DISTRIBUTED PARTICLE FILTERING IN THE PRESENCE OF MUTUALLY CORRELATED SENSOR NOISES

Ondrej Hlinka and Franz Hlawatsch

Institute of Telecommunications, Vienna University of Technology, Austria ({ohlinka,fhlawats}@nt.tuwien.ac.at)

ABSTRACT

We propose two distributed particle filter (DPF) algorithms for sensor networks with mutually correlated measurement noises at different sensors. With both algorithms, each sensor runs a local particle filter that knows the global (all-sensors) likelihood function and is thus able to compute a global state estimate based on the measurements of all sensors. We propose two alternative distributed, consensus-based methods for computing the global likelihood function at each sensor. Simulation results for a target tracking problem demonstrate that both DPF algorithms exhibit excellent performance, however with very different communications requirements.

***Index Terms*—** Distributed particle filter, correlated sensor noises, consensus, distributed target tracking, sensor network.

1. INTRODUCTION

Contribution and relation to previous work. For distributed sequential state estimation in wireless sensor networks without a fusion center, distributed particle filters (DPFs) are attractive solutions in nonlinear and/or non-Gaussian scenarios [1]. In particular, DPFs that employ consensus algorithms (e.g., [2–5]) are advantageous because they are robust to sensor and communication link failures and can obtain a global estimate at each sensor. To the best of our knowledge, all the existing consensus-based DPFs rely on the assumption that the measurement noises at the various sensors are mutually uncorrelated. However, there are many applications where this assumption is not satisfied. An example is given by acoustic measurements corrupted by an interfering ambient sound source (e.g., wind).

In this paper, we propose two consensus-based DPFs for additive Gaussian measurement noises that may be mutually correlated. The proposed DPFs extend the DPFs presented in [3, 4] and in [5]—which assume uncorrelated measurement noises—to the correlated case. In both DPFs, each sensor runs a local particle filter (PF) that obtains a global estimate (it is "global" in that it is based on the measurements of all sensors). The calculation of the weights at the local PFs uses the *global likelihood function* (GLF), which has to be provided to each sensor by a distributed computation scheme. In the first DPF, as in [3, 4], the value of the GLF for each particle is computed using a separate consensus algorithm. In the second DPF, inspired by the *likelihood consensus* introduced in [5], consensus algorithms are used to compute the coefficients of a basis expansion approximation of the GLF.

Paper outline. In Section 2, we introduce the system model and briefly review sequential Bayesian estimation. A generic DPF algorithm using the GLF is described in Section 3. In Section 4, we present two alternative methods for a distributed calculation of the GLF for correlated sensor noises. Finally, Section 5 demonstrates the excellent performance of our DPFs for a target tracking problem.

This work was supported by the Austrian Science Fund (FWF) under grant S10603.

2. SYSTEM MODEL

We consider a random, time-varying state vector $\mathbf{x}_n = (x_{n,1} \cdots x_{n,M})^\top$ of dimension M. The state evolves according to

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

where $\mathbf{g}_n(\cdot,\cdot)$ is a generally nonlinear function and $\mathbf{u}_n$ is white driving noise with a known probability density function (pdf) $f(\mathbf{u}_n)$. At time n, $\mathbf{x}_n$ is sensed by a network of K sensors according to the sensor measurement models

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

Here, $\mathbf{z}_{n,k}$ of dimension $N_{n,k}$ is the local measurement vector of sensor k at time n, $\mathbf{h}_{n,k}(\cdot)$ is a generally nonlinear local measurement function, and $\mathbf{v}_{n,k}$ is a local measurement noise vector. We assume that for any given n, the $\mathbf{v}_{n,k}$ for $k \in \{1, \dots, K\}$ are jointly Gaussian with zero mean; that $\mathbf{v}_{n,k}$ and $\mathbf{v}_{n',k'}$ are independent for $n \neq n'$, and that $\mathbf{v}_{n,k}$ is independent of the driving noise $\mathbf{u}_{n'}$ for all n'. However, for any given n, we allow the noises at different sensors, $\mathbf{v}_{n,k}$ and $\mathbf{v}_{n,k'}$ for $k \neq k'$, to be correlated.

With (2), the total (all-sensors) measurement model is given by

$$\mathbf{z}_n = \mathbf{h}_n(\mathbf{x}_n) + \mathbf{v}_n, \tag{3}$$

where $\mathbf{z}_n \triangleq (\mathbf{z}_{n,1}^\top \cdots \mathbf{z}_{n,K}^\top)^\top$ is the total measurement vector of dimension $N_n = \sum_{k=1}^K N_{n,k}$, $\mathbf{h}_n(\cdot) \triangleq (\mathbf{h}_{n,1}^\top(\cdot) \cdots \mathbf{h}_{n,K}^\top(\cdot))^\top$, and $\mathbf{v}_n \triangleq (\mathbf{v}_{n,1}^\top \cdots \mathbf{v}_{n,K}^\top)^\top \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_n)$. We assume that sensor k knows $\mathbf{g}_n(\cdot,\cdot)$ and $\mathbf{h}_{n,k}(\cdot)$ for all n, but not $\mathbf{h}_{n,k'}(\cdot)$ for $k' \neq k$.

The state-transition model (1) and the measurement model (3) 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 GLF $f(\mathbf{z}_n|\mathbf{x}_n)$. In particular, the GLF is given by

$$f(\mathbf{z}_n|\mathbf{x}_n) = f(\mathbf{v}_n)\big|_{\mathbf{v}_n = \mathbf{z}_n - \mathbf{h}_n(\mathbf{x}_n)} = C_n \exp\left(-\frac{1}{2} S_n(\mathbf{z}_n, \mathbf{x}_n)\right), \tag{4}$$

where $C_n \triangleq 1/\sqrt{(2\pi)^{N_n} \det\{\mathbf{C}_n\}}$ and

$$S_n(\mathbf{z}_n, \mathbf{x}_n) \triangleq [\mathbf{z}_n - \mathbf{h}_n(\mathbf{x}_n)]^\top \mathbf{Q}_n [\mathbf{z}_n - \mathbf{h}_n(\mathbf{x}_n)], \tag{5}$$

with the *precision matrix* $\mathbf{Q}_n \triangleq \mathbf{C}_n^{-1}$ ($\mathbf{C}_n$ is assumed nonsingular). Due to (4), calculation of the GLF $f(\mathbf{z}_n|\mathbf{x}_n)$ (up to the factor C_n, which is irrelevant to our estimation task) reduces to calculation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$ in (5). We note that, in accordance with the stacked structure of $\mathbf{v}_n = (\mathbf{v}_{n,1}^\top \cdots \mathbf{v}_{n,K}^\top)^\top$, the $N_n \times N_n$ precision matrix $\mathbf{Q}_n$ has a block structure with K^2 blocks $\mathbf{Q}_{n;k,k'}$ of dimensions $N_{n,k} \times N_{n,k'}$. Using these blocks, we can expand (5) as

$$S_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{k=1}^K \sum_{k'=1}^K [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]^\top \mathbf{Q}_{n;k,k'} [\mathbf{z}_{n,k'} - \mathbf{h}_{n,k'}(\mathbf{x}_n)]. \tag{6}$$

Our goal is to estimate the state $\mathbf{x}_n$ using the measurements of all sensors from time 1 to time n, $\mathbf{z}_{1:n} \triangleq (\mathbf{z}_1^\top \cdots \mathbf{z}_n^\top)^\top$. We consider the minimum mean-square error (MMSE) estimator [6]

$$\hat{\mathbf{x}}_n^{\text{MMSE}} \triangleq \mathrm{E}\{\mathbf{x}_n|\mathbf{z}_{1:n}\} = \int_{\mathbb{R}^M} \mathbf{x}_n\, f(\mathbf{x}_n|\mathbf{z}_{1:n})\, d\mathbf{x}_n\,. \quad (7)$$

The posterior pdf $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ in (7) can be calculated sequentially from the previous posterior $f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1})$ and the GLF $f(\mathbf{z}_n|\mathbf{x}_n)$ [7]. A computationally feasible approximation of this *sequential MMSE state estimation* is provided by the PF [8–10], which represents $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ by samples/particles and associated weights.

The total measurement noise vector $\mathbf{v}_n$ is a zero-mean Gaussian Markov random field [11], whose distribution is described by the precision matrix $\mathbf{Q}_n$. Gaussian Markov random fields are frequently used to model spatial correlations (e.g., [11–13]). In applications, $\mathbf{Q}_n$ tends to be approximately sparse, with only the blocks $\mathbf{Q}_{n;k,k'}$ corresponding to *spatially close* sensors k and k' being significantly nonzero. In fact, if $\mathbf{Q}_{n;k,k'} = \mathbf{0}$ for two sensors k and k', the corresponding noise vectors $\mathbf{v}_{n,k}$ and $\mathbf{v}_{n,k'}$ are conditionally independent given all the other noise vectors in the network [11]. For a given sensor k, we define the *neighbor set* $\mathcal{N}_k$ as the set of all sensors $k' \neq k$ such that $\mathbf{Q}_{n;k,k'} \neq \mathbf{0}$. We assume that sensor k is able to communicate with all sensors $k' \in \mathcal{N}_k$; this is consistent with the fact that, typically, these sensors are spatially close to sensor k. Furthermore, we assume that sensor k knows $\mathbf{Q}_{n;k,k'}$ for all $k' \in \mathcal{N}_k$.

3. DISTRIBUTED PARTICLE FILTERING

We propose two DPF algorithms in which each sensor tracks a particle representation of the global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$ using a *local PF*. At each time n, the local PF at sensor k calculates a state estimate $\hat{\mathbf{x}}_{n,k}$ that is based on $\mathbf{z}_{1:n}$, i.e., the measurements of *all* sensors up to time n. This requires the GLF $f(\mathbf{z}_n|\mathbf{x}_n)$ (or, equivalently, $S_n(\mathbf{z}_n, \mathbf{x}_n)$) in the weight calculation step of the local PFs. Since each sensor initially knows only its own local likelihood function $f(\mathbf{z}_{n,k}|\mathbf{x}_n)$ and its local measurement $\mathbf{z}_{n,k}$, a distributed method to obtain $S_n(\mathbf{z}_n, \mathbf{x}_n)$ at each sensor is needed. While the two proposed DPFs are based on the same generic DPF algorithm, which is stated below, they differ in the distributed computation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$.

GENERIC DPF ALGORITHM

The local PF at sensor k is initialized at time $n=0$ with J particles $\mathbf{x}_{0,k}^{(j)}$, $j \in \{1,\ldots,J\}$ randomly drawn from the prior pdf $f(\mathbf{x}_0)$. The weights are initially equal, i.e., $w_{0,k}^{(j)} = 1/J$ for all j.

At time $n \geq 1$, the local PF at sensor k performs the following steps, which are identical for all k.

1. For each previous particle $\mathbf{x}_{n-1,k}^{(j)}$, a new particle $\mathbf{x}_{n,k}^{(j)}$ is drawn from $f(\mathbf{x}_n|\mathbf{x}_{n-1,k}^{(j)}) \equiv f(\mathbf{x}_n|\mathbf{x}_{n-1})\big|_{\mathbf{x}_{n-1}=\mathbf{x}_{n-1,k}^{(j)}}$.

2. Nonnormalized weights associated with the particles $\mathbf{x}_{n,k}^{(j)}$ are calculated according to

$$\tilde{w}_{n,k}^{(j)} = \exp\left(-\frac{1}{2} S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})\right), \quad j \in \{1,\ldots,J\}. \quad (8)$$

   Two alternative methods for a distributed computation of $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$ will be discussed in Sections 4.2 and 4.3.

3. The weights $\tilde{w}_{n,k}^{(j)}$ are normalized as

$$w_{n,k}^{(j)} = \frac{\tilde{w}_{n,k}^{(j)}}{\sum_{j'=1}^{J} \tilde{w}_{n,k}^{(j')}}, \quad j \in \{1,\ldots,J\}.$$

   The set $\left\{\left(\mathbf{x}_{n,k}^{(j)}, w_{n,k}^{(j)}\right)\right\}_{j=1}^{J}$ provides a particle representation of the current global posterior $f(\mathbf{x}_n|\mathbf{z}_{1:n})$.

4. From the particle representation $\left\{\left(\mathbf{x}_{n,k}^{(j)}, w_{n,k}^{(j)}\right)\right\}_{j=1}^{J}$, an approximation of the global MMSE state estimate (7) is computed according to $\hat{\mathbf{x}}_{n,k} = \sum_{j=1}^{J} w_{n,k}^{(j)} \mathbf{x}_{n,k}^{(j)}$.

5. The set $\left\{\left(\mathbf{x}_{n,k}^{(j)}, w_{n,k}^{(j)}\right)\right\}_{j=1}^{J}$ can be resampled if necessary (see, e.g., [9] for indications when resampling should be performed). This produces J resampled particles $\mathbf{x}_{n,k}^{(j)}$. The weights are then redefined to be identical, i.e., $w_{n,k}^{(j)} = 1/J$.

We note that in this algorithm, each sensor performs almost the same operations as the fusion center in a centralized PF. The only difference is Step 2, in which a distributed computation of $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$ is performed. This is also the only step of the DPF algorithm that requires communications with neighboring sensors.

4. DISTRIBUTED CALCULATION OF $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$

We now present two alternative methods for the distributed computation of $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$, which is required in (8).

4.1. Uncorrelated Measurement Noises

First, we briefly review the case of mutually uncorrelated measurement noises $\mathbf{v}_{n,k}$. Here, the precision matrix $\mathbf{Q}_n$ is block-diagonal, and hence (6) simplifies as

$$S_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{k=1}^{K} [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]^\top \mathbf{Q}_{n;k,k} [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]\,. \quad (9)$$

As proposed in [3, 4], each sensor k can calculate $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$ based on (9) by first calculating $[\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_{n,k}^{(j)})]^\top \mathbf{Q}_{n;k,k} [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_{n,k}^{(j)})]$ for each particle $\mathbf{x}_{n,k}^{(j)}$, $j \in \{1,\ldots,J\}$ (this can be done locally) and then using an average consensus algorithm [14, 15] to compute the sum over all sensors in (9) (this requires communications with neighboring sensors). One consensus is executed for each particle, i.e., for each $j \in \{1,\ldots,J\}$. This method requires identical particle sets $\left\{\mathbf{x}_{n,k}^{(j)}\right\}_{j=1}^{J}$ at each sensor, which can be achieved by (i) using random number generators that are synchronized across all sensors and (ii) executing an additional max-consensus algorithm [15] for each particle (see Section 4.2 for details). The communication requirements tend to be high because two consensus algorithms—one average consensus and one max-consensus—are needed for each particle and the number of particles J may be as high as several thousands.

An alternative method is provided by the *likelihood consensus*, which relies on the following approximate (finite-order) basis expansion of $S_n(\mathbf{z}_n, \mathbf{x}_n)$ [5, 16]:

$$S_n(\mathbf{z}_n, \mathbf{x}_n) \approx \sum_{r=1}^{R} a_{n,r}(\mathbf{z}_n)\, \psi_{n,r}(\mathbf{x}_n)\,. \quad (10)$$

Here, the $a_{n,r}(\mathbf{z}_n)$ are expansion coefficients and the $\psi_{n,r}(\mathbf{x}_n)$ are basis functions that are known to all sensors. For a block-diagonal precision matrix $\mathbf{Q}_n$, the coefficients $a_{n,r}(\mathbf{z}_n)$ can be calculated in a distributed way using R instances of an average consensus algorithm (see [5, 16] for details). Each sensor can then locally calculate an approximation of $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$ for all $j \in \{1,\ldots,J\}$. Since the number of expansion coefficients R is typically much lower than the number of particles, significant savings in communications can be

achieved compared to the first method. Furthermore, this method does not require synchronized random number generators. On the other hand, only an approximation of $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$ is obtained, which may lead to a certain performance loss.

4.2. Method 1: Direct Per-Particle Evaluation

We now consider the case of mutually correlated measurement noises $\mathbf{v}_{n,k}$. Here, $S_n(\mathbf{z}_n, \mathbf{x}_n)$ is given by the double sum expression (6). To extend the first consensus-based method of Section 4.1 to the correlated case, we rewrite (6) as

$$S_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{k=1}^{K} [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)]^\top \sum_{k' \in \tilde{\mathcal{N}}_k} \mathbf{Q}_{n;k,k'} [\mathbf{z}_{n,k'} - \mathbf{h}_{n,k'}(\mathbf{x}_n)], \quad (11)$$

with $\tilde{\mathcal{N}}_k \triangleq \mathcal{N}_k \cup \{k\}$ (here, $\mathcal{N}_k$ was defined in Section 2). Note that this expression involves only the nonzero blocks $\mathbf{Q}_{n;k,k'}$.

The calculation now works as follows. First, the measurements $\mathbf{z}_{n,k'}$ and suitable descriptions of the measurement functions $\mathbf{h}_{n,k'}(\cdot)$ of all neighbors $k' \in \mathcal{N}_k$ are transmitted to sensor k. Thus, sensor k is now able to locally calculate

$$s_{n,k}\big(\{\mathbf{z}_{n,k'}\}_{k' \in \tilde{\mathcal{N}}_k}, \mathbf{x}_{n,k}^{(j)}\big) \triangleq [\mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_{n,k}^{(j)})]^\top \sum_{k' \in \tilde{\mathcal{N}}_k} \mathbf{Q}_{n;k,k'} [\mathbf{z}_{n,k'} - \mathbf{h}_{n,k'}(\mathbf{x}_{n,k}^{(j)})],$$

for $j \in \{1, \ldots, J\}$. From (11),

$$S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)}) = \sum_{k=1}^{K} s_{n,k}\big(\{\mathbf{z}_{n,k'}\}_{k' \in \tilde{\mathcal{N}}_k}, \mathbf{x}_{n,k}^{(j)}\big), \quad j \in \{1, \ldots, J\}. \quad (12)$$

These J sums over all sensors k can be computed by executing J instances of an average consensus algorithm.

This method presupposes that identical sets of particles $\{\mathbf{x}_{n,k}^{(j)}\}_{j=1}^{J}$ are sampled at each sensor. This, in turn, requires that the local random number generators at all sensors are synchronized—such that the same pseudo-random numbers are obtained in the entire network—and that identical particle representations $\{(\mathbf{x}_{n-1,k}^{(j)}, w_{n-1,k}^{(j)})\}_{j=1}^{J}$ of the previous global posterior $f(\mathbf{x}_{n-1}|\mathbf{z}_{1:n-1})$ are available at each sensor. However, the $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$ and consequently the weights $\tilde{w}_{n,k}^{(j)}$ (see (8)) obtained at different sensors will differ slightly since only a finite number of consensus iterations can be performed. As proposed in [3, 4] for the uncorrelated case, identical weights can be obtained by using max-consensus algorithms to produce at each sensor the maximum of the weights of all sensors. One max-consensus has to be executed for each $j \in \{1, \ldots, J\}$.

The communication requirements of this method at any given time n are as follows. Prior to executing the consensus algorithms, each sensor k broadcasts to its neighbors $M_0 = N_{n,k} + H_{n,k}$ real numbers. Here, $N_{n,k}$ is the dimension of $\mathbf{z}_{n,k}$ and $H_{n,k}$ is the count of real numbers describing $\mathbf{h}_{n,k}(\cdot)$. While executing the consensus algorithms, each sensor broadcasts to its neighbors $M_c = J(I_a + I_m)$ real numbers. Here, J is the number of particles and I_a and I_m denote the number of iterations of the average consensus and of the max-consensus, respectively. Since the number of particles J can be large, the communication requirements can be very high; they can, however, be reduced by using proposal adaptation to decrease J [4] or by computing only the largest weights [3].

4.3. Method 2: Basis Expansion Approximation

Motivated by the high communication requirements of the method presented above, we now extend the second consensus-based method of Section 4.1 to the correlated case. We approximate the function $\mathbf{b}_{n,k}(\mathbf{x}_n; \mathbf{z}_{n,k}) \triangleq \mathbf{z}_{n,k} - \mathbf{h}_{n,k}(\mathbf{x}_n)$ occurring in (11) by a finite-order basis expansion, i.e.,

$$\mathbf{b}_{n,k}(\mathbf{x}_n; \mathbf{z}_{n,k}) \approx \sum_{r=1}^{R} \boldsymbol{\alpha}_{n,k,r}(\mathbf{z}_{n,k})\, \varphi_{n,r}(\mathbf{x}_n). \quad (13)$$

Here, the $\boldsymbol{\alpha}_{n,k,r}(\mathbf{z}_{n,k})$ are expansion coefficients that contain all the sensor-local information (including the sensor measurement $\mathbf{z}_{n,k}$) and the $\varphi_{n,r}(\mathbf{x}_n)$ are fixed, sensor-independent basis functions that are known to all sensors. At each time n, the expansion coefficients $\boldsymbol{\alpha}_{n,k,r}(\mathbf{z}_{n,k})$ in (13) are calculated locally at each sensor k by means of least squares fitting [17] based on the J data points $\{(\mathbf{x}_{n,k}^{(j)}, \mathbf{b}_{n,k}(\mathbf{x}_{n,k}^{(j)}; \mathbf{z}_{n,k}))\}_{j=1}^{J}$. Here, the use of the particles $\mathbf{x}_{n,k}^{(j)}$ drawn in Step 1 of the generic DPF algorithm ensures a good approximation in those regions of the state space where $S_n(\mathbf{z}_n, \mathbf{x}_n)$ is evaluated in Step 2 (see (8)). Substituting (13) into (11) and changing the order of summation, we obtain the following basis expansion approximation of $S_n(\mathbf{z}_n, \mathbf{x}_n)$:

$$S_n(\mathbf{z}_n, \mathbf{x}_n) \approx \tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n) \triangleq \sum_{r_1=1}^{R} \sum_{r_2=1}^{R} a_{n,r_1,r_2}(\mathbf{z}_n)\, \varphi_{n,r_1}(\mathbf{x}_n)\, \varphi_{n,r_2}(\mathbf{x}_n), \quad (14)$$

where

$$a_{n,r_1,r_2}(\mathbf{z}_n) = \sum_{k=1}^{K} \boldsymbol{\alpha}_{n,k,r_1}^\top(\mathbf{z}_{n,k}) \sum_{k' \in \tilde{\mathcal{N}}_k} \mathbf{Q}_{n;k,k'} \boldsymbol{\alpha}_{n,k',r_2}(\mathbf{z}_{n,k'}). \quad (15)$$

We note that, using any one-to-one index mapping $(r_1, r_2) \to r$, the approximation $\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n)$ in (14) can also be written in the form (10), i.e., $\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n) = \sum_{r=1}^{R'} a'_{n,r}(\mathbf{z}_n)\, \psi_{n,r}(\mathbf{x}_n)$, with $R' = R^2$, $a'_{n,r}(\mathbf{z}_n) = a_{n,r_1,r_2}(\mathbf{z}_n)$, and $\psi_{n,r}(\mathbf{x}_n) = \varphi_{n,r_1}(\mathbf{x}_n)\, \varphi_{n,r_2}(\mathbf{x}_n)$.

The overall calculation works as follows. First, the coefficients $\{\boldsymbol{\alpha}_{n,k',r}(\mathbf{z}_{n,k'})\}_{r=1}^{R}$ of all neighbors $k' \in \mathcal{N}_k$ are transmitted to sensor k, which locally calculates $\boldsymbol{\alpha}_{n,k,r_1}^\top(\mathbf{z}_{n,k}) \sum_{k' \in \tilde{\mathcal{N}}_k} \mathbf{Q}_{n;k,k'} \times \boldsymbol{\alpha}_{n,k',r_2}(\mathbf{z}_{n,k'})$ for $(r_1, r_2) \in \{1, \ldots, R\}^2$. Then, the sum over all sensors k in (15) is computed using average consensus algorithms. Thereby, each sensor obtains approximations of the coefficients $a_{n,r_1,r_2}(\mathbf{z}_n)$. Note that one consensus algorithm is needed for each $(r_1, r_2) \in \{1, \ldots, R\}^2$. Finally, using the approximate $a_{n,r_1,r_2}(\mathbf{z}_n)$, each sensor is able to approximately evaluate $\tilde{S}_n(\mathbf{z}_n, \mathbf{x}_n)$ for any value of $\mathbf{x}_n$, including the particles $\mathbf{x}_{n,k}^{(j)}$ as required in Step 2 of the generic DPF algorithm.

The communication requirements of this method at any given time n are as follows. To transmit the coefficients $\{\boldsymbol{\alpha}_{n,k,r}(\mathbf{z}_{n,k})\}_{r=1}^{R}$ to all neighbors, sensor k needs to broadcast $M_0 = R N_{n,k}$ real numbers. Furthermore, the consensus algorithms computing the sum over all k in (15) require each sensor to broadcast to its neighbors $M_c = I_a R^2$ real numbers, where I_a is the number of consensus iterations. Typically, these communication requirements are significantly lower than those of the method presented in Section 4.2. Furthermore, since the local PFs at the various sensors operate independently (only $S_n(\mathbf{z}_n, \mathbf{x}_n)$ is computed in a distributed manner), the local random number generators need not be synchronized. On the other hand, the basis expansion approximation may result in a certain performance loss.

5. SIMULATION RESULTS

We consider a target tracking application in which the state vector $\mathbf{x}_n = (x_n\ y_n\ \dot{x}_n\ \dot{y}_n)^\top$ represents the 2D position and velocity of a

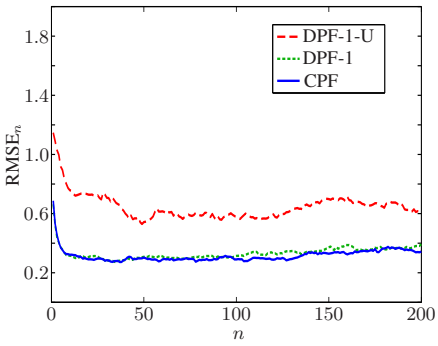

Fig. 1. RMSE$_n$ versus time n for DPF-1, DPF-1-U, and CPF.

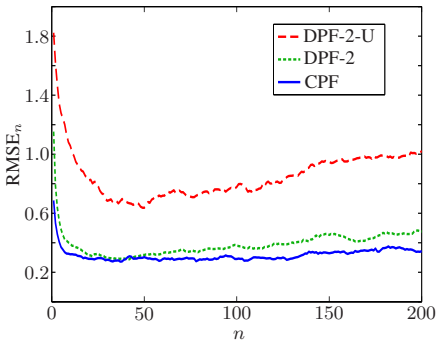

Fig. 2. RMSE$_n$ versus time n for DPF-2, DPF-2-U, and CPF.

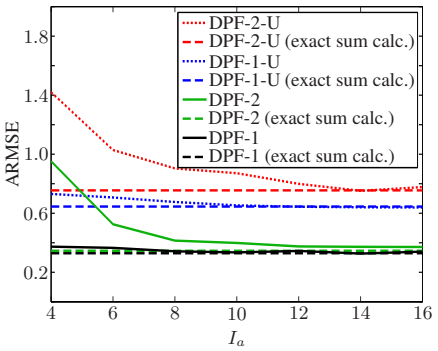

Fig. 3. ARMSE versus the number of iterations of average consensus.

single target in the x-y plane. The state vector evolves according to $\mathbf{x}_n = \mathbf{G}\mathbf{x}_{n-1} + \mathbf{W}\mathbf{u}_n$, $n = 1, 2, \ldots$, where the matrices $\mathbf{G} \in \mathbb{R}^{4\times 4}$ and $\mathbf{W} \in \mathbb{R}^{4\times 2}$ are chosen as in [1] and the driving noise vectors $\mathbf{u}_n \in \mathbb{R}^2$ are independent and identically distributed according to $\mathcal{N}(\mathbf{0}_2, \sigma_u^2 \mathbf{I}_2)$ with $\sigma_u^2 = 0.00035$. The target emits an acoustic or radio signal with a known, constant transmit power $A = 10$.

The network consists of $K = 25$ sensors, which are deployed on a jittered grid within a square region of size $d_a \times d_a$ with $d_a = 40$. Each sensor communicates with other sensors within a range of $d_c = 15$. The (scalar) measurement of sensor k is given by (cf. (2))

$$z_{n,k} = \frac{A}{\|\boldsymbol{\rho}(\mathbf{x}_n) - \boldsymbol{\xi}_k\|^2} + v_{n,k}\,,$$

where $\boldsymbol{\xi}_k$ is the position of sensor k and $\boldsymbol{\rho}(\mathbf{x}_n) \triangleq (x_n \;\; y_n)^\top$ is the target position. The measurement noise $v_{n,k}$ is assumed to be zero-mean Gaussian, independent for different n, and correlated across the sensors k. The entries of the (time-independent) precision matrix $\mathbf{Q}$ of the all-sensors measurement noise vector $\mathbf{v}_n = (v_{n,1} \cdots v_{n,25})^\top$ are chosen as [12]

$$Q_{k,k'} = \begin{cases} q > 0\,, & k = k' \\ -c(d_a - r_{k,k'})\sqrt{Q_{k,k} Q_{k',k'}}\,, & k \neq k',\ r_{k,k'} \leq d_c \\ 0\,, & k \neq k',\ r_{k,k'} > d_c\,, \end{cases}$$

where $r_{k,k'} \triangleq \|\boldsymbol{\xi}_k - \boldsymbol{\xi}_{k'}\|$. We set $q = 1000$ and $c = 0.00615$; this ensures sufficient correlation while the precision matrix remains positive definite [12]. Note that since the $v_{n,k}$ are now scalars, the blocks $\mathbf{Q}_{n;k,k'}$ in (6) reduce to the scalar entries $Q_{k,k'}$.

We simulated two DPFs, briefly referred to as DPF-1 and DPF-2, that use the generic DPF algorithm of Section 3. For the distributed calculation of $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$ in Step 2 of that algorithm, DPF-1 uses Method 1 in Section 4.2 and DPF-2 uses Method 2 in Section 4.3. We also simulated a centralized PF (CPF) that processes all sensor measurements at a fusion center. Finally, we simulated two standard consensus-based DPFs, referred to as DPF-1-U and DPF-2-U, that use the generic DPF algorithm of Section 3 based on the (incorrect) assumption of uncorrelated measurement noises, i.e., disregarding the nonzero off-diagonal entries of $\mathbf{Q}$. For the distributed calculation of $S_n(\mathbf{z}_n, \mathbf{x}_{n,k}^{(j)})$, DPF-1-U and DPF-2-U use, respectively, the first [3,4] and second method [5] reviewed in Section 4.1. For DPF-2 and DPF-2-U, we approximate $\mathbf{b}(\mathbf{x}_n; z_{n,k})$ by a multivariate polynomial of degree $R_p = 2$, which corresponds to a basis expansion order $R = 6$. The basis expansion of $S_n(\mathbf{z}_n, \mathbf{x}_n)$ in (14) is thus obtained as a polynomial of degree $2R_p = 4$. For all DPFs, the relevant sums over all sensors k (e.g., in (12) for DPF-1 and in (15) for DPF-2) are computed by an average consensus algorithm with Metropolis weights [18], using $I_a = 10$ consensus iterations unless stated otherwise. The number of particles at each sensor (for the DPFs) and at the fusion center (for the CPF) is $J = 5000$.

As a performance measure, we use the root-mean-square error of the estimated $\boldsymbol{\rho}(\mathbf{x}_n)$, denoted RMSE$_n$, which is computed as the square root of the average of the squared estimation error over all sensors and over 1000 simulation runs. We also compute the *average* RMSE (ARMSE) by averaging RMSE$_n^2$ over all 200 simulated time instants n and taking the square root of the result.

Figures 1 and 2 show the evolution of RMSE$_n$. It can be seen that both DPF-1 and DPF-2 perform almost as well as the CPF and significantly outperform DPF-1-U and DPF-2-U. Furthermore, DPF-1 performs slightly better than DPF-2. However, this comes at the cost of substantially higher communication requirements: in the case of DPF-1, during one time step, each sensor transmits 50003 real numbers, compared to only 156 in the case of DPF-2. The communication requirements of DPF-1-U and DPF-2-U are 50000 and 150, respectively, and thus only slightly lower.

Figure 3 shows the ARMSE versus the number of iterations used by the average consensus algorithms, I_a. As a benchmark, we also consider hypothetical DPFs in which the approximate sum calculations performed by the consensus algorithms are replaced by direct, exact sum calculations; this corresponds to an infinite number of consensus iterations. As expected, the ARMSE of DPF-1, DPF-2, DPF-1-U, and DPF-2-U decreases with growing I_a and approaches the ARMSE of the respective hypothetical DPF. Note, however, that the communication requirements increase with growing I_a. Furthermore, we see that DPF-1 and DPF-1-U require fewer iterations than DPF-2 and DPF-2-U to achieve an ARMSE that is close to that of the respective hypothetical DPFs. Due to the approximation involved in DPF-2 and DPF-2-U, these methods are more sensitive to variations across the sensors of the quantities calculated by the consensus algorithms. Hence, they require more consensus iterations to achieve good performance. Nevertheless, the communication requirements of DPF-2 and DPF-2-U are still significantly smaller than those of DPF-1 and DPF-1-U.

6. CONCLUSION

We extended two existing consensus-based distributed particle filter algorithms to the case of mutually correlated sensor noises. In both algorithms, the state estimates computed by the local particle filters at the various sensors take into account the past and present measurements of all sensors. This is enabled by two alternative methods for a distributed approximate calculation of the global likelihood function. Simulation results for a target tracking problem demonstrated the good performance of both proposed algorithms, with one algorithm slightly outperforming the other at the cost of significantly higher communications requirements.

7. REFERENCES

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

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

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

[4] 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.

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

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

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

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

[9] 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.

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

[11] H. Rue and L. Held, *Gaussian Markov Random Fields: Theory and Applications*. Boca Raton, FL: Chapman & Hall/CRC, 2005.

[12] 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.

[13] Y. Sung, H. V. Poor, and H. Yu, "How much information can one get from a wireless ad hoc sensor network over a correlated random field?," *IEEE Trans. Signal Process.*, vol. 55, pp. 2827–2847, Jun. 2009.

[14] 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.

[15] A. Tahbaz-Salehi and A. Jadbabaie, "A one-parameter family of distributed consensus algorithms with boundary: from shortest paths to mean hitting times," in *Proc. IEEE CDC-06*, San Diego, CA, pp. 4664–4669, Dec. 2006.

[16] 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.

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

[18] 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.