], and *Markus Rupp*[a]

[a]Institute of Communications and Radio-Frequency Engineering, Vienna University of Technology (ohlinka@nt.tuwien.ac.at)
[b]Department of Electrical and Computer Engineering, Stony Brook University (djuric@ece.sunysb.edu)

***Abstract*—We propose a distributed method for computing the joint (all-sensors) likelihood function (JLF) in a wireless sensor network. A consensus algorithm is used for a decentralized, iterative calculation of a sufficient statistic that describes an approximation to the JLF. After convergence of the consensus algorithm, the approximate JLF—which epitomizes the measurements of all sensors—is available at each sensor. This "likelihood consensus" method requires only communications between neighboring sensors. We implement the likelihood consensus method in a distributed particle filtering scheme. Each sensor runs a local particle filter that computes a global state estimate. The updating of the particle weights of each local particle filter uses the JLF. The performance of this distributed particle filter is demonstrated on a target tracking problem.**

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

I. Introduction

We consider a wireless sensor network (WSN) without a fusion center. The WSN performs a global statistical inference task through local processing and communications, in a way such that the final inference result is available at each sensor node. This fully decentralized approach is needed in certain applications (e.g., sensor-actuator networks [1]), and it is desirable for its robustness, scalability, and flexibility in deployment.

In many inference tasks, the joint (all-sensors) likelihood function (JLF) plays a central role. The JLF epitomizes the measurements of all sensors. For example, a particle filter (PF), which processes all the sensor measurements, relies on the pointwise evaluation of the JLF to perform its weight update.

In this paper, we propose a distributed method for calculating the JLF or an approximation to the JLF at each sensor. Our method is suited for sensors with (generally) nonlinear measurement functions and additive Gaussian measurement noise. A consensus algorithm 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. After convergence of the consensus algorithm, the JLF is available at each sensor. Consequently, we refer to our method as *likelihood consensus* (LC). An important advantage of LC is that it requires communication only between neighboring sensors and operates without routing protocols.

We furthermore propose an application of the LC method in a distributed PF (DPF) scheme. Each sensor runs a local PF (LPF) that computes a global state estimate. At any given PF recursion, each LPF draws a set of particles and updates their weights by evaluating the JLF at these particles. An alternative consensus-based DPF scheme was introduced in [2]. However, the number of consensus algorithms that need to be executed in parallel, and hence the amount of communication, can be significantly higher than in our approach. Other consensus-based DPFs, proposed in [3] and [4], rely on Gaussian or Gaussian mixture approximations of the filtering distribution and use only the local likelihood functions instead of the JLF.

This paper is organized as follows. After a description of the system model in Section II, the LC method is presented in Section III. The application of LC to distributed particle filtering is considered in Section IV. Finally, Section V presents simulation results for a target tracking problem.

II. System Model

We consider a WSN consisting of K sensors. At a given time, the WSN performs some type of statistical inference (estimation, detection, or classification) related to an M-dimensional state vector $\mathbf{x} = (x_1 \cdots x_M)^\top \in \mathbb{R}^M$. Let the N_k-dimensional vector $\mathbf{z}_k \in \mathbb{R}^{N_k}$ denote the measurement acquired by the kth sensor ($k \in \{1,\dots,K\}$), and let $\mathbf{z} \triangleq (\mathbf{z}_1^\top \cdots \mathbf{z}_K^\top)^\top$ contain the measurements of all sensors. The statistical relationship between $\mathbf{z}_k$ and $\mathbf{x}$ is described by the *local* likelihood function[1] $f(\mathbf{z}_k|\mathbf{x})$, and the statistical relationship between $\mathbf{z}$ and $\mathbf{x}$ is described by the JLF $f(\mathbf{z}|\mathbf{x})$.

We consider the measurement model

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}) + \mathbf{w}_k\,, \quad k = 1,\dots,K\,, \tag{1}$$

where $\mathbf{h}_k(\cdot)$ is the measurement function of sensor k and $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0},\mathbf{Q}_k)$ is zero-mean Gaussian measurement noise. We assume that the noise vectors $\mathbf{w}_k$ at different sensors k are independent, and that $\mathbf{h}_k(\cdot)$ and $\mathbf{Q}_k$ are known to sensor k.

This work was supported by the FWF under Awards S10603 and S10609 within the National Research Network SISE, by the NSF under Award CCF-1018323, and by the ONR under Award N00014-09-1-1154.

[1] The notation $f(\mathbf{z}_k|\mathbf{x})$ suggests that $\mathbf{x}$ is a random vector. However, for the LC method, $\mathbf{x}$ is also allowed to be deterministic, in which case the notation $f(\mathbf{z}_k;\mathbf{x})$ would be more appropriate. Also, $\mathbf{x}$ may be time-varying, but we omit the time index in this section (it will be needed only in Section IV).

From (1) and the Gaussianity of $\mathbf{w}_k$, the local likelihood function of sensor k follows as

$$f(\mathbf{z}_k|\mathbf{x}) \,=\, C_k \exp\!\bigg(\!-\frac{1}{2}\,(\mathbf{z}_k - \mathbf{h}_k(\mathbf{x}))^\top \mathbf{Q}_k^{-1}(\mathbf{z}_k - \mathbf{h}_k(\mathbf{x}))\bigg)\,,$$

with $C_k \triangleq [(2\pi)^{N_k} \det\{\mathbf{Q}_k\}]^{-1/2}$. Due to the independence of all $\mathbf{w}_k$, we then obtain the JLF as

$$f(\mathbf{z}|\mathbf{x}) \,=\, \prod_{k=1}^{K} f(\mathbf{z}_k|\mathbf{x}) \,=\, C \exp\!\bigg(\!-\frac{1}{2} S(\mathbf{z},\mathbf{x})\bigg), \qquad (2)$$

where $C \triangleq \prod_{k=1}^{K} C_k$ and

$$S(\mathbf{z},\mathbf{x}) \,\triangleq\, \sum_{k=1}^{K} (\mathbf{z}_k - \mathbf{h}_k(\mathbf{x}))^\top \mathbf{Q}_k^{-1}(\mathbf{z}_k - \mathbf{h}_k(\mathbf{x}))\,. \qquad (3)$$

Evidently, the calculation of the JLF $f(\mathbf{z}|\mathbf{x})$ (up to the factor C, which is irrelevant as it does not depend on $\mathbf{x}$) reduces to calculation of $S(\mathbf{z},\mathbf{x})$. Straightforward evaluation of $S(\mathbf{z},\mathbf{x})$ at a given sensor according to (3) presupposes that the sensor knows the measurements $\mathbf{z}_k$, the measurement functions $\mathbf{h}_k(\cdot)$, and the noise covariances $\mathbf{Q}_k$ of *all* sensors. The amount of inter-sensor communications required to provide all this information to all sensors will often be prohibitively high; furthermore, complex routing protocols may be required.

III. Likelihood Consensus

We will now show how a consensus algorithm can be used for a distributed approximate calculation of $S(\mathbf{z},\mathbf{x})$ that requires only low-rate communication between neighboring sensors.

A. Polynomial approximation

Let us approximate the measurement function $\mathbf{h}_k(\mathbf{x})$ by a multivariate vector-valued polynomial in $\mathbf{x}$ of degree R:

$$\mathbf{h}_k(\mathbf{x}) \,\approx\, \tilde{\mathbf{h}}_k(\mathbf{x}) \,\triangleq\, \sum_{\mathbf{r}=\mathbf{0}}^{R} \boldsymbol{\alpha}_{\mathbf{r},k}\, p_{\mathbf{r}}(\mathbf{x})\,, \quad \text{with } p_{\mathbf{r}}(\mathbf{x}) \triangleq \prod_{m=1}^{M} x_m^{r_m}\,. \qquad (4)$$

Here, $\mathbf{r} \triangleq (r_1 \cdots r_M) \in \{0,\ldots,R\}^M$; $\boldsymbol{\alpha}_{\mathbf{r},k} \in \mathbb{R}^{N_k}$ is the coefficient vector associated with the monomial $p_{\mathbf{r}}(\mathbf{x})$; and $\sum_{\mathbf{r}=\mathbf{0}}^{R}$ is short for $\sum_{r_1=0}^{R} \cdots \sum_{r_M=0}^{R}$ with the constraint $\sum_{m=1}^{M} r_m \leq R$. Such a polynomial approximation[2] is given, e.g., by a truncated Taylor series expansion of $\mathbf{h}_k(\mathbf{x})$ about some point $\mathbf{x}'_k$. (The choice of $\mathbf{x}'_k$ will be discussed in Section IV-D.) Substituting $\tilde{\mathbf{h}}_k(\mathbf{x})$ for $\mathbf{h}_k(\mathbf{x})$ in (3) yields the following approximation to $S(\mathbf{z},\mathbf{x})$:

$$\tilde{S}(\mathbf{z},\mathbf{x}) \,\triangleq\, \sum_{k=1}^{K} (\mathbf{z}_k - \tilde{\mathbf{h}}_k(\mathbf{x}))^\top \mathbf{Q}_k^{-1}(\mathbf{z}_k - \tilde{\mathbf{h}}_k(\mathbf{x}))\,. \qquad (5)$$

Via (2), this corresponds to an approximation to the JLF given by $\tilde{f}(\mathbf{z}|\mathbf{x}) \triangleq C \exp\!\big(-\frac{1}{2}\tilde{S}(\mathbf{z},\mathbf{x})\big)$.

The function $\tilde{S}(\mathbf{z},\mathbf{x})$ is itself a polynomial in $\mathbf{x}$. Indeed, using the decomposition[3] $\mathbf{Q}_k^{-1} = \mathbf{A}_k \mathbf{A}_k^\top$, we can rewrite (5) as

$$\tilde{S}(\mathbf{z},\mathbf{x}) \,=\, \sum_{k=1}^{K} \big\|\mathbf{A}_k^\top(\mathbf{z}_k - \tilde{\mathbf{h}}_k(\mathbf{x}))\big\|^2\,. \qquad (6)$$

Inserting (4) into (6) and evaluating the squared norm, we obtain

$$\tilde{S}(\mathbf{z},\mathbf{x}) \,=\, \sum_{k=1}^{K} \Bigg[\sum_{\mathbf{r}=\mathbf{0}}^{2R} \beta_{\mathbf{r},k}(\mathbf{z}_k)\, p_{\mathbf{r}}(\mathbf{x})\Bigg]\,, \qquad (7)$$

where the coefficients $\beta_{\mathbf{r},k}(\mathbf{z}_k)$ are defined for all $\mathbf{r} \in \{0,\ldots,2R\}^M$ such that $\sum_{m=1}^{M} r_m \leq 2R$. These coefficients depend on $\mathbf{z}_k$, $\boldsymbol{\alpha}_{\mathbf{r},k}$, and $\mathbf{Q}_k$ according to

$$\beta_{\mathbf{r},k}(\mathbf{z}_k) \,=\, \sum_{l=1}^{N_k} \left(\sum_{\mathbf{j}=\mathbf{0}}^{R} \bar{\alpha}_{\mathbf{j},k}^{(l)}\, \bar{\alpha}_{\mathbf{r}-\mathbf{j},k}^{(l)}\right)\,, \qquad (8)$$

where $\mathbf{j} \triangleq (j_1 \cdots j_M)$ and $\bar{\alpha}_{\mathbf{r},k}^{(l)}$ denotes the lth entry of the vector $\bar{\boldsymbol{\alpha}}_{\mathbf{r},k}$ defined as

$$\bar{\boldsymbol{\alpha}}_{\mathbf{r},k} \,\triangleq\, \begin{cases} \mathbf{A}_k^\top(\boldsymbol{\alpha}_{\mathbf{r},k} - \mathbf{z}_k)\,, & \mathbf{r} = \mathbf{0}\,, \\ \mathbf{A}_k^\top \boldsymbol{\alpha}_{\mathbf{r},k}\,, & 0 < \sum_{m=1}^{M} r_m \leq R \\ & \text{and } r_1,\ldots,r_M \geq 0\,, \\ \mathbf{0}\,, & \text{else}\,. \end{cases}$$

By changing the order of summation in (7), we finally obtain

$$\tilde{S}(\mathbf{z},\mathbf{x}) \,=\, \sum_{\mathbf{r}=\mathbf{0}}^{2R} T_{\mathbf{r}}(\mathbf{z})\, p_{\mathbf{r}}(\mathbf{x})\,,$$

which is a polynomial in $\mathbf{x}$ with coefficients

$$T_{\mathbf{r}}(\mathbf{z}) \,=\, \sum_{k=1}^{K} \beta_{\mathbf{r},k}(\mathbf{z}_k)\,. \qquad (9)$$

B. Distributed computation of the approximate JLF

The vector $\mathbf{T}(\mathbf{z}) \triangleq (T_{\mathbf{r}}(\mathbf{z}))$ of all coefficients $T_{\mathbf{r}}(\mathbf{z})$ for $\sum_{m=1}^{M} r_m \leq 2R$ can be viewed as a *sufficient statistic* [6] that fully describes $\tilde{S}(\mathbf{z},\mathbf{x})$ and, in turn, the approximate JLF

$$\tilde{f}(\mathbf{z}|\mathbf{x}) \,\propto\, \exp\!\bigg(\!-\frac{1}{2}\tilde{S}(\mathbf{z},\mathbf{x})\bigg) \,=\, \exp\!\bigg(\!-\frac{1}{2}\sum_{\mathbf{r}=\mathbf{0}}^{2R} T_{\mathbf{r}}(\mathbf{z})\, p_{\mathbf{r}}(\mathbf{x})\bigg), \qquad (10)$$

for all $\mathbf{x}$. Thus, a sensor that knows $\mathbf{T}(\mathbf{z})$ is able to evaluate $\tilde{f}(\mathbf{z}|\mathbf{x})$ for any value of the state $\mathbf{x}$.

The sum in (9) defining $\mathbf{T}(\mathbf{z}) = (T_{\mathbf{r}}(\mathbf{z}))$ can be computed at each sensor by means of a distributed, iterative consensus algorithm that requires only communication between neighboring sensors. We will here use a *linear* consensus algorithm [7] for simplicity; however, other consensus algorithms could be used as well (e.g. [8]). In what follows, the superscript $^{(i)}$ denotes

[2] It is possible to obtain a more general approximation of the measurement function by a linear combination of basis functions that are different from the monomials $p_{\mathbf{r}}(\mathbf{x})$ used in (4). As long as these basis functions do not depend on the sensor index k, our method can be used with certain modifications.

[3] This decomposition exists since $\mathbf{Q}_k^{-1}$ is positive definite. The matrix $\mathbf{A}_k \in \mathbb{R}^{N_k \times N_k}$ is uniquely defined up to an arbitrary unitary matrix factor. The Cholesky decomposition [5] is an example.

the iteration index and $\mathcal{N}_k \subseteq \{1,\ldots,K\}\backslash\{k\}$ denotes a fixed set of sensors that are neighbors of sensor k.

To compute $T_{\mathbf{r}}(\mathbf{z}) = \sum_{k'=1}^{K} \beta_{\mathbf{r},k'}(\mathbf{z}_{k'})$, sensor k first initializes its local "state" as $T_{\mathbf{r},k}^{(0)} = \beta_{\mathbf{r},k}(\mathbf{z}_k)$, where $\beta_{\mathbf{r},k}(\mathbf{z}_k)$ is calculated using (8). This calculation only involves the measurement $\mathbf{z}_k$, the polynomial coefficients $\boldsymbol{\alpha}_{\mathbf{r},k}$ (see (4)), and the noise covariance $\mathbf{Q}_k$, *all of which are available at sensor* k; thus, no communication is required at this initialization stage. Then, at the ith iteration of the consensus algorithm ($i=1,2,\ldots$), the following steps are performed by sensor k.

- The past states $T_{\mathbf{r},k'}^{(i-1)}$ of all neighbor sensors $k' \in \mathcal{N}_k$ are received. Together with the past local state, $T_{\mathbf{r},k}^{(i-1)}$, they are used to update the local state according to
$$T_{\mathbf{r},k}^{(i)} = \omega_{k,k}^{(i)} T_{\mathbf{r},k}^{(i-1)} + \sum_{k' \in \mathcal{N}_k} \omega_{k,k'}^{(i)} T_{\mathbf{r},k'}^{(i-1)} .$$
The choice of the weights $\omega_{k,k'}^{(i)}$ is discussed in, e.g., [9], [10]. Here, we use the Metropolis weights [10] given by
$$\omega_{k,k'}^{(i)} = \omega_{k,k'} = \begin{cases} \frac{1}{1+\max\{|\mathcal{N}_k|,|\mathcal{N}_{k'}|\}}\,, & k \neq k'\,, \\ 1 - \sum_{k'' \in \mathcal{N}_k} \omega_{k,k''}\,, & k = k'\,, \end{cases} \quad (11)$$
where $|\mathcal{N}_k|$ denotes the number of neighbors of sensor k.
- The new local state $T_{\mathbf{r},k}^{(i)}$ is broadcasted to all neighbors $k' \in \mathcal{N}_k$.

These two steps are repeated in an iterative manner. Under conditions specified in [7], $T_{\mathbf{r},k}^{(i)}$ asymptotically converges to the average $\frac{1}{K}\sum_{k'=1}^{K} \beta_{\mathbf{r},k'}(\mathbf{z}_{k'}) = \frac{1}{K} T_{\mathbf{r}}(\mathbf{z})$ as $i \to \infty$. Thus, for i sufficiently large, $T_{\mathbf{r}}(\mathbf{z}) \approx K T_{\mathbf{r},k}^{(i)}$.

This algorithm is executed for each component $T_{\mathbf{r}}(\mathbf{z})$ of $\mathbf{T}(\mathbf{z})$, i.e., for each $\mathbf{r}$ such that $\sum_{m=1}^{M} r_m \leq 2R$, except $\mathbf{r}=\mathbf{0}$. (According to (10), $T_{\mathbf{0}}(\mathbf{z})$ corresponds to a JLF factor that does not depend on $\mathbf{x}$ and is hence irrelevant.) Thus, the number of consensus algorithms is $N_c = \binom{2R+M}{2R} - 1$; this is also the number of real numbers broadcasted by each sensor.

C. Distributed computation of the exact JLF

The polynomial approximation used in the previous section can be avoided if the JLF $f(\mathbf{z}|\mathbf{x})$ has a special structure. In that case, the *exact* JLF can be computed in a distributed way (up to approximation errors due to the limited number of consensus iterations performed). Consider a sufficient statistic $\widetilde{\mathbf{T}}(\mathbf{z}) = \big(\widetilde{T}_1(\mathbf{z}) \cdots \widetilde{T}_P(\mathbf{z})\big)^{\top}$, where the $\widetilde{T}_p(\mathbf{z})$ denote the individual entries of vector $\widetilde{\mathbf{T}}(\mathbf{z})$. According to the Neyman-Fisher factorization theorem [6], $f(\mathbf{z}|\mathbf{x})$ can be written as
$$f(\mathbf{z}|\mathbf{x}) = f_1(\mathbf{z})\, f_2(\widetilde{\mathbf{T}}(\mathbf{z}), \mathbf{x})\,.$$
The factor $f_1(\mathbf{z})$ can be disregarded since it does not depend on $\mathbf{x}$. It follows that $\widetilde{\mathbf{T}}(\mathbf{z})$ epitomizes the total measurement $\mathbf{z}$, in the sense that a sensor that knows $\widetilde{\mathbf{T}}(\mathbf{z})$ and the functional form of $f_2(\widetilde{\mathbf{T}}(\mathbf{z}), \mathbf{x})$ is able to evaluate $f(\mathbf{z}|\mathbf{x})$ (up to an irrelevant factor) for any value of $\mathbf{x}$.

Suppose now that the components of $\widetilde{\mathbf{T}}(\mathbf{z})$ have the form
$$\widetilde{T}_p(\mathbf{z}) = \sum_{k=1}^{K} \phi_{p,k}(\mathbf{z}_k)\,, \quad p = 1,\ldots,P\,, \quad (12)$$
with arbitrary functions $\phi_{p,k}(\cdot)$, and that sensor k knows the functions $\phi_{p,k}(\cdot)$ (but not the other functions $\phi_{p,k'}(\cdot)$, $k' \neq k$). Based on the sum expression (12), we can then use a consensus algorithm as described in Section III-B, with obvious modifications, to evaluate $\widetilde{\mathbf{T}}(\mathbf{z})$ and, thus, the JLF $f(\mathbf{z}|\mathbf{x})$.

IV. Distributed Particle Filtering

We next show how the LC method presented in the previous section can be applied to obtain a DPF.

A. State-space model

We consider a nonlinear, non-Gaussian dynamic system with additive Gaussian measurement noises. The random state vector $\mathbf{x}_n = (x_{n,1} \cdots x_{n,M})^{\top}$ evolves with discrete time n according to the state-transition equation
$$\mathbf{x}_n = \mathbf{g}_n(\mathbf{x}_{n-1}, \mathbf{u}_n)\,, \quad n = 1,2,\ldots\,, \quad (13)$$
where $\mathbf{u}_n$ is a temporally white driving noise with a known probability density function (pdf) $f(\mathbf{u}_n)$. A WSN with K sensors acquires measurements according to the measurement equations (cf. (1))
$$\mathbf{z}_{n,k} = \mathbf{h}_{n,k}(\mathbf{x}_n) + \mathbf{w}_{n,k}\,, \quad k = 1,2,\ldots,K\,. \quad (14)$$
Here, $\mathbf{z}_{n,k} \in \mathbb{R}^{N_{n,k}}$ is the measurement vector at time n and at sensor k, and $\mathbf{w}_{n,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{n,k})$ is zero-mean Gaussian measurement noise. We assume that (i) $\mathbf{w}_{n,k}$ and $\mathbf{w}_{n',k'}$ are independent for $n \neq n'$ and/or $k \neq k'$; (ii) the initial state $\mathbf{x}_0$ and the sequences $\mathbf{u}_n$ and $\mathbf{w}_{n,k}$ are all independent; and (iii) sensor k knows $\mathbf{g}_n(\cdot,\cdot)$, $\mathbf{h}_{n,k}(\cdot)$, and $\mathbf{Q}_{n,k}$ for all n but not $\mathbf{h}_{n,k'}(\cdot)$ and $\mathbf{Q}_{n,k'}$ for $k' \neq k$. In what follows, $\mathbf{z}_n \triangleq (\mathbf{z}_{n,1}^{\top} \cdots \mathbf{z}_{n,K}^{\top})^{\top}$ denotes the vector containing the measurements of all sensors at time n, and $\mathbf{z}_{1:n} \triangleq (\mathbf{z}_1^{\top} \cdots \mathbf{z}_n^{\top})^{\top}$ denotes the vector of all measurements up to time n.

Equations (13) and (14) together with our statistical assumptions determine the state-transition pdf $f(\mathbf{x}_n|\mathbf{x}_{n-1})$, the local likelihood function $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$, and the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$. In particular, the JLF is given by (2) and (3) with an additional time index, i.e., $f(\mathbf{z}_n|\mathbf{x}_n) \propto \exp\!\big(-\frac{1}{2} S_n(\mathbf{z}_n, \mathbf{x}_n)\big)$ with
$$S_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{k=1}^{K} (\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n))^{\top} \mathbf{Q}_{n,k}^{-1} (\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n))\,.$$
Based on this expression, we can again use LC to compute (an approximation to) the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ at each sensor k, as explained in Section III. This has to be repeated for each n.

B. Sequential MMSE estimation

We now consider distributed estimation of the state $\mathbf{x}_n$ from the past and present measurements, $\mathbf{z}_{1:n}$, using the minimum mean-square error (MMSE) estimator

$$\hat{\mathbf{x}}_n \triangleq \mathrm{E}\{\mathbf{x}_n|\mathbf{z}_{1:n}\} = \int \mathbf{x}_n\, f(\mathbf{x}_n|\mathbf{z}_{1:n})\, d\mathbf{x}_n\,. \tag{15}$$

Based on the state-space model as expressed by the state-transition pdf $f(\mathbf{x}_n|\mathbf{x}_{n-1})$, the current posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ involved in (15) can be calculated 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 temporal recursion [11]

$$f(\mathbf{x}_n|\mathbf{z}_{1:n}) = \frac{f(\mathbf{z}_n|\mathbf{x}_n)\int f(\mathbf{x}_n|\mathbf{x}_{n-1})f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1})\,d\mathbf{x}_{n-1}}{f(\mathbf{z}_n|\mathbf{z}_{1:n-1})}\,. \tag{16}$$

However, for the nonlinear/non-Gaussian case considered, the computational complexity of sequential MMSE state estimation as given by (15) and (16) is typically prohibitive. A computationally feasible approximation is provided by the PF [11], [12]. In a PF, the (non-Gaussian) posterior is represented by a set of samples (or particles) $\mathbf{x}_n^{(j)}$ and corresponding weights $w_n^{(j)}$.

C. Distributed particle filtering using likelihood consensus

Let us consider a DPF implementation of the MMSE estimator (15) within a WSN. Each sensor node sequentially tracks the global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ using an LPF and, for each n, 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 $\mathbf{x}_n$. This knowledge is provided by the LC method.

At a given time instant n, the LPF at sensor k performs the following steps, which are identical for all k.

Step 1: At the previous time instant $n-1$, sensor k calculated a set of particles and weights $\{\mathbf{x}_{n-1,k}^{(j)}, w_{n-1,k}^{(j)}\}_{j=1}^{J}$, which represents the previous 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 $\bar{\mathbf{x}}_{n-1,k}^{(j)}$ and identical weights $\bar{w}_{n-1,k}^{(j)} \equiv 1/J$ (see, e.g., [11], [12] for details).

Step 2: For each $\bar{\mathbf{x}}_{n-1,k}^{(j)}$, a new, "predicted" particle $\mathbf{x}_{n,k}^{(j)}$ is randomly drawn from $f(\mathbf{x}_n|\mathbf{x}_{n-1})\big|_{\mathbf{x}_{n-1}=\bar{\mathbf{x}}_{n-1,k}^{(j)}}$.

Step 3: The JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ (or an approximation to $f(\mathbf{z}_n|\mathbf{x}_n)$) is computed by means of LC as described in Section III. This step requires communication with neighboring sensors.

Step 4: The weights associated with the predicted particles $\mathbf{x}_{n,k}^{(j)}$ drawn in Step 2 are calculated according to [11], [12]

$$w_{n,k}^{(j)} = \frac{f(\mathbf{z}_n|\mathbf{x}_{n,k}^{(j)})}{\sum_{j'=1}^{J} f(\mathbf{z}_n|\mathbf{x}_{n,k}^{(j')})}\,. \tag{17}$$

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

Step 5: From the weighted particles $\{\mathbf{x}_{n,k}^{(j)}, w_{n,k}^{(j)}\}_{j=1}^{J}$, an approximation to the MMSE state estimate (15) is computed as

$$\hat{\mathbf{x}}_{n,k} = \sum_{j=1}^{J} w_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)}\,. \tag{18}$$

The recursion defined by Steps 1–5 is initialized at $n=0$ by J particles $\mathbf{x}_{0,k}^{(j)}$ randomly drawn from a suitable prior pdf and equal weights $w_{0,k}^{(j)} \equiv 1/J$. As can be seen, each sensor tracks (a particle representation of) the posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$, using at each time n the current JLF that involves the measurements of all sensors. Thus, each sensor performs operations that are normally possible only at a fusion center after collecting measurements from the entire WSN. 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 introduced by insufficiently converged consensus algorithms.

D. Choosing the reference points

We assume that, at sensor k, $\tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n)$ in (4) is a truncated Taylor series expansion of $\mathbf{h}_{n,k}(\mathbf{x}_n)$ about some reference point $\mathbf{x}'_{n,k}$. Evidently, $\mathbf{x}'_{n,k}$ should be chosen such that $\tilde{\mathbf{h}}_{n,k}(\mathbf{x}_n)$ is close to $\mathbf{h}_k(\mathbf{x})$ at the points $\mathbf{x}_{n,k}^{(j)}$ where the JLF is going to be evaluated in (17). A natural choice for $\mathbf{x}'_{n,k}$ would be the state estimate $\hat{\mathbf{x}}_{n,k}$ in (18) since the $\mathbf{x}_{n,k}^{(j)}$ will usually lie in the vicinity of the true state $\mathbf{x}_n$ and $\hat{\mathbf{x}}_{n,k}$ can be expected to be reasonably close to $\mathbf{x}_n$. However, $\hat{\mathbf{x}}_{n,k}$ cannot be calculated because the JLF is not yet available. We thus propose to use for the reference point $\mathbf{x}'_{n,k}$ a preliminary local estimate $\hat{\mathbf{x}}_{n,k}^{\mathrm{prel}}$ that is calculated from the *local* measurement $\mathbf{z}_{n,k}$ and the predicted particles $\mathbf{x}_{n,k}^{(j)}$ obtained in Step 2. This is done in the following two additional substeps performed after Step 2:

Step 2a: Sensor k calculates temporary weights

$$\tilde{w}_{n,k}^{(j)} = \frac{f(\mathbf{z}_{n,k}|\mathbf{x}_{n,k}^{(j)})}{\sum_{j'=1}^{J} f(\mathbf{z}_{n,k}|\mathbf{x}_{n,k}^{(j')})}$$

using the *local* likelihood function $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ instead of the JLF $f(\mathbf{z}_n|\mathbf{x}_n)$ used in (17).

Step 2b: The preliminary estimate $\hat{\mathbf{x}}_{n,k}^{\mathrm{prel}}$ is calculated as in (18) but using the weights $\tilde{w}_{n,k}^{(j)}$, i.e., $\hat{\mathbf{x}}_{n,k}^{\mathrm{prel}} = \sum_{j=1}^{J} \tilde{w}_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)}$.

V. Simulation Results

As an example application for our DPF, we consider target tracking. The state $\mathbf{x}_n = (x_n\; y_n\; v_x\; v_y)^{\top}$ represents the 2D position and 2D velocity of a target in the x-y plane. The state-transition equation (cf. (13)) is $\mathbf{x}_n = \mathbf{G}\mathbf{x}_{n-1} + \mathbf{u}_n$, where

$$\mathbf{G} = \begin{pmatrix} 1 & 0 & 0.2 & 0 \\ 0 & 1 & 0 & 0.2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

and $\mathbf{u}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_u)$ with $\mathbf{C}_u = \mathrm{diag}\{1, 1, 0.01, 0.01\}$. The target emits a sound of amplitude $A=40$. The network consists of $K=25$ acoustic amplitude sensors that are deployed on a jittered grid within a rectangular region of size $200\,\mathrm{m} \times 200\,\mathrm{m}$. The (scalar) measurement $z_{n,k}$ of sensor k is given by (cf. (14))

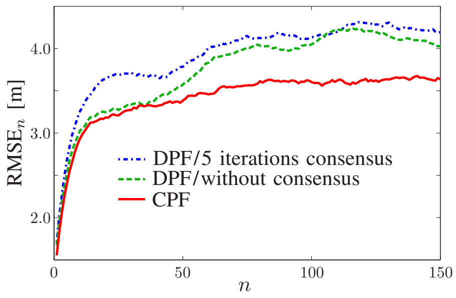

Figure 1. RMSE$_n$ versus time n.

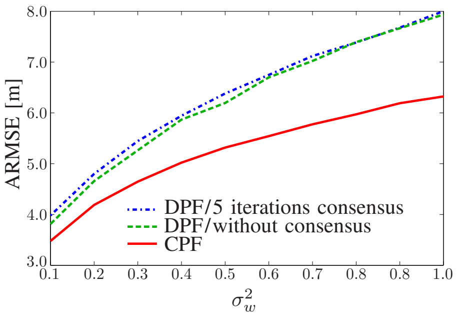

Figure 2. ARMSE versus measurement noise variance σ_w^2.

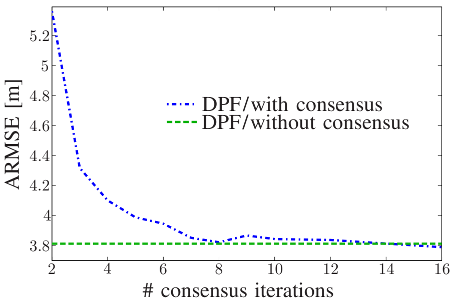

Figure 3. ARMSE versus number of consensus iterations.

$$z_{n,k} = h_{n,k}(\mathbf{x}_n) + w_{n,k} = \frac{A}{\|(x_n \; y_n)^\top - \boldsymbol{\xi}_{n,k}\|} + w_{n,k} \,,$$

where $\boldsymbol{\xi}_{n,k}$ denotes the position of sensor k at time n and $w_{n,k} \sim \mathcal{N}(0, \sigma_w^2)$ [13]. Each sensor communicates with other sensors within a range of 90m.

For LC, we use a Taylor series approximation of degree $R = 2$ about a reference point chosen according to Section IV-D. The sums in (9) are computed by an average consensus algorithm based on the Metropolis weight model in (11), using 5 iterations unless noted otherwise. At each iteration, 14 real numbers are broadcasted by each sensor. As performance benchmarks, we also consider an impractical DPF using exact, direct calculation of (9) instead of LC (denoted "without consensus"), as well as a centralized PF (CPF) that processes all sensor measurements at a fusion center. The number of particles at each sensor of the DPF and at the fusion center of the CPF is $J = 1000$.

In Fig. 1, we show the evolution of the root-mean-square error of the state estimates $\hat{\mathbf{x}}_{n,k}$, denoted RMSE$_n$, for measurement noise variance $\sigma_w^2 = 0.1$. RMSE$_n$ is computed as the square root of the average of the squared estimation error over all sensors and over 4000 simulation runs. Fig. 2 shows the *average* RMSE (ARMSE) versus the measurement noise variance σ_w^2. The ARMSE is computed by averaging RMSE$_n^2$ over all 150 simulated time instants n and taking the square root of the result. From both figures, we see that the performance of the DPF is worse—but not dramatically worse—than that of the CPF. This performance loss is smaller for lower values of σ_w^2. The impractical DPF without consensus has a slightly better performance than the DPF with consensus. Note that the performance loss of the DPF without consensus relative to the CPF is caused only by the Taylor approximation of the measurement functions, whereas the performance loss of the DPF with consensus is also caused by errors introduced by the consensus algorithm.

In Fig. 3, the dependence of the ARMSE on the number of consensus iterations is depicted. As the number of iterations increases, the performance of the DPF with consensus approaches the performance of the DPF without consensus.

VI. Conclusion

We proposed a distributed, consensus-based method for computing the joint—i.e., all-sensors—likelihood funtion (JLF) in a wireless sensor network. This "likelihood consensus" method uses an iterative consensus algorithm to compute, at each sensor, a sufficient statistic that describes an approximation to the JLF as a function of the state to be estimated. The consensus algorithm requires only local communications between neighboring sensors and operates without complex routing protocols. We also demonstrated the use of the likelihood consensus method for distributed 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. Finally, we applied the proposed distributed particle filter to a target tracking problem. Our results indicate good performance even in comparison with a centralized particle filter.

References

[1] A. Nayak and I. Stojmenović, *Wireless Sensor and Actuator Networks: Algorithms and Protocols for Scalable Coordination and Data Communication*. Hoboken, NJ: Wiley, 2010.

[2] S. Farahmand, S. I. Roumeliotis, and G. B. Giannakis, "Particle filter adaptation for distributed sensors via set membership," in *Proc. IEEE ICASSP-10*, Dallas, TX, pp. 3374–3377, Mar. 2010.

[3] D. Gu, "Distributed particle filter for target tracking," in *Proc. IEEE ICRA-07*, Rome, Italy, pp. 3856–3861, Apr. 2007.

[4] 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.

[5] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins University Press, 1996.

[6] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ: Prentice-Hall, 1998.

[7] 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.

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

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

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

[11] B. Ristic and S. Arulampalam, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Boston, MA: Artech House, 2004.

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

[13] F. Zhao and L. J. Guibas, *Wireless Sensor Networks: An Information Processing Approach*. Amsterdam, The Netherlands: Morgan Kaufmann, 2004.