Joint 48th IEEE Conference on Decision and Control and
28th Chinese Control Conference
Shanghai, P.R. China, December 16-18, 2009

Markov Chain Distributed Particle Filters (MCDPF)

Sun Hwan Lee* and Matthew West†

Abstract— **Distributed particle filters (DPF) are known to provide robustness for the state estimation problem and can reduce the amount of information communication compared to centralized approaches. Due to the difficulty of merging multiple distributions represented by particles and associated weights, however, most uses of DPF to date tend to approximate the posterior distribution using a parametric model or to use a predetermined message path. In this paper, the Markov Chain Distributed Particle Filter (MCDPF) algorithm is proposed, based on particles performing random walks across the network. This approach maintains robustness since every sensor only needs to exchange particles and weights locally and furthermore enables more general representations of posterior distributions because there are no a priori assumptions on distribution form. The paper provides a proof of weak convergence of the MCDPF algorithm to the corresponding centralized particle filter and the optimal filtering solution, and concludes with a numerical study showing that MCDPF leads to a reliable estimation of the posterior distribution of a nonlinear system.**

I. Introduction

Distributed Particle Filters (DPF) have been emerging as an efficient tool for state estimation, for instance in target tracking with a robotic navigation system [1][2]. The general benefits of distributed estimation include the robustness of the estimation, the reduction of the amount of information flow, and estimation results comparable to the centralized approach. Much effort has been directed toward the realization of the decentralized Kalman filtering [3][4][5] but decentralized particle filters were thought to be challenging due to the difficulty of merging probability distributions represented by particles and weights [6]. The currently existing distributed particle filtering methods, however, are not able to gain all of these advantages or turn out to benefit from these properties only with relatively low dimensional systems by introducing an assumption such as Gaussian Mixture Model (GMM). The developed methods so far try to avoid exchanging the raw data, namely particles and associated weights, mainly due to the large amount of information that implies. The communication of such raw data scales better with system dimension, however, than the existing methods do. The distributed particle filter proposed in this paper exchanges particles and weights only between nearest neighbor nodes and estimates the true state by assimilating data with an algorithm based on a Markov chain random walk.

Past work on distributed particle filters can be broadly categorized into two approaches, namely massage passing approaches and consensus-based local information exchange methods. Message passing approaches transfer information along a predetermined route covering an entire network. For example, [7] passes parameters of the parametric model of the posterior distribution while [8] transmits the raw information, particles and weights, or the parameters of a GMM approximation of the posterior distribution. Consensus based methods communicate the information only between the nearest nodes and achieve global consistency by consensus filtering. The type of exchanged information can be, for example, the parameters of a GMM approximation [2] or the local mean and covariance [9].

The massage passing approaches [7][8] can have reduced robustness because the distributed algorithms cannot themselves cope with the failure of even one node since the system uses fixed message paths. Furthermore, the assumption of synchronization with identical particles at every node can cause fragility. On the other hand, the consensus based approaches so far proposed [2][9] all use approximations of the posterior distribution with GMM because to reduce information flow. The amount of reduced information, however, is not significant compared with transmitting full particles when the dimension of the system is very large, due to the covariance matrix of the posterior distribution. For an n-dimensional system, consensus based DPF with a GMM approximation has to transmit $\mathcal{O}(cn^2|E|)$ data through the entire network per consensus step, where c is the number of Gaussian mixtures and $|E|$ is the number of network edges. If the DPF is realized with exchanging N particles, however, then the amount of information per Markov chain iteration is $\mathcal{O}(nmN)$, where m is the number of nodes. A detailed description of the Markov chain iteration is given below. For a system with $cn|E| \gg mN$, a GMM approximation no longer benefits from the effect of reduced information flow. Furthermore, it frequently happens that the posterior distribution is not well-approximated by small number of combination of Gaussian distribution.

In this paper we propose a distributed particle filter based on exchanging particles and associated weights according to a Markov chain random walk. This approach maintains the robustness of a distributed system since each node only needs local information and it scales well in the case of non-Gaussian and high dimensional systems.

The contents of this paper are as follows. In section II, a review of basic properties and theorems for random walks on graphs is provided. Section III contains the centralized particle filter (CPF) algorithm and the proposed decentralized particle filter algorithm. In addition, the weak convergence of the posterior distribution of the DPF to that of the CPF

*S. H. Lee is with the Department of Aeronautics and Astronautics, Stanford University, sunhlee@stanford.edu
†M. West is with the Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, mwest@illinois.edu

and the optimal filter is proved in this section. A numerical example using a bearing only tracking problem with four sensors is given in section IV. Conclusion and future works are discussed in section V.

II. Random Walks on Graphs

A sensor network system can modeled as a graph, $G = (V, E)$, with a normalized adjacency matrix $\mathcal{A}$. The vertices $V = \{1, \ldots, m\}$ correspond to nodes or sensors in the network and edges E represent the connections between sensors. The neighbors of node i are given by $N_i = \{j \in V : \mathcal{A}_{ij} \neq 0\}$. Matrix $\mathcal{A}$ is a Markov transition probability matrix defined on the graph because it satisfies $\mathcal{A} \geq 0$ and $\mathcal{A}\mathbf{1} = \mathbf{1}$. Consequently a random walk on the network can be defined according to matrix $\mathcal{A}$ and we assume no self-loops in the chain. Here we review several properties of random walks on graphs which are useful for the development of DPF.

Proposition 2.1: If $\mathcal{A}$ is the normalized adjacency matrix of an undirected, connected graph G then Markov chain defined by $\mathcal{A}$ has a unique stationary distribution Π and $\Pi_i > 0$ for all $i \in V$. Moreover, for any initial distribution,

$$\lim_{k\to\infty} \frac{M(i,k)}{k} = \Pi_i, \tag{1}$$

where $M(i,k)$ is the number of visits to state i during k steps.

Theorem 2.2: If $\mathcal{A}$ is the normalized adjacency matrix of an undirected connected graph G then the stationary distribution of the Markov chain defined by $\mathcal{A}$ is given by $\Pi = (\Pi_1, \Pi_2, \ldots, \Pi_m)$, where $\Pi_i = \frac{d(i)}{2|E|}$. Here $d(i)$ is the degree of node i and $|E|$ is the number of edges in the graph.

Proof: We compute

$$(\Pi\mathcal{A})_j = \sum_{i=1}^{m} \Pi_i \mathcal{A}_{ij} = \sum_{i=1}^{m} \frac{d(i)}{2|E|} \frac{|E_{ij}|}{d(i)} = \Pi_j, \tag{2}$$

where $|E_{ij}|$ is the number of edges connecting nodes i and j and $\sum_{i=1}^{m} |E_{ij}| = d(j)$. Since Π satisfies $\Pi = \Pi\mathcal{A}$, Π is the stationary distribution of the Markov chain defined by $\mathcal{A}$. ■

III. Particle Filters

Suppose we have the general state space model,

$$x_{t+1} = f(x_t, w_t) \tag{3}$$
$$y_t = g(x_t, v_t), \tag{4}$$

where $x_t \in \mathbf{R}^n$, $y_t \in \mathbf{R}^p$, and w_t, v_t are process and measurement noises respectively. We define two stochastic processes, $X = \{X_t, t \in \mathbb{N}\}$ and $Y = \{Y_t, t \in \mathbb{N}\}$, where X and Y are a signal process and an observation process respectively. The signal process X is a Markovian process with an initial distribution $\mu(x_0)$ and transition kernel $K(dx_t|x_{t-1})$ and the observation process Y is conditionally independent given X. For simplicity, we assume that the kernel and conditional probability distribution of Y attain Lebesgue measures.

$$\Pr(X_t \in A | X_{t-1} = x_{t-1}) = \int_A K(x_t|x_{t-1}) dx_t \tag{5}$$
$$\Pr(Y_t \in B | X_t = x_t) = \int_B \rho(y_t|x_t) dy_t \tag{6}$$

where $\rho(y_t|x_t)$ is the transition probability density of a measurement y_t given the state x_t.

The filtering problem is to estimate the true state x_t at time t given the time series of observations $y_{1:t}$. The prediction and updating of the optimal filtering based on Bayes' recursion are given as follows.

$$p(x_t|y_{1:t-1}) = \int_{\mathbf{R}^n} p(x_{t-1}|y_{1:t-1}) K(x_t|x_{t-1})\, dx_{t-1} \tag{7}$$
$$p(x_t|y_{1:t}) = \frac{\rho(y_t|x_t) p(x_t|y_{1:t-1})}{\int_{\mathbf{R}^n} \rho(y_t|x_t) p(x_t|y_{1:t-1})\, dx_t}. \tag{8}$$

Analytic solutions for the posterior distribution in (8) do not generally exist except in special cases, such as linear dynamical systems with Gaussian noise. In the particle filtering setting, the posterior distribution is represented by a group of particles and associated weights so that the integral in (8) is approximated by the sum of discrete values.

A. Centralized Particle Filters

Particle filtering is a recursive method to estimate the true state, given the time series of measurements [10][11]. Suppose the posterior distribution at time $t-1$, $\pi_{t-1|t-1}(dx_{t-1})$, is approximated by N particles $\{x_{t-1}^i\}_{i=1}^N$. Then we have

$$\begin{aligned} p(x_{t-1}|y_{1:t-1}) &\triangleq \pi_{t-1|t-1}(dx_{t-1}) \\ &\approx \pi_{t-1|t-1}^N(dx_{t-1}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_{t-1}^i}(dx_{t-1}). \end{aligned} \tag{9}$$

where particle i is at position x_{t-1}^i in state space. Now, particles go through the prediction and measurement update steps to approximate the posterior distribution at time t. Given N particles, new particles are sampled from the transition kernel density, $\tilde{x}_t^i \sim \pi_{t-1|t-1}^N K(dx_t) = \frac{1}{N}\sum_{i=1}^N K(x_t|x_{t-1}^i)$. This set of particles is the approximation of $\pi_{t|t-1}$,

$$p(x_t|y_{1:t-1}) \triangleq \pi_{t|t-1}(dx_t) \tag{10}$$
$$\approx \tilde{\pi}_{t|t-1}^N(dx_t) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\tilde{x}_t^i}(d\tilde{x}_t). \tag{11}$$

If the empirical distribution in (11) is substituted in (8), we have the following distribution approximating the posterior distribution $p(x_t|y_{1:t})$.

$$\begin{aligned} \tilde{\pi}_{t|t}^N(dx_t) &\triangleq \frac{\rho(y_t|x_t)\tilde{\pi}_{t|t-1}^N(dx_t)}{\int_{\mathbb{R}^n} \rho(y_t|x_t)\tilde{\pi}_{t|t-1}^N(dx_t)\, dx_t} \\ &= \frac{\sum_{i=1}^N \rho(y_t|\tilde{x}_t^i)\delta_{\tilde{x}_t^i}(d\tilde{x}_t)}{\sum_{i=1}^N \rho(y_t|\tilde{x}_t^i)} = \sum_{i=1}^{N} w_t^i \delta_{\tilde{x}_t^i}(d\tilde{x}_t), \end{aligned} \tag{12}$$

where $\sum_{i=1}^N w_t^i = 1$ and w_t^i are called the importance weights. To avoid the degeneracy problem, particles are selected according to a resampling step that samples N particles from the empirical distribution, $\tilde{\pi}^N_{t|t}(dx_t)$, and resets the weights to $\frac{1}{N}$. We then have the empirical distribution approximating the posterior at time t given by

$$\pi^N_{t|t}(dx_t) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^i_t}(dx_t). \tag{13}$$

B. The Markov Chain Distributed Particle Filter (MCDPF)

The main difference between the CPF and DPF is that the CPF has a central unit to collect the entire measurements from all nodes and update particles using all whole measurements simultaneously. During the process of data collection, the CPF might suffer from the bottlenecks in information flow. On the other hand, a DPF can overcome this problem by passing information only locally between connected nodes. If we have m nodes measuring the partial observations independently, then we can decompose the general state space model (4) as follows.

$$x_{t+1} = f(x_t, w_t) \tag{14}$$

$$\begin{bmatrix} y_{1,t} \\ y_{2,t} \\ \vdots \\ y_{m,t} \end{bmatrix} = \begin{bmatrix} g_1(x_t, v_{1,t}) \\ g_2(x_t, v_{2,t}) \\ \vdots \\ g_m(x_t, v_{m,t}) \end{bmatrix}. \tag{15}$$

Here $x_t \in \mathbf{R}^n$, $y_{i,t} \in \mathbf{R}^{p_i}$ with $\sum_{i=1}^m p_i = p$ and subscript i represents node i. In addition, the measurement noise at each nodes is assumed to be uncorrelated, $\mathbf{E}[v_t v_t^T] = \text{diag}(R_1, R_2, \ldots, R_m)$. Uncorrelated noise structure enables us to have conditionally independent measurements at each node, $y_{i,t}$, given the true state x_t. As a consequence of this assumption, the function $\rho(y_t|x^i_t)$ in (8) can be factorized by a product of $\rho_j(y_{j,t}|x^i_t)$ at each node,

$$\rho(y_t|x^i_t) = \prod_{j=1}^{m} \rho_j(y_{j,t}|x^i_t). \tag{16}$$

We propose a distributed particle filtering method using a random walk on the graph defined by the network topology. In the sensor network, node i measures the partial observation $y_{i,t}$ at time t and data at every node has to be fused to reach the global estimation of the true state. While achieving a global estimate by exchanging data, it is desirable to maintain a robustness with respect to the unexpected changes of global properties such as losing a node. The DPF proposed in this paper is robust since the information, consisting of particles and weights, is transferred only to the connected neighborhood of each node. In other words, every node only needs local information. Transferring particle data is inefficient for low-dimensional systems, but scales well (only linearly) with dimension size, as opposed to existing methods using GMM approximations of posterior distribution [9][2][8]. As briefly explained in section I, communicating raw data is more efficient in terms of bandwidth capacity for relatively high-order systems.

MCDPF moves particles around the network according to the Markov chain on the network defined by the normalized adjacency matrix $\mathcal{A}$ to compute the importance weights. The main idea is that each particle gains $\rho_i(y_{i,t}|x_t)$ exponentially proportional to the expected number of visit to node i. Suppose we have the graph $G = (V, E)$ based on the sensor network and the normalized adjacency matrix $\mathcal{A}$. In the MCDPF setting, the Markov chain is run k steps on every particle after the prediction step and the number of visits to the i-th node is defined by $M(i,k)$. Considering the number of visits to each node, each particle multiplies $\rho_i(y_{i,t}|x_t)^{\frac{2|E(G)|}{kd(i)}}$ to its previous weight every time it visits the i-th node. If we have N particles after k Markov chain steps at a node, then the posterior distribution of the MCDPF is given as follows.

$$\begin{aligned} \tilde{\pi}^N_{t|t,k}(dx_t) &= \frac{\sum_{i=1}^N \prod_{j=1}^m \rho_j(y_{j,t}|\tilde{x}^i_t)^{\frac{2|E(G)|}{kd(j)}\times M(j,k)} \delta_{\tilde{x}^i_t}(d\tilde{x}_t)}{\sum_{i=1}^N \prod_{j=1}^m \rho_j(y_{j,t}|\tilde{x}^i_t)^{\frac{2|E(G)|}{kd(j)}\times M(j,k)}} \\ &= \sum_{i=1}^N w^i_{t,k}\delta_{\tilde{x}^i_t}(d\tilde{x}_t). \end{aligned} \tag{17}$$

The MCDPF is defined in algorithm 1 below. We use the notation $x^i_{j,t}$ for the i-th particle of node j at time t and $N(j)$ for the number of particle at node j. Also $\mathcal{I}_{i\to j}$ is the indices of particles moving from node i to j in the current Markov chain step and we recall that $\mathcal{A}$ is the adjacency matrix of the network.

Algorithm 1 Markov Chain Distributed Particle Filter (MCDPF)

```
Initialization:
  ${x^i_{j,0}}_{i=1}^N \sim p(x_0)$, ${w^i_{j,0}}_{i=1}^N = \frac{1}{N}$ for $j = 1,\ldots,m$
Importance Sampling: For $j = 1,\ldots,m$
  $\{\tilde{x}^i_{j,t}\}_{i=1}^{N(j)} \sim p\left(x_t|\{x^i_{j,t-1}\}_{i=1}^{N(j)}\right)$, $\{\tilde{w}^i_{j,t}\}_{i=1}^{N(j)} = 1$
  for $k$ iterations do
    Move $\{\tilde{x}^i_{\cdot,t}\}_{i=1}^{N(j)}$, $\{\tilde{w}^i_{\cdot,t}\}_{i=1}^{N(j)}$ according to matrix $\mathcal{A}$
    for $j = 1$ to $m$ do
      $\{\tilde{x}^i_{j,t}\}_{i=1}^{N(j)} = \bigcup_{l\in N_j}\{\tilde{x}^i_{l,t}\}_{i\in\mathcal{I}_{l\to j}}$
      $\{\tilde{w}^i_{j,t}\}_{i=1}^{N(j)} = \bigcup_{l\in N_j}\{\tilde{w}^i_{l,t}\}_{i\in\mathcal{I}_{l\to j}}$
      $\{\tilde{w}^i_{j,t}\}_{i=1}^{N(j)} \leftarrow \{\tilde{w}^i_{j,t}\}_{i=1}^{N(j)} \times \rho_j(y_{j,t}|\{\tilde{x}^i_{j,t}\}_{i=1}^{N(j)})^{\frac{2|E(G)|}{kd(i)}}$
    end for
  end for
Resample: For $j = 1,\ldots,m$
  Resample $\{x^i_{j,t}\}_{i=1}^{N(j)}$ according to $\{\tilde{w}^i_{j,t}\}_{i=1}^{N(j)}$ and set
  weights $\{w^i_{j,t}\}_{i=1}^{N(j)} = \frac{1}{N(j)}$
```

C. Convergence to CPF and Algorithm

We will show that the empirical posterior distribution of the MCDPF converges weakly to that of the CPF as the Markov chain steps k per measurement goes to infinity. The notation that will be used throughout the proof mainly follows that of [10]. In the stochastic filtering problem, functions a_t and b_t are defined on a metric space (E, d) to itself and are considered as continuous maps from $\pi_{t|t-1} \to \pi_{t|t}$ and $\pi_{t-1|t-1} \to \pi_{t|t-1}$, respectively. Additionally, a^k_t is also a continuous function mapping $\pi_{t|t-1} \to \pi_{t|t,k}$, given by

$$a_t^k(p(x_t|y_{1:t-1})) = \frac{\prod_{j=1}^m \rho_j(y_{j,t}|x_t)^{\frac{2|E(G)|}{kd(j)}\times M(j,k)} p(x_t|y_{1:t-1})}{\int_{\mathbf{R}^n} \prod_{j=1}^m \rho_j(y_{j,t}|x_t)^{\frac{2|E(G)|}{kd(j)}M(j,k)} p(x_t|y_{1:t-1})\,dx_t}. \tag{18}$$

The perturbation c^N is defined as a function that maps from a measure ν to a random sample size of size N of the measure, so that

$$c^{N,w}(\nu) = \frac{1}{N}\sum_{j=1}^N \delta_{\{V_j(w)\}}, \tag{19}$$

where $V_j : \Omega \to \mathbf{R}^n$ is an IID random variable with the distribution ν and $w \in \Omega$. For notational simplicity, let $h_t^N, h_{1:t}^N$ be defined as the composition of functions a_t, b_t, c^N as follows.

$$h_t^N \triangleq c^N \circ a_t \circ c^N \circ b_t, \qquad h_{1:t}^N \triangleq h_t^N \circ \cdots \circ h_1^N \tag{20}$$

$$h_{t,k}^N \triangleq c^N \circ a_t^k \circ c^N \circ b_t, \quad h_{1:t,k}^N \triangleq h_{t,k}^N \circ \cdots \circ h_{1,k}^N. \tag{21}$$

Thus the posterior distribution of CPF and MCDPF at time t can then be expressed as

$$\pi_{t|t}^N = h_t^N(\pi_{t-1|t-1}^N) = h_{1:t}^N(\pi_0) \tag{22}$$

$$\pi_{t|t,k}^N = h_{t,k}^N(\pi_{t-1|t-1,k}^N) = h_{1:t,k}^N(\pi_0). \tag{23}$$

To prove $\lim_{k\to\infty} \pi_{t|t,k}^N = \pi_{t|t}^N$, several lemmas are reviewed here.

Lemma 3.1: Let (E,d) be a metric space with functions $a_t^k, a_t, b_t : E \to E$ such that $\lim_{k\to\infty} a_t^k = a_t$ pointwise for each t. Then

$$\lim_{k\to\infty} h_{1:t,k}^N = h_{1:t}^N. \tag{24}$$

pointwise for each t and N.

Proof: For $e \in E$ and arbitrary t, we have $c^N(b_t(e)) \in E$. Since we assumed pointwise convergence of a_t^k to a_t, for all $\epsilon > 0$ there exists $k(e,\varepsilon)$ such that for $k > k(e,\varepsilon)$,

$$\|a_t^k(c^N(b_t(e))) - a_t(c^N(b_t(e)))\| < \varepsilon, \tag{25}$$

where $\|\cdot\|$ is the supremum norm on functions from (E,d) to itself. Equivalently, $\lim_{k\to\infty} h_{t,k}^N = h_t^N$ pointwise for all t. By induction over t we have (24). ■

Lemma 3.2: For the MCDPF and CPF as defined above,

$$\lim_{k\to\infty} a_t^k = a_t \tag{26}$$

pointwise for all t.

Proof: For any $e \in E$,

$$\begin{aligned}
&\lim_{k\to\infty} \|a_t^k(e) - a_t(e)\| \\
&= \lim_{k\to\infty} \left\| \frac{\prod_{j=1}^m \rho_j(y_{j,t}|x_t)^{\frac{2|E(G)|}{kd(j)}\times M(j,k)} e(x_t)}{\int_{\mathbf{R}^n} \prod_{j=1}^m \rho_j(y_{j,t}|x_t)^{\frac{2|E(G)|}{kd(j)}M(j,k)} e(dx_t)} - \frac{\rho(y_t|x_t)e(x_t)}{\int_{\mathbf{R}^n}\rho(y_t|x_t)e(dx_t)} \right\| \\
&= \left\| \frac{\prod_{j=1}^m \rho_j(y_{j,t}|x_t)e(x_t)}{\int_{\mathbf{R}^n}\prod_{j=1}^m \rho_j(y_{j,t}|x_t)e(dx_t)} - \frac{\rho(y_t|x_t)e(x_t)}{\int_{\mathbf{R}^n}\rho(y_t|x_t)e(dx_t)} \right\| \\
&= 0.
\end{aligned} \tag{27}$$

The first equality is due to proposition 2.1 and the second equality comes from conditional independence of the measurements at each node. ■

Theorem 3.3: Consider a connected sensor network with measurements at different nodes conditionally independent given the true state. Then the estimated distribution of the MCDPF in Algorithm 1 converges weakly to the estimated distribution of the CPF as the number of Markov chain steps k per measurement goes to infinity. That is,

$$\lim_{k\to\infty} \pi_{t|t,k}^N = \pi_{t|t}^N. \tag{28}$$

pointwise.

Proof: Combining lemmas 3.1 and 3.2 with the optimal Bayesian filtering functions a_t, b_t and a_t^k gives (28). ■

D. Convergence to Optimal Filtering

So far we showed that the MCDPF converges weakly to the CPF. The next step is to prove the convergence of MCDPF to the optimal filtering distribution as $N \to \infty$ as well as $k \to \infty$. The convergence of the classical particle filter to optimal filtering distribution is shown in [10][12]. The only difference between the MCDPF and the standard particle filter used in the proof is the function a_t^k, which needs to satisfy the following condition to ensure the convergence of MCDPF to optimal filtering. For all sequences $e_N \to e \in E$ we have

$$\lim_{N\to\infty}\lim_{k\to\infty} a_t^k(e_N) = a_t(e). \tag{29}$$

This property is not obvious because the function a_t^k converges only pointwise to a function a_t. Fortunately, however, a_t is continuous, which is sufficient to give (29), as we see from the following lemma.

Lemma 3.4: Suppose (E,d) is a metric space and $a^k, a : E \to E$ are continuous functions so that a^k converges pointwise to a as $k \to \infty$. For a convergent sequence $\lim_{N\to\infty} e_N = e \in E$, we have

$$\lim_{N\to\infty}\lim_{k\to\infty} a^k(e_N) = \lim_{k\to\infty}\lim_{N\to\infty} a^k(e_N) = a(e). \tag{30}$$

Proof: For functions $a^k \to a$ pointwise and $e_N \to e \in E$, we have

$$\lim_{k\to\infty} a^k(e_N) = a(e_N) \text{ for all } N \tag{31}$$

$$\Rightarrow \lim_{N\to\infty}\lim_{k\to\infty} a^k(e_N) = \lim_{N\to\infty} a(e_N) = a(e), \tag{32}$$

since a is continuous. Conversely, continuity of a^k gives

$$\lim_{N\to\infty} a^k(e_N) = a^k(e) \text{ for all } k \tag{33}$$

$$\Rightarrow \lim_{k\to\infty}\lim_{N\to\infty} a^k(e_N) = \lim_{k\to\infty} a^k(e) = a(e). \tag{34}$$

■

Lemma 3.5: Consider a_t, b_t from (8), a_t^k from (18), and c^N from (8). Assume that the sequence a_t^k satisfies the property (29) for each t. Then we have

$$\lim_{N\to\infty}\lim_{k\to\infty} h_{1:t,k}^N = h_{1:t}. \tag{35}$$

Moreover for all sequences $e^N \to e \in E$ we have

$$\lim_{N\to\infty}\lim_{k\to\infty} h_{1:t,k}^N(e_N) = h_{1:t}(e). \tag{36}$$

Proof: From [10, Lemma 2], for all $e_N \to e \in E$, c^N satisfies

$$\lim_{N\to\infty} c^N(e_N) = e. \tag{37}$$

Thus, for all $e_N \to e \in E$ and any continuous function b_t,

$$\lim_{N\to\infty} c^N(b_t(e_N)) = b_t(e). \tag{38}$$

From the property (29),

$$\lim_{N\to\infty} c^N(b_t(e_N)) = b_t(e) \tag{39}$$

$$\Rightarrow \lim_{N\to\infty}\lim_{k\to\infty} a_t^k(c^N(b_t(e_N))) = a_t(b_t(e)). \tag{40}$$

And again from the property (37) of c^N,

$$\lim_{N\to\infty}\lim_{k\to\infty} a_t^k(c^N(b_t(e_N))) = a_t(b_t(e)) \tag{41}$$

$$\Rightarrow \lim_{N\to\infty}\lim_{k\to\infty} c^N(a_t^k(c^N(b_t(e_N)))) = a_t(b_t(e)). \tag{42}$$

Thus we have $\lim_{N\to\infty}\lim_{k\to\infty} h_{t,k}^N(e_N) = h_t(e)$ and from induction over t we can conclude $\lim_{N\to\infty}\lim_{k\to\infty} h_{1:t,k}^N(e_N) = h_{1:t}(e)$. ■

Recall that a kernel K is said to have the Feller property if $K\varphi$ is a continuous bounded function whenever φ is a continuous bounded function. For such kernels we have the following result.

Lemma 3.6: Suppose a_t and b_t are functions defined in (8). Then a_t is continuous provided the function $\rho(y_t|\cdot)$ is bounded, continuous and strictly positive. Furthermore, b_t is a continuous function if the transition kernel K is Feller.

Proof: See [10, Section IV.B.]. ■

Putting all the above lemmas together gives the following main result.

Theorem 3.7: Assume that the kernel K is Feller and the function ρ is bounded, continuous and strictly positive. Then the estimated distribution of the MCDPF in Algorithm 1 converges to the optimal filtering distribution as the number of particles N and the number of Markov chain steps k per measurement go to infinity:

$$\lim_{N\to\infty}\lim_{k\to\infty} \pi_{t|t,k}^N = \pi_{t|t}. \tag{43}$$

Proof: For the initial probability measure μ_0, we know that $\lim_{N\to\infty} \mu_0^N = \mu_0$. From lemma 3.5,

$$\lim_{N\to\infty}\lim_{k\to\infty} \pi_{t|t,k}^N = \lim_{N\to\infty}\lim_{k\to\infty} h_{1:t,k}^N(\mu_0^N) \tag{44}$$

$$= h_{1:t}(\mu_0) = \pi_{t|t}, \tag{45}$$

giving the desired result. ■

IV. Numerical Example

We consider a bearing-only tracking example in this section for the purpose of demonstrating the performance of the MCDPF. The dynamic model was a time-dependent linear system but the measurement model was nonlinear, with one moving target tracked by 4 bearing sensors linked in a network. There were two modes for the movement of the target, straight and turning mode. The target moved with linear dynamics, turning to the right by 90 degrees between 0.5 and 1, 2 and 2.5, and 3.5 and 4 seconds. The state vector is $[x_t \ y_t \ \dot{x}_t \ \dot{y}_t]$ and the state-space system and measurements at each sensor were given by

$$x_{t+1} = e^{F_t \Delta t} x_t + q_t, \tag{46}$$

$$\theta_t^i = \arctan\left(\frac{y_t - s^i(y)}{x_t - s^i(x)}\right) + r_t^i, \tag{47}$$

where

$$F_t = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & a_t \\ 0 & 0 & -a_t & 0 \end{bmatrix}, \ a_t = \begin{cases} 0 & \text{Straight mode} \\ \frac{\pi}{2\times 51\Delta t} & \text{Turning mode} \end{cases}$$

$$r_t^i \sim \mathcal{N}(0, 0.05^2), \ q_t \sim \mathcal{N}\left(0, \begin{bmatrix} \frac{\Delta t^3}{3} & 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & \frac{\Delta t^3}{3} & 0 & \frac{\Delta t^2}{2} \\ \frac{\Delta t^2}{2} & 0 & \Delta t & 0 \\ 0 & \frac{\Delta t^2}{2} & 0 & \Delta t \end{bmatrix}\right).$$

Here $\Delta t = 0.01$, $(s^i(x), s^i(y))$ was the position of ith sensor, and r_t^i was the measurement noise. Each sensor was connected to its nearest two neighboring sensors. The true trajectory of the moving target was estimated by the CPF and MCDPF. The centralized particle filter tracked the true trajectory with $N = 400$ particles and bearing information gathered from all four different sensors. The trajectory was also estimated by the MCDPF with $N = 400$ particles at each node.

Figures 1 and 2 show the estimation results of the CPF and MCDPF with $k = 4$ Markov chain steps per measurement for the MCDPF. Even with such a small number of Markov chain steps per measurement, the MCDPF at each sensors obtained a reasonable estimate of the true trajectory by exchanging information only with connected neighbor according to a random walk of particles and weights on the sensor network.

To numerically study the convergence of the posterior distribution of the MCDPF at each node to the posterior distribution of the CPF (as proved in theorem 3.3), we define the root mean square error (RMSE) to be

$$\mathrm{RMSE}(\hat{x}_{\mathrm{DPF}}) = \left(\mathbf{E}\left[\|\hat{x}_{\mathrm{DPF}} - \hat{x}_{\mathrm{opt}}\|^2\right]\right)^{1/2} \tag{48}$$

$$\approx \left(\mathbf{E}\left[\|\hat{x}_{\mathrm{DPF}} - \hat{x}_{\mathrm{CPF}}\|^2\right]\right)^{1/2}. \tag{49}$$

Figure 3 shows the RMSE versus the number of Markov Chain steps k per measurement, computed by averaging 2×10^5 executions of the CPF and MCDPF at time $t = 0.1$ in the

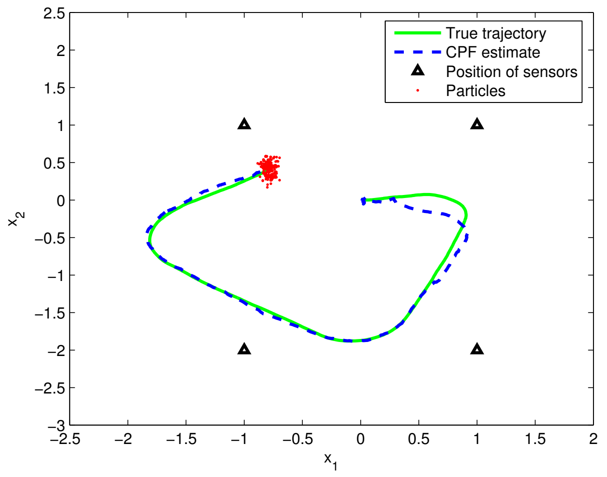

Fig. 1. The trajectory estimation by CPF.

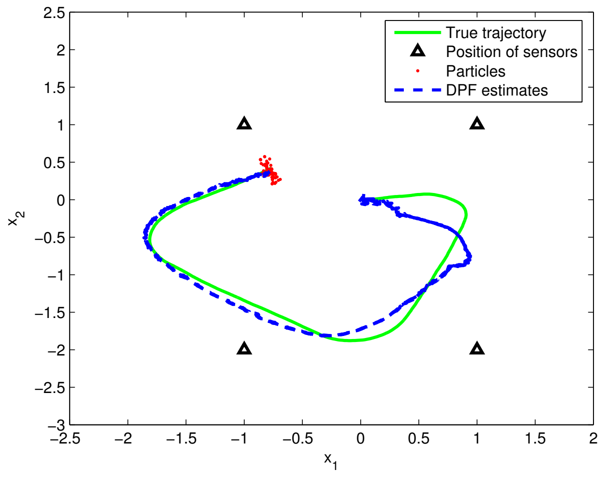

Fig. 2. The trajectory estimation by DPF with Markov chain steps $k = 4$.

simulation. This figure confirms numerically the convergence proved in Theorem 3.3. Based on Figure 3 it appears that the RMSE converges at a rate of $k^{1/2}$ (see [13] for a proof).

V. Conclusions and Future Work

In this paper, we introduced the new Markov Chain Distributed Particle Filter (MCDPF) which exchanges particles between distributed sensor nodes, thus providing robust state estimation and good scaling in the dimension of the system being estimated. The robustness is maintained by the fact that the proposed MCDPF algorithm only exchanges information locally, while good scaling is due to the use of only particles to store state. This does mean, however, that the MCDPF will be less efficient for lower-dimensional systems that can be efficiently represented by Gaussian Mixture Models or other similar approximations. We proved that the estimated distribution of the MCDPF algorithm converges weakly to that of the classical Centralized Particle Filter (CPF) as the

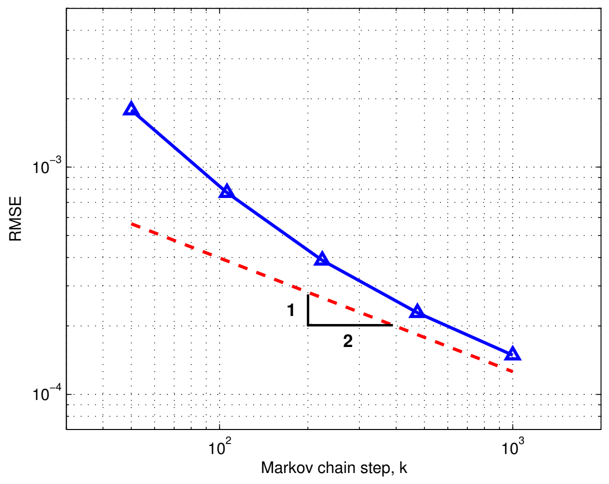

Fig. 3. Convergence of RMSE in number of Markov chain steps k.

number of Markov chain steps k per measurement tends to infinity, and we presented a numerical example to verify this.

References

[1] M. Rosencrantz and G. Gordon and S. Thrun, "Decentralized sensor fusion with distributed particle filters", *In Proc. of UAI*, 2003.

[2] D. Gu, "Distributed Particle Filter for Target Tracking", *Robotics and Automation, 2007 IEEE International Conference on*, 2007, pp. 3856-3861.

[3] B.S. Rao and H.F. Durrant-Whyte, Fully decentralised algorithm for multisensor Kalman filtering, *Control Theory and Applications, IEEE Proceedings D*, 1999, pp 413-420.

[4] Olfati-Saber, R., "Distributed Kalman Filter with Embedded Consensus Filters", *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC '05. 44th IEEE Conference on*, 2005, pp. 8179-8184.

[5] Olfati-Saber, R., "Distributed Kalman filtering for sensor networks", *Decision and Control, 2007 46th IEEE Conference on*, 2007, pp. 5492-5498.

[6] L. Ong and B. Upcroft and M. Ridley and T. Bailey and S. Sukkarieh and H. Durrant-Whyte, "Consistent methods for Decentralised Data Fusion using Particle Filters", *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, 2006, pp. 85-91.

[7] M. Coates, "Distributed particle filters for sensor networks", *In Proc. of 3nd workshop on Information Processing in Sensor Networks*, 2004, pp. 99-107.

[8] X. Sheng and Y. Hu and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network", *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, LA, CA, 2005, pp. 24.

[9] D. Gu and J. Sun and Z. Hu and H. Li, "Consensus based distributed particle filter in sensor networks", *Information and Automation, 2008. ICIA 2008. International Conference on*, 2008, pp. 302-307.

[10] Crisan, D. and Doucet, A., A survey of convergence results on particle filtering methods for practitioners, *IEEE Transactions on Signal Processing*, vol. 50, 2002, pp 736-746.

[11] Doucet, A. and De Freitas, N. and Gordon, N., *Sequential Monte Carlo methods in practice*, Springer-Verlag, 2001.

[12] Hu, Xiao-Li and Schon, T. B. and Ljung, L., A Basic Convergence Result for Particle Filtering, *IEEE Transactions on Signal Processing*, vol. 56, 2008, pp 1337-1348.

[13] S. Lee and M. West, Convergence of Markov Chain Distributed Particle Filters, *in preparation*.