2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

Distributed Particle Filter for Target Tracking

Dongbing Gu

Abstract— **This paper investigates target tracking using a distributed particle filter over sensor networks. Gaussian mixture model is adopted to approximate the posterior distribution of weighted particles in this distributed particle filter. The parameters of Gaussian mixture model are exchanged between neighbor sensor nodes. Each node can obtain the Gaussian mixture model representing particle's posterior distribution through the parameter exchange. With the posterior distribution, the distributed particle filter can draw particles from it, predicted particles and observations, update particle weights, and re-sample particles based the predicted weights. The parameter exchange is key to implement the distributed operation. It is implemented by using an average consensus filter. Through this consensus filter, each sensor node can gradually diffuse its local statistics of weighted particles over the entire network and asymptotically obtain the estimated global statistics. The parameters of Gaussian mixture model can be calculated by using the estimated global statistics. Because the average consensus filter only requires that each sensor node communicate with its neighbors, the proposed distributed particle filter is scalable and robust. Simulations of tracking tasks in a sensor network with 100 sensor nodes are given.**

I. INTRODUCTION

One of the major goals in sensor networks is to detect and track changes in the monitored environment[1][2]. Particle filter is one of the widely used tracking algorithms due to its applicability to non-linear and non-Gaussian dynamic systems [3][4]. Usually the energy cost related to computation in each sensor node and communication between sensor nodes is significant when using such algorithm in sensor networks. Reducing the energy cost in computation and communication can significantly increase the node lifespan[5].

Currently there are several distributed particle filters $(DPFs)$ that have been developed [6][7][8][9]. In these algorithms, the distributed nature is achieved by either transmitting local statistics of particles to a centralized unit or using the message passing method. Transmitting local statistics of particles to a centralized unit is not an efficient approach. It is also not robust. Failure of the centralized unit is vital to the entire network. In the message passing method, the algorithms construct a path through the networks, which passes through all nodes. Global statistics of particles are accumulated by adding local statistics in each node through a forward pass. Then there needs a backward pass, which runs the important sampling and selection steps in each sensor node by using the accumulated global statistics. In [6], the factorized likelihood function is used and each partial likelihood function is updated at individual sensors using only local observations and partial likelihood function estimated in the preceding sensors. The partial likelihood function is represented by a parameterized model and the parameters are transmitted through the path. The same strategy to communicate the highly compact distribution is also used in robotics for map building in [10]. In [7], a set of uncorrelated sensor cliques is used and they are automatically constructed according to moving target trajectories. The algorithm uses a low dimensional Gaussian mixture model (GMM) to describe the posterior pdf. Model parameters rather than weighted particles are transmitted over the network. Using a GMM to approximate the posterior distribution is also adopted in [8] where the estimated parameters of GMM are transmitted to a fusion center. In [9], the particles are distributed in a sensor network, i.e. each sensor node holds part of particles. Local statistics of particles are calculated and transmitted to a centralized unit.

In this paper, we propose to distribute the whole particle set evenly across the entire network and use GMM to approximate the posterior distribution in each node. We also propose to exchange local statistics of particles between neighbor nodes to estimate global statistics of all particles. Because the calculations of global statistics are in the average form, global statistics can be estimated by using an average consensus filter. The consensus filter can diffuse local statistics over the entire network through communication with neighbor nodes [11] [12][13] and estimate global statistics using local statistics and neighbor's local statistics. Based on the estimated global statistics, an Expectation and Maximization (EM) algorithm is developed to estimate the GMM. Using the estimated GMM, each node can predict the next step particles, update weights and re-sample particles. The consensus filter only requires local communication, i.e. each node only needs to communicate with its neighbors and gradually gains global statistics. Thus, this distributed algorithm is scalable. It is also robust. Failures of any nodes do not affect the algorithm performance given the network is still connected. The estimated results can be accessed from any nodes in the network.

In the rest of this paper, a centralized particle filter in sensor network environment is described in Section II. Section III presents the EM algorithm to estimate the parameters of GMM. The average consensus filter and distributed particle filter are given in Section IV. Section V provides simulation results. Finally, conclusions are summarized in Section VI.

II. PARTICLE FILTER TRACKING IN SENSOR NETWORKS

We consider a network of M sensors, which is used to track a moving object. The moving object is modeled by a

Dongbing Gu is with Department of Computer Science, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, United Kingdom dgu@essex.ac.uk

discrete state equation:

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}) + \mathbf{v}_{k-1} \tag{1}$$

where $\mathbf{x}_k$ is the n_x dimensional state vector and $\mathbf{v}_k$ is the Gaussian noise with mean zero and covariance Q_k. The state is also modeled as a Markov process with initial distribution $p(\mathbf{x}_0) \sim \mathcal{N}(\mathbf{x}_0, Q_0)$ and state transition probability $p(\mathbf{x}_k|\mathbf{x}_{k-1})$.

Each sensor m can make a n_z dimensional observation $\mathbf{z}_{m,k}(m = 1, \ldots, M)$ at time k. The observation state equation is assumed as follows:

$$\mathbf{z}_{m,k} = \mathbf{h}_m(\mathbf{x}_k) + \mathbf{w}_k \tag{2}$$

where $\mathbf{w}_k$ is the Gaussian noise with mean zero and covariance R_k. The observation equation (2) is also modeled as a likelihood function $p(\mathbf{z}_{m,k}|\mathbf{x}_k)$. It is assumed that the state noise and observation noise are independent, $E[\mathbf{v}_k\mathbf{w}_k^T] = 0$.

We build up a centralized particle filter (CPF) for the tracking purpose first in this section. Let $\mathbf{x}_{0:k}$ denote $\{\mathbf{x}_t, t = 0, \ldots, k\}$ and $\mathbf{z}_{m,1:k}$ denote $\{\mathbf{z}_{m,t}, t = 0, \ldots, k\}$. Then, the tracking purpose of a particle filter is to estimate the posterior *pdf* $p(\mathbf{x}_{0:k}|\mathbf{z}_{m,1:k})$, or $p(\mathbf{x}_k|\mathbf{z}_{m,1:k})$.

In CPF, observations from all sensors are transmitted to a centralized unit. It is assumed that the centralized unit receives only one observation from one sensor at each k, The centralized unit maintains a set of weighted particles $(\mathbf{x}_k^{(n)}, \omega_k^{(n)})$. When the centralized unit receives an observation $\mathbf{z}_{m,k}$ at k, it uses the CPF to predict particles $\mathbf{x}_k^{(n)}$ and update weights $\omega_k^{(n)}, (n = 1, \ldots, N)$, where N is the number of particles.

Let $\{\mathbf{x}_{0:k}^{(n)}, \omega_k^{(n)}\}_{n=1}^N$ denote a random measure that characterizes the posterior *pdf* $p(\mathbf{x}_{0:k}|\mathbf{z}_{m,1:k})$, where the weights are normalized, $\sum_{n=1}^N \omega_k^{(n)} = 1$. The posterior *pdf* at k can be approximated by

$$p(\mathbf{x}_{0:k}|\mathbf{z}_{m,1:k}) \approx \sum_{n=1}^N \omega_k^{(n)} \delta(\mathbf{x}_{0:k} - \mathbf{x}_{0:k}^{(n)}) \tag{3}$$

where δ denotes the Dirac delta function. Expectation of a function $g(\mathbf{x}_k)$ can be approximated by

$$\begin{aligned} E[g(\mathbf{x}_{0:k})] &\approx \sum_{n=1}^N g(\mathbf{x}_{0:k}^{(n)}) p(\mathbf{x}_{0:k}^{(n)}|\mathbf{z}_{m,1:k}) \\ &= \sum_{n=1}^N \omega_k^{(n)} g(\mathbf{x}_{0:k}^{(n)}) \end{aligned} \tag{4}$$

Since it is often impossible to sample directly from the posterior *pdf*, it is a normal practice to sample from a known proposal distribution $q(\mathbf{x}_{0:k}|\mathbf{z}_{m,1:k})$, $\mathbf{x}_{0:k}^{(n)} \sim q(\mathbf{x}_{0:k}^{(n)}|\mathbf{z}_{m,1:k})$. Therefore, the weights in (3) are defined to be

$$\omega_k^{(n)} \propto \frac{p(\mathbf{x}_{0:k}^{(n)}|\mathbf{z}_{m,1:k})}{q(\mathbf{x}_{0:k}^{(n)}|\mathbf{z}_{m,1:k})} \tag{5}$$

Assume the proposal distribution meets the following condition:

$$q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{m,1:k}) = q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{m,1:k}) \tag{6}$$

Then, we have

$$\begin{aligned} & q(\mathbf{x}_{0:k}|\mathbf{z}_{m,1:k}) \\ & = q(\mathbf{x}_k|\mathbf{x}_{0:k-1}, \mathbf{z}_{m,1:k}) q(\mathbf{x}_{0:k-1}|\mathbf{z}_{m,1:k-1}) \\ & = q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{m,1:k}) q(\mathbf{x}_{0:k-1}|\mathbf{z}_{m,1:k-1}) \end{aligned} \tag{7}$$

The weight updating equation (5) can be rewritten in a recursive form:

$$\omega_k^{(n)} \propto \omega_{k-1}^{(n)} \frac{p(\mathbf{z}_{m,k}|\mathbf{x}_k^{(n)}) p(\mathbf{x}_k^{(n)}|\mathbf{x}_{k-1}^{(n)})}{q(\mathbf{x}_k^{(n)}|\mathbf{x}_{k-1}^{(n)}, \mathbf{z}_{m,k})} \tag{8}$$

Finally, the filtering posterior *pdf* $p(\mathbf{x}_k|\mathbf{z}_{m,1:k})$ can be approximated as

$$p(\mathbf{x}_k|\mathbf{z}_{m,1:k}) \approx \omega_k^{(n)} \delta(\mathbf{x}_k - \mathbf{x}_k^{(n)}) \tag{9}$$

And expectation of a function $g(\mathbf{x}_k)$ can be approximated as follows:

$$E[g(\mathbf{x}_k)] \approx \sum_{n=1}^N \omega_k^{(n)} g(\mathbf{x}_k^{(n)}) \tag{10}$$

The choice of the proposal distribution is one of the important steps in particle filters. The most popular practice is to choose the state transition probability as the proposal distribution.

$$q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_{m,1:k}) = p(\mathbf{x}_k|\mathbf{x}_{k-1}) \tag{11}$$

This choice minimizes the variance of the importance weights [4]. By this choice, the weights updating equation (8) can be easily implemented:

$$\omega_k^{(n)} \propto \omega_{k-1}^{(n)} p(\mathbf{z}_{m,k}|\mathbf{x}_k^{(n)}) \tag{12}$$

III. EM ALGORITHM FOR GAUSSIAN MIXTURES

To implement a distributed particle filter (DPF), particles and weights are distributed over entire network. Each sensor should maintain N particles $\mathbf{x}_{m,k}^{(n)}$ and weights $\omega_{m,k}^{(n)}$. Ideally, particles and weights from all sensor nodes should represent the posterior distribution to be estimated.

The posterior distribution of particle filter is assumed to be a GMM with C mixture probabilities $\alpha_{m,c}, (c = 1, \ldots, C)$. The unobserved state of GMM is denoted as y and y_c represents $y = c$. For each unobserved state y_c, observation $\mathbf{z}_{m,k}$ follows a Gaussian distribution with mean μ_c and variance Σ_c:

$$p(\mathbf{z}_{m,k}|\mu_c, \Sigma_c) = \frac{1}{\sqrt{2\pi}\|\Sigma_c\|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{z}_{m,k}-\mu_c)^T \Sigma_c^{-1} (\mathbf{z}_{m,k}-\mu_c)} \tag{13}$$

The Gaussian mixture distribution for observation $\mathbf{z}_{m,k}$ is

$$p(\mathbf{z}_{m,k}|\theta) = \sum_{c=1}^C \alpha_{m,c} p(\mathbf{z}_{m,k}|\mu_c, \Sigma_c) \tag{14}$$

where θ is the set of the distribution parameters to be estimated, $\theta = \{\alpha_{m,c}, \mu_c, \Sigma_c; c = 1, \ldots, C, m = 1, \ldots, M\}$.

Assume all observed data from all nodes are sent to a centralized unit where a standard EM algorithm is used to estimate the parameter set θ. The log-likelihood for the observed data satisfies

$$\begin{aligned} L(\theta|\mathbf{z}) &= \log \prod_{m=1}^{M} \prod_{j=1}^{k} p(\mathbf{z}_{m,j}|\theta) \\ &= \sum_{m=1}^{M} \sum_{j=1}^{k} \log p(\mathbf{z}_{m,j}|\theta) \\ &= \sum_{m=1}^{M} \sum_{j=1}^{k} \log \left(\sum_{c=1}^{C} \alpha_{m,c} p(\mathbf{z}_{m,j}|\mu_c, \Sigma_c) \right) \end{aligned} \tag{15}$$

In the standard EM algorithm, given observation $\mathbf{z}$ and current parameter set θ^t where t is the time step between two consecutive sensor observations at k and $k+1$, the conditional expectation of joint distribution $p(z, \mathbf{y}|\theta)$ is defined as

$$Q(\theta, \theta^t) = E[\log p(\mathbf{z}, y|\theta)|\mathbf{z}, \theta^t] \tag{16}$$

As

$$\begin{aligned} p(\mathbf{z}_{m,k}, y_c|\theta) &= p(y_c|\mathbf{z}_{m,k}, \theta) p(\mathbf{z}_{m,k}|\mu_c, \Sigma_c) \\ &= \alpha_{m,k} p(\mathbf{z}_{m,k}|\mu_c, \Sigma_c) \end{aligned}$$

then, equation (16) can be rewritten as follows:

$$\begin{aligned} & Q(\theta, \theta^t) \\ =& \sum_{c=1}^{C} \sum_{m=1}^{M} \sum_{j=1}^{k} \log p(\mathbf{z}_{m,j}, y_c|\theta) p(y_c|\mathbf{z}_{m,j}, \theta^t) \\ =& \sum_{c=1}^{C} \sum_{m=1}^{M} \sum_{j=1}^{k} \log[\alpha_{m,c} p(\mathbf{z}_{m,j}|\mu_c, \Sigma_c)] p(y_c|\mathbf{z}_{m,j}, \theta^t) \end{aligned}$$

In the E step, the conditional expectation $Q(\theta, \theta^t)$ can be calculated by using the following equation:

$$\begin{aligned} \alpha_{m,k,c}^{t+1} &= p(y_c|\mathbf{z}_{m,k}, \theta^t) \\ &= \frac{\alpha_{m,c}^{t} p(\mathbf{z}_{m,k}|\mu_c^t, \Sigma_c^t)}{\sum_{i=1}^{C} \alpha_{m,i}^{t} p(\mathbf{z}_{m,k}|\mu_i^t, \Sigma_i^t)} \end{aligned} \tag{17}$$

In the M step, the parameter set is updated by maximizing

$$\theta^{t+1} = \arg\max_{\theta} Q(\theta, \theta^t) \tag{18}$$

The iteration algorithm for all parameters is

$$\begin{aligned} & \alpha_{m,c}^{t+1} = \frac{1}{k} \sum_{j=1}^{k} \alpha_{m,j,c}^{t+1} \\ & \mu_c^{t+1} = \frac{\sum_{m=1}^{M} \sum_{j=1}^{k} \alpha_{m,j,c}^{t+1} \mathbf{z}_{m,k}}{\sum_{m=1}^{M} \sum_{j=1}^{k} \alpha_{m,j,c}^{t+1}} \\ & \Sigma_c^{t+1} = \\ & \frac{\sum_{m=1}^{M} \sum_{j=1}^{k} \alpha_{m,j,c}^{t+1} (\mathbf{z}_{m,j} - \mu_c^{t+1})(\mathbf{z}_{m,j} - \mu_c^{t+1})^T}{\sum_{m=1}^{M} \sum_{j=1}^{k} \alpha_{m,j,c}^{t+1}} \end{aligned} \tag{19}$$

The iteration algorithm in (19) can be further written as a compact form:

$$\begin{aligned} \mu_c^{t+1} &= \frac{\sum_{m=1}^{M} a_{m,c}^{t+1}}{\sum_{m=1}^{M} k\alpha_{m,c}^{t+1}} \\ \Sigma_c^{t+1} &= \frac{\sum_{m=1}^{M} b_{m,c}^{t+1}}{\sum_{m=1}^{M} k\alpha_{m,c}^{t+1}} \end{aligned} \tag{20}$$

where the local statistics (or called local summary quantities) is defined as:

$$\begin{aligned} \alpha_{m,c}^{t} &= \frac{1}{k} \sum_{j=1}^{k} \alpha_{m,j,c}^{t} \\ a_{m,c}^{t} &= \sum_{j=1}^{k} \alpha_{m,j,c}^{t} \mathbf{z}_{m,k} \\ b_{m,c}^{t} &= \sum_{j=1}^{k} \alpha_{m,j,c}^{t} (\mathbf{z}_{m,j} - \mu_c^t)(\mathbf{z}_{m,j} - \mu_c^t)^T \end{aligned} \tag{21}$$

The global statistics (or called global summary quantities) can be defined as:

$$\begin{aligned} \alpha_c^t &= \sum_{m=1}^{M} k\alpha_{m,c}^{t} \\ a_c^t &= \sum_{m=1}^{M} a_{m,c}^{t} \\ b_c^t &= \sum_{m=1}^{M} b_{m,c}^{t} \end{aligned} \tag{22}$$

Using the global summary quantities defined above, the estimated parameters are:

$$\begin{aligned} \mu_c^{t+1} &= \frac{a_c^{t+1}}{\alpha_c^{t+1}} \\ \Sigma_c^{t+1} &= \frac{b_c^{t+1}}{\alpha_c^{t+1}} \end{aligned} \tag{23}$$

IV. Distributed Particle Filter

In the EM algorithm mentioned above, it can be found that the local summary quantities can be calculated locally, while the global summary quantities can not be calculated locally. However, the global summary quantities can be viewed as the averages of the local summary quantities from all nodes in equation (22). This view can be made more clear by redefining the global summary quantities in (22) as the average as follows:

$$\begin{aligned} \alpha_c^t &= \frac{1}{M} \sum_{m=1}^{M} k\alpha_{m,c}^{t} \\ a_c^t &= \frac{1}{M} \sum_{m=1}^{M} a_{m,c}^{t} \\ b_c^t &= \frac{1}{M} \sum_{m=1}^{M} b_{m,c}^{t} \end{aligned} \tag{24}$$

This redefinition does not affect the parameter estimation in equation (23).

Due to the average expressions in (24), the idea of the average consensus filter proposed in [11][14] can be used to estimate the global summary quantities through information diffusion over the network. Each node exchanges the local summary quantities with its neighbors and estimates the global summary quantities based on neighbor's local summary quantities through the consensus filter.

Let $\bar{\alpha}^t_{m,c}, \bar{a}^t_{m,c}, \bar{b}^t_{m,c}$ denote the estimates of the global statistics α^t_c, a^t_c, b^t_c in node m. Let a vector $\zeta^t_{m,c}$ denote one of the estimates of the global statistics, $\bar{\alpha}^t_{m,c}$, $\bar{a}^t_{m,c}$, or $\bar{b}^t_{m,c}$. Let a vector $\mathbf{u}^t_{m,c}$ denote one of the local summary quantities $k\alpha^t_{m,c}$, $a^t_{m,c}$, or $b^t_{m,c}]^T$. The consensus filter in node m takes as inputs the local summary quantities $\mathbf{u}^t_{m,c}$. It outputs the estimated global summary quantities $\zeta^t_{m,c}$.

A sensor network can be modeled by using algebraic graph theory. A graph can be used to represent interconnections between sensor nodes. A vertex of the graph corresponds to a node and edges of the graph capture the dependence of interconnections. Formally, a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V} = \{v_1, ..., v_M\}$, indexed by nodes in the network, and a set of edges $\mathcal{E} = \{(v_i, v_j) \in \mathcal{V} \times \mathcal{V}\}$, containing unordered pairs of distinct vertices. Assuming the graph has no loops, i.e. $(v_i, v_j) \in \mathcal{E}$ implies $v_i \neq v_j$.

Let R denote the distance that a node can communicate via wireless radio links. Edge (v_i, v_j) is connected if the Euclidean distance d_{ij} between nodes i and j is less than or equal to R.

A graph is connected if for any vertices $(v_i, v_j) \in \mathcal{V}$, there exists a path of edges in $\mathcal{E}$ from v_i to v_j. The set of neighbors of vertex i is defined as $\mathcal{N}_i = \{j \in \mathcal{V} : (i,j) \in \mathcal{E}\}$. The degree of vertex i is defined as $d_i = |\mathcal{N}_i|$ and maximum degree is $d_{max} = \max_i d_i$. Let Δ be the degree matrix, $\Delta = diag(d_i)$. The adjacency matrix $\mathcal{A}$ is the integer matrix with rows and columns indexed by the vertices, such as the ij-entry of $\mathcal{A}$ is equal to the number of edges from i to j. Following [15], Laplacian matrix of a graph $\mathcal{G}$ is defined as L:

$$L = \Delta - \mathcal{A} \tag{25}$$

For a connected graph, Laplacian matrix L is symmetric and positive semi-definite. Its minimum eigenvalue is 0 and the corresponding eigenvector is $\mathbf{1} = [1, \ldots, 1]^T$ or $L\mathbf{1} = 0$ [15].

An average consensus filter in a sensor node m is designed as follows in the discrete form:

$$\zeta^{t+1}_{m,c} = \zeta^t_{m,c} + \epsilon \left[\sum_{j \in \mathcal{N}_m} (\zeta^t_{j,c} - \zeta^t_{m,c}) + (\mathbf{u}^t_{m,c} - \zeta^t_{m,c}) \right] \tag{26}$$

where ϵ is the updating rate and should be

$$\epsilon \leq \frac{1}{d_{max}} \tag{27}$$

This requirement guarantees the stability of the discrete consensus filter according to Gersgorin theorem. $\zeta_{m,c}$ can asymptotically converge to the average of local inputs $\mathbf{u}_{m,c}$:

$$\zeta_{m,c} \rightarrow \frac{1}{M} \sum_{m=1}^{M} \mathbf{u}_{m,c} \tag{28}$$

Since $\zeta_{m,c}$ represents the estimates of the global statistics and $\mathbf{u}_{m,c}$ represents the local statistics , we have:

$$\begin{aligned} \bar{\alpha}^t_{m,c} &\rightarrow \alpha^t_{m,c} = \sum_{m=1}^{M} k\alpha^t_{m,c} \\ \bar{a}^t_{m,c} &\rightarrow a^t_c = \sum_{m=1}^{M} a^t_{m,c} \\ \bar{b}^t_{m,c} &\rightarrow b^t_c = \sum_{m=1}^{M} b^t_{m,c} \end{aligned} \tag{29}$$

The estimated parameters are:

$$\begin{aligned} \bar{\mu}^{t+1}_c &= \frac{\bar{a}^t_{m,c}}{\bar{\alpha}^t_{m,c}} \\ \bar{\Sigma}^{t+1}_c &= \frac{\bar{b}^t_{m,c}}{\bar{\alpha}^t_{m,c}} \end{aligned} \tag{30}$$

Once the estimated global statistics are obtained, particles can be drawn from the Gaussian mixture distribution. Then they should be propagated through the state transition probability to generate the predicted particles. With the predicted particles, the weight updating step and re-sampling step are the same as the steps in the CPF.

From the observation equation (2), the predicted observations are $\mathbf{z}_{m,k}$:

$$\mathbf{z}_{m,k} = \mathbf{h}_m(\mathbf{x}_{m,k}) \tag{31}$$

Finally, the DPF is summarized in Algorithm 1.

V. Simulations

A sensor network with 100 nodes ($M = 100$) is used for simulation. The sensors are randomly placed in a square $5m \times 5m$ shown in figure 1. We take the communication distance in figure 1 as $R = 0.9m$. This results in a connected graph and its maximum degree is $d_{max} = 14$. The connectivity can be verified by finding that the rank of Laplacian matrix is 99 or $(M-1)$.

We assume that a moving object has a state equation:

$$\mathbf{x}_k = A\mathbf{x}_{k-1} + B\mathbf{v}_{k-1} \tag{32}$$

where $A = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} T^2/2 & 0 \\ 0 & T^2/2 \\ T & 0 \\ 0 & T \end{bmatrix}$.
The state $\mathbf{x}_k$ includes 2D coordinates and 2D speed vectors. T is the discrete sampling time and selected as 0.05. The observation equation is a sonar-like model, which can observe the distance and angle between a sensor positioned in $\mathbf{q}_m$ and the moving object.

$$\mathbf{z}_{m,k} = \begin{bmatrix} \|\mathbf{x}_{m,k} - \mathbf{q}_m\| \\ \gamma(\mathbf{x}_{m,k}, \mathbf{q}_m) \end{bmatrix} + \mathbf{w}_k \tag{33}$$

Algorithm 1 Distributed Particle Filter (DPF)

```
Initialization:
    Draw $N$ particles $\mathbf{x}_0^{(n)}$ from the prior $p(\mathbf{x}_0)$
    and $\omega_0^{(n)} = 1/N$
Importance sampling step:
    Calculate the local statistics
        $\mathbf{u}_{m,c}^t = [k\alpha_{m,c}^t, a_{m,c}^t, b_{m,c}^t]^T$ using (21)
    Estimate iteratively the global statistics
        $\bar{\alpha}_{m,c}^t, \bar{a}_{m,c}^t, \bar{b}_{m,c}^t$ using (26)
    Sample $N$ particles $\mathbf{x}_{m,k}^{(n)}$ from the estimated
        Gaussian mixtures
    Calculate the predicted particles $\mathbf{x}_{m,k}$
        using $\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1})$
    Calculate the predicted observations $\mathbf{z}_{m,k}$ using (31)
    Update the importance weights using
        $\omega_{m,k}^{(n)} = \omega_{m,k-1}^{(n)} p(\mathbf{z}_{m,k}|\mathbf{x}_{m,k}^{(n)})$
    Normalize the importance weights
        $\omega_{m,k}^{(n)} = \omega_{m,k}^{(n)} \left[\sum_{n=1}^{N} \omega_{m,k}^{(n)}\right]^{-1}$
Selection step:
    Re-sample $N$ particles $\mathbf{x}_{m,k}^{(n)}$ according to $\omega_{m,k}^{(n)}$
    and $\omega_{m,k}^{(n)} = \frac{1}{N}$
```

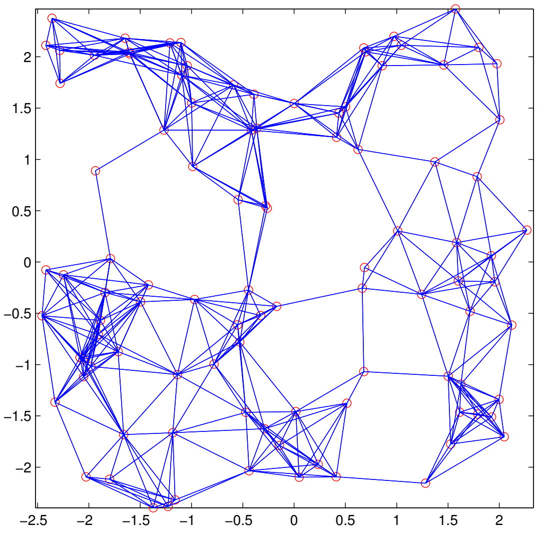

Fig. 1. Network connection

where $\|\mathbf{x}_{m,k} - \mathbf{q}_m\|$ presents the Euclidean distance between $\mathbf{x}_{m,k}$ and $\mathbf{q}_m$. $\gamma(\mathbf{x}_{m,k}, \mathbf{q}_m)$ represents the angle between $\mathbf{x}_{m,k}$ and $\mathbf{q}_m$. The covariances of Gaussian noises $\mathbf{v}_k$ and $\mathbf{w}_k$ are $Q_k = \begin{bmatrix} 0.25 & 0 \\ 0 & 0.25 \end{bmatrix}$ and $R_k = \begin{bmatrix} 0.01 & 0 \\ 0 & 0.0004 \end{bmatrix}$.

The posterior distribution GMM is assumed to have $C = 3$ mixture probabilities. The parameter estimation in the EM algorithm is executed 10 times between two consecutive observations.

The DPF is tested in 100 sensor nodes allocated as shown in figure 1. The target starts from $(0, 0)$. Each of sensor nodes contains 50 particles and the particles are initially distributed around $(0, 0)$. 200 step tracking result in one of the sensor nodes is shown in figure 2. It can be seen that the estimated trajectory (dotted line) is very close to the true trajectory (solid line). The estimated particles at the beginning and the end of tracking are also shown in figure 2 (see the black dot clusters). They represent the covariance changes from large cluster at the beginning to small cluster at the end.

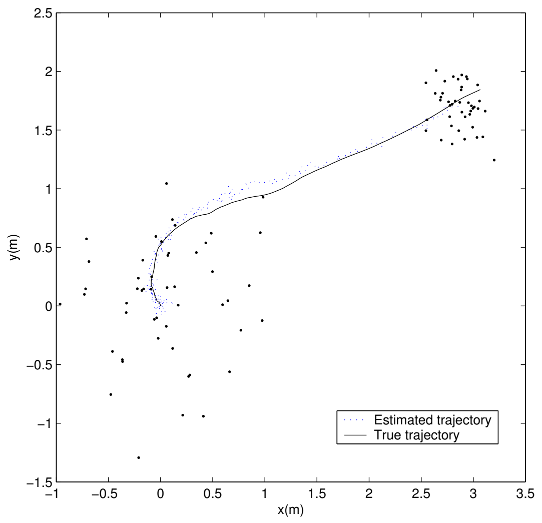

Fig. 2. Tracking result in one of the sensor nodes

Next the target moving along a circle trajectory is simulated. The circle trajectory is defined as follows:

$$x_k = 5\cos(0.01(k-1)), y_k = 5\sin(0.01(k-1)) \quad (34)$$

Its initial position is $(5, 0)$. The proposed DPF still uses the state equation (32) and the observation equation (33) to track this circle trajectory. Each of sensor nodes contains 50 particles and the particles are initially distributed around $(5, 0)$. All other parameters are the same as the first test. 600 step tracking result in one of the sensor nodes is shown in figure 3. It can be seen that the estimated trajectory (dotted line) can track the true trajectory (solid line). The covariance decreases from large cluster at the beginning to small cluster at the end.

The speeds in x and y directions during the tracking are shown in figure 4. They are both assumed to be zero at the beginning of the tracking. They asymmetrically converge to $\sin$ and $\cos$ functions required in the circle tracking.

VI. Conclusion

This paper presents a distributed particle filter for target tracking. The main idea is to use an average consensus filter to estimate global statistics of particles' posterior probability. Due to the use of the average consensus filter, this DPF is an approximation to the CPF. When the network connectivity is guaranteed and the updating rate of the consensus filter meets a condition, the estimated global statistics converge to the true values.

This DPF only needs information exchanges between neighbor sensor nodes. The global information can be diffused over the entire network through local information

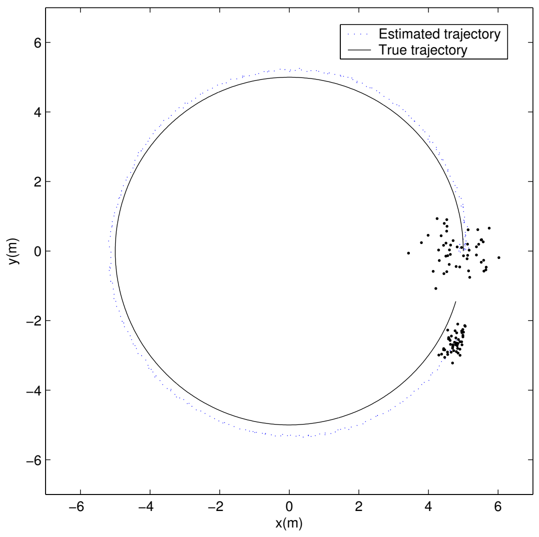

Fig. 3. Circle tracking result in one of the sensor nodes

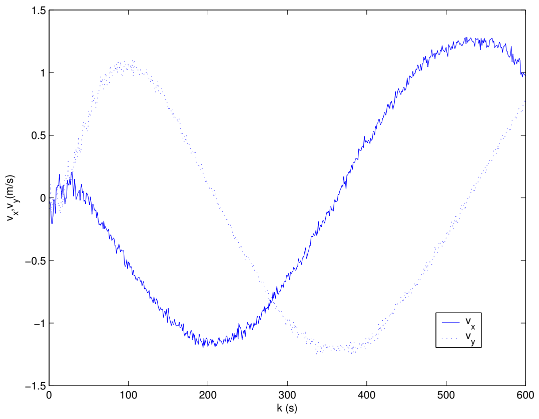

Fig. 4. Speed results of circle tracking in one of the sensor nodes

exchanges. It is scalable as the adding of more nodes does not affect the algorithm performance. It is also robust as it can still produce the right results even if failures of some nodes occur.

References

[1] F. Martinerie, "Data fusion and tracking using HMMs in a distributed sensor network," *IEEE Trans. Aerospace and Electronic Systems*, vol. 33, pp. 11–28, 1997.

[2] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration for tracking applications," *IEEE Signal Processing Magazine*, vol. 19, pp. 61–72, 2002.

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

[4] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo methods*. Springer-Verlag, New York, 2001.

[5] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 3, pp. 388–404, 2000.

[6] M. J. Coates, "Distributed particle filtering for sensor networks," in *Proc. of Int. Symp. Information Processing in Sensor Networks (IPSN)*, Berkeley, CA, April 2004.

[7] Y. Sheng, X. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor networks," in *Proc. of the 4th Int. Symposium on Information Processing in Sensor Networks*, Apr. 2005.

[8] L. Zuo, K. Mehrotra, P. Varshney, and C. Mohan, "Bandwidth-efficient target tracking in distributed sensor networks using particle filters," in *Proc. of 14th European Signal Processing Conference EURASIP2006*, Florence, Italy, Sept. 2006.

[9] A. S. Bashi, V. P. Jilkov, X. R. Li, and H. Chen, "Distributed implementations of particle filters," in *Proc. 2003 International Conf. Information Fusion*, Cairns, Australia, July 2003, pp. 1164–1171.

[10] M. Rosencrantz, G. Gordon, and S. Thrun, "Decentralized sensor fusion with distributed particle filters," in *Proc. of Conf. Uncertainty in Artificial Intelligence*, Acapulco, Mexico, Aug. 2003.

[11] N. A. Lynch, *Distributed Algorithms*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1996.

[12] R. Olfati-Saber and J. S. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *Proc. of the 44th IEEE Conference on Decision and Control*, 2005.

[13] D. P. Spanos, R. Olfati-Saber, and R. M. Murray, "Dynamic consensus on mobile networks," in *Proc. of the 16th IFAC World Congress*, Prague, Czech, Apr. 2005.

[14] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delay," *IEEE Trans. Automatic Control*, vol. 49, no. 9, pp. 101–115, 2004.

[15] C. Godsil and G. Royle, *Algebraic graph theory*. Springer-Verlag, 2001.