Progress In Electromagnetics Research C, Vol. 11, 171–182, 2009

DISTRIBUTED PARTICLE FILTER FOR TARGET TRACKING IN SENSOR NETWORKS

H. Q. Liu, H. C. So, F. K. W. Chan, and K. W. K. Lui

Department of Electronic Engineering
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong, China

Abstract—In this paper, we present a distributed particle filter (DPF) for target tracking in a sensor network. The proposed DPF consists of two major steps. First, particle compression based on support vector machine is performed to reduce the cost of transmission among sensors. Second, each sensor fuses the compressed information from its neighboring nodes with use of consensus or gossip algorithm to estimate the target track. Computer simulations are included to verify the effectiveness of the proposed approach.

1. INTRODUCTION

The research topic of sensor networks has attracted much attention over the past few years because it has wide applications in environmental, medical, food-safety and habitat monitoring, assessing the health of machines, vehicles and civil engineering structures, energy management, inventory control, home and building automation, homeland security and military initiatives [1, 2]. Distributed computation has found very successful applications in sensor networks [2–5] particularly when a powerful central unit is not available. In this paper, we address the problem of target tracking in sensor networks using a distributed processing scheme. The advantages of the distributed approach over the centralized one are twofold: First, it does not need a central unit, which makes it robust. By robust we mean that the whole system still goes on smoothly even when a single sensor fails or joins/leaves the sensor networks. On the other hand, the system cannot be carried on if the central unit fails in the centralized case. Second, it is scalable. By scalable we

Corresponding author: H. Q. Liu (hqliuster@gmail.com).

mean that each sensor only needs its neighboring node information for target track computation. Particle filter (PF) is a standard technique for target tracking [6, 7]. The distributed particle filter (DPF) is a distributed realization of PF. In [5], a DPF is presented to factorize the likelihood function, and each partial likelihood function is updated at individual sensors using only local observations. The partial likelihood function is represented by a parameterized model and the parameters are transmitted to neighboring sensors. In [8], a low dimensional Gaussian mixture model (GMM) is suggested to describe the posterior probability density function (PDF) and model parameters are transmitted over the network. The GMM is also adopted in [9] where a distributed expectation maximization procedure is used to estimate the parameters. In this work, we propose to use support vector machine (SVM) [10, 11] in density estimation to compress the particles because it can always find the global optimum and gives the sparse solution. SVM has already found successful applications in pattern recognition and regression estimation [12, 13]. Since each sensor independently processes its own particles, we need to fuse the sensors' information. To achieve distributed information fusion, a consensus algorithm [4] and a gossip method [14, 15] are adopted. The consensus algorithm is synchronous, that is, all sensors activate and communicate with their neighbors at each iteration. On the other hand, gossip method is asynchronous, in which only one sensor activates to transmit the information to its neighbors at each iteration. The main contribution of this paper is that we use SVM to find the sparse representation of particles and use averaging algorithms to fuse sensor information in the DPF development. For a comparative study between consensus and gossip algorithms, the interested reader is referred to [3].

The rest of the paper is organized as follows. The problem formulation of target tracking is presented in Section 2. In Section 3, we present a SVM-based density estimation for sparse representation of particles. The distributed evaluation is described with the use of consensus and gossip algorithms in Section 4. In Section 5, simulation results for evaluating the tracking performance of the proposed approach are provided. Finally, conclusions are drawn in Section 6.

2. PROBLEM FORMULATION

The model-based methods for tracking applications generally require two models, namely, state model and measurement model. The former describes the evolution of the state with time while the latter defines

the relationship between the noisy observations and state. In case of two-dimensional (2D) target tracking, let $\mathbf{x}_{t,j} = [x_{t,j},\ y_{t,j}\ , \dot{x}_{t,j},\ \dot{y}_{t,j}]^T$ be the state vector that contains the coordinates and velocities of a moving target at time t of the jth sensor. In this study, we assume a linear state [2]:

$$\mathbf{x}_{t,j} = f(\mathbf{x}_{t-1,j}) + \mathbf{v}_{t,j} = \mathbf{F}\mathbf{x}_{t-1,j} + \mathbf{v}_{t,j} \tag{1}$$

where

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The T_s is the sampling interval, $\mathbf{v}_{t,j}$ is a 4×1 independent and identically distributed process noise vector with $\mathbf{v}_{t,j} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$ where $\mathbf{0}$ is a zero vector and $\mathbf{Q}$ is the covariance matrix of the form of $\mathbf{D}\text{diag}(\sigma_x^2, \sigma_y^2)\mathbf{D}^T$, with σ_x^2 and σ_y^2 are the variances in x-coordinate and y-coordinate, and $\mathbf{D}$ is given as

$$\mathbf{D} = \begin{bmatrix} T_s^2/2 & 0 \\ 0 & T_s^2/2 \\ T_s & 0 \\ 0 & T_s \end{bmatrix}$$

On the other hand, we consider that time-of-arrival (TOA) and angle-of-arrival (AOA) measurements are obtained at the sensors. By multiplying the TOAs with known propagation speed, range information is acquired. The observed distance measurement $r_{t,j}$ and AOA measurement $\phi_{t,j}$ at time t of the jth sensor are:

$$r_{t,j} = d_{t,j} + \varepsilon_{t,j},\ t = 1, 2, \ldots,\ j = 1, 2, \ldots, M \tag{2}$$

$$\phi_{t,j} = \theta_{t,j} + q_{t,j},\ t = 1, 2, \ldots,\ j = 1, 2, \ldots, M \tag{3}$$

where $d_{t,j} = \sqrt{(x_t - x_j)^2 + (y_t - y_j)^2}$, $\theta_{t,j} = \tan^{-1}(\frac{y_t - y_j}{x_t - x_j})$, M is the total number of sensors in the network, (x_j, y_j) denotes the known coordinates of the jth sensor, $\varepsilon_{t,j} \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ is noise for range measurement and $q_{t,j} \sim \mathcal{N}(0, \sigma_q^2)$ is noise for the AOA measurement. Putting the TOA and AOA measurements of the jth sensor together yields the vector for measurement model, denoted by $\mathbf{z}_{t,j}$:

$$\mathbf{z}_{t,j} = g(\mathbf{x}_{t,j}) + \mathbf{w}_{t,j} \tag{4}$$

where $\mathbf{z}_{t,j} = [r_{t,j}\ \ \theta_{t,j}]^T$ and $\mathbf{w}_{t,j} = [\varepsilon_{t,j}\ \ q_{t,j}]^T$. For such a nonlinear target tracking problem, PF has been successfully applied [6, 7, 16–19]. The detailed information of PF is summarized in Table 1. The goal of this work is to develop a DPF instead of using the centralized

Table 1. Particle filter algorithm.

For $t = 1, 2, \ldots$
(i) **Initialization**:
For $i = 1, \ldots, N$, sample the state particle $\mathbf{x}_0^i \sim p(\mathbf{x}_0)$
(ii) **Prediction of particles**:
For $i = 1, \ldots, N$, draw particles
$\mathbf{x}_t^{(i)} \sim q\left(\mathbf{x}_t | \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_{1:t}\right)$
where $q(\cdot)$ is an importance function,
$\mathbf{z}_{1:t} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t\}$ denotes all the observations up to the current time t and $\mathbf{z}_t = [z_{t,1}, \ldots, z_{t,M}]^T$.
(iii) **Update**:
For $i = 1, \ldots, N$, evaluate the importance weight:

$$w_t^{(i)} \propto w_{t-1}^{(i)} \times \frac{p\left(\mathbf{z}_t \middle| \mathbf{x}_t^{(i)}\right) p\left(\mathbf{x}_t^{(i)} \middle| \mathbf{x}_{t-1}^{(i)}\right)}{q\left(\mathbf{x}_t \middle| \mathbf{x}_{t-1}^{(i)}, \mathbf{z}_{1:t}\right)}$$

For $i = 1, \ldots, N$, normalize the importance weight:

$$\tilde{w}_t^i = w_t^i / \sum_{j=1}^N w_t^j$$

(iv) **Resampling step**: (if necessary)
Eliminate samples with low importance weights and multiply samples with high importance weights.
For $i = 1, \ldots, N$, set $w_t^i = 1/N$.
(v) **Estimation step**:
The minimum mean square error estimate of state is obtained as:

$$\hat{\mathbf{x}}_t \approx \sum_{i=1}^N w_t^{(i)} \mathbf{x}_t^{(i)}$$

one. To develop a DPF, there are two most important questions need to be handled. The first one is that what information we should communicate between sensors. The second one is that how we fuse the information contained in each sensor. In the following sections, we will answer these two questions respectively.

3. DENSITY ESTIMATION USING SUPPORT VECTOR MACHINE

In this section, we answer the first question. According to the state and measurement models, the particles are sampled to represent the posterior PDF of the target position [7]. Considering the communication energy consumption, it is not feasible to transmit all particles over the sensor network. Given the set of particles $\{\mathbf{x}_i\}_{i=1}^N$ at each sensor, we want to find a sparse representation of those particles. A standard way is to use Gaussian mixtures to approximate the particle distribution but the maximum likelihood estimate does not exist [20]. In this paper, we seek a non-parametric method, namely SVM, to achieve the task. SVM has found a lot of successful applications in regression and classification problems [12, 13]. In what follows, the use of SVM in density estimation will be employed to compress the particles. Note that we only consider one sensor's particles, other sensors do the same thing in parallel. For clarity, the subscripts are omitted as long as it does not introduce any confusions. The objective is to estimate the PDF $p(\mathbf{x})$ that represents the particles from a sensor at each time. The estimate of $p(\mathbf{x})$ can be obtained through the following equation:

$$\int_{-\infty}^{\mathbf{x}} p(\mathbf{s})d\mathbf{s} = F(\mathbf{x}) \tag{5}$$

where $F(\mathbf{x})$ is the cumulative distribution function (CDF). Estimating $p(\mathbf{x})$ requires solving (5) from the unknown function $F(\mathbf{x})$ based on the particles $\{\mathbf{x}_i\}_{i=1}^N$. The multidimensional empirical distribution function $F_l(\mathbf{x})$ to approximate the actual distribution $F(\mathbf{x})$ is

$$F(\mathbf{x}) \approx F_l(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}\prod_{h=1}^{d} u\left(\mathbf{x}^h - \mathbf{x}_i^h\right) \tag{6}$$

where $u(\cdot)$ is the step function, $d = 4$ is the dimension of $\mathbf{x}$. The problem now is to solve (5) based on $\{(\mathbf{x}_1, F_l(\mathbf{x}_1)), \ldots, (\mathbf{x}_N, F_l(\mathbf{x}_N))\}$. It is proved in the SVM theory that the global estimates for the unknown CDF and PDF are:

$$\hat{F}(\mathbf{x}) = \sum_{i=1}^{N}\beta_i K(\mathbf{x}_i, \mathbf{x}), \qquad \hat{p}(\mathbf{x}) = \sum_{i=1}^{N}\beta_i \tilde{K}(\mathbf{x}_i, \mathbf{x}) \tag{7}$$

where $\{\beta_i\}_{i=1}^N$ are weight parameters, $K(\cdot,\cdot)$ is a kernel function, and $\tilde{K}(\cdot,\cdot)$ is a 'cross-kernel' function. The two functions are related by:

$$K(\mathbf{x}, \mathbf{y}) = \int_0^{\mathbf{y}} \tilde{K}(\mathbf{x}, \mathbf{s})d\mathbf{s} \tag{8}$$

In our study, the Gaussian cross-kernel function with variance σ_j^2 at the jth sensor is adopted, that is,

$$\tilde{K}(\mathbf{x},\mathbf{s}) = \frac{1}{\sigma_j(2\pi)^{d/2}} \exp\left(-\frac{\|\mathbf{x}-\mathbf{s}\|^2}{2\sigma_j^2}\right) \tag{9}$$

The weight parameters $\{\beta_i\}_{i=1}^N$ are obtained by solving the following constrained optimization problem [10]

$$\begin{aligned} &\min_{\beta_i,\xi_i}\left\{\sum_{i=1}^N \xi_i^2 + \sum_{i=1}^N \frac{\beta_i}{\sigma_j}\right\} \\ &s.t. \sum_{i=1}^N \beta_i K(\mathbf{x}_i,\mathbf{x}_l) + \xi_i = F_l(\mathbf{x}_l), \quad l = 1,\ldots,N \\ &\sum_{i=1}^N \beta_i = 1 \\ &\beta_i \geq 0 \end{aligned} \tag{10}$$

where $\{\xi_i\}_{i=1}^N$ is a set of slack variables. This optimization solution gives only a small number of non-zero β_i, and the corresponding particles $\{\mathbf{x}_i\}$ are called support vectors. Using the SVM approach, particle compression is achieved. In this way, only support vectors will be communicated between neighbor sensors in the sensor network instead of the whole set of particles.

4. DISTRIBUTED PARTICLE FILTER

In this section, we answer the second question. To achieve DPF processing, we need to fuse the information contained in each sensor in a distributed manner. Now, we describe two classes of fusion algorithms, namely, consensus and gossip algorithms, which will be used for DPF. The goal is to compute the average value of the support vector extracted from each sensor data. In what follows, we will introduce these two algorithms in details.

• Consensus algorithm [4, 21]:

The consensus filter is a synchronous method. All sensor nodes wake up at each iteration k, communicate with their neighbors and update their values. Suppose that the estimated density function in the jth sensor at time t is $\hat{p}_j^t(\mathbf{x})$, then its update equation at iteration $(k+1)$ is

$$\hat{p}_j^t(\mathbf{x})(k+1) = \hat{p}_j^t(\mathbf{x})(k) + \eta\left\{\sum_{i\in\mathcal{N}_j}\left(\hat{p}_i^t(\mathbf{x})(k) - \hat{p}_j^t(\mathbf{x})(k)\right)\right\} \tag{11}$$

where η is the iteration step, $\mathcal{N}_j$ is a set of neighbors of the jth sensor. After long enough iterations, each sensor will have the same value, denoted by $\bar{p}(\mathbf{x})$:

$$\bar{p}^t(\mathbf{x}) = \frac{1}{M}\sum_{j=1}^{M}\hat{p}_j^t(\mathbf{x}) = \frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{N}\beta_i^j\tilde{K}\left(\mathbf{x}_i^j, \mathbf{x}\right) \tag{12}$$

where β_i^j and $\mathbf{x}_i^j$ denote weight parameters and particles at the jth sensor respectively. For the detailed study of the consensus algorithm, the interested reader is referred to [21].

• Gossip method [15]:

The broadcast gossip method is an asynchronous algorithm. At each iteration step, only one sensor activates. Suppose that the jth sensor activates at iteration step k, broadcasts its information to its neighbors, and neighbors update their values. The update equation is:

$$\hat{p}_i^t(\mathbf{x})(k+1) = \gamma\hat{p}_i^t(\mathbf{x})(k) + (1-\gamma)\hat{p}_j^t(\mathbf{x})(k), \quad \forall i \in \mathcal{N}_j \tag{13}$$

The broadcast gossip algorithm converges to average of the all sensor values in the mean sense [15]. However, the convergence of the consensus-DPF is superior to that of the gossip-DPF in terms of number of iterations, which is demonstrated in our simulation studies. With the use of distributed average algorithms, the DPF algorithm at each sensor is summarized in Table 2. Note that the global function at time t should be calculated before going to time $t+1$. Nevertheless, it is computed in an iterative manner, that is, we take the converged global function for a fixed number of iterations as the finalized function.

5. SIMULATION RESULTS

Computer simulations have been conducted to evaluate the tracking performance of the proposed methods. We consider that $M = 110$ sensors are randomly deployed on a 2D sensor network of dimension 500 m $\times$ 500 m and the connection between sensors is shown in Figure 1 [9]. The initial state vector of the target is $[50\,\text{m}, 40\,\text{m}, 7\,\text{m/s}, 6\,\text{m/s}]^T$. The target trajectory is generated without process noise. The sampling time is $T_s = 1$ s. The number of particles is $N = 10$ due to the limited storage in sensors. In the consensus algorithm, $\eta = 0.01$ and $\gamma = 0.5$ is assigned in the gossip method. The iteration number for the consensus and gossip algorithms in (11) and (13) is 100. In the kernel function, the variance is set to be $\sigma_j^2 = 1$ for all sensors. The DPF are randomly initialized around the true values. All results provided are averages of 50 independent runs. In the centralized PF, the particle number is 1100. In Figures 2

and 3, the mean square position error (MSPE) performance for DPF with consensus and gossip schemes is plotted for all sensors at $\sigma_\varepsilon^2 = \sigma_q^2 = 1$ and $\sigma_\varepsilon^2 = \sigma_q^2 = 5$, respectively. The DPFs have comparable performance with the centralized PF, which proves the validity of the proposed DPF. Intuitively, the centralized PF should have the best performance when compared with the DPFs. However, the results show that the latter can outperform the former. One possible reason for the superior performance of the latter is that distributed computation introduces diversity between sensors, which makes DPF better in our case. Here, diversity refers to different information from different sensors. Another possible reason is that the number of particles in the centralized method is not sufficiently large. On the other hand, it is

Table 2. Distributed particle filter algorithm.

For $t = 1, 2, \ldots$
(i) Initialization: For $i = 1, \ldots, N$, sample state particle $\mathbf{x}_0^i \sim p^t(\mathbf{x}_0)$
(ii) Calculate local function $\hat{p}_j^t(\mathbf{x})$ based on SVM at each sensor.
(iii) Compute global function $\bar{p}^t(\mathbf{x})$ based on consensus filter or broadcast gossip algorithm using the local functions for a fixed number of iterations.
(iv) Sample particles from estimated $\bar{p}^t(\mathbf{x})$.
(v) Use bootstrap method to calculate predicted particles, predicted observations and importance weights.
(vi) Resample if necessary.

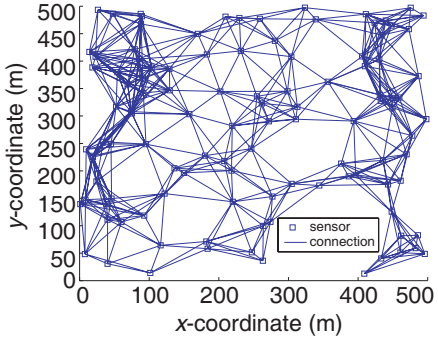

Figure 1. Sensor network geometry and connection.

observed that the centralized algorithm is more robust to measurement noises. It is observed that the consensus-DPF is superior to gossip-DPF because the convergence of the consensus filter is faster that of the gossip algorithm. The MSPE of DPFs is plotted in Figure 4 without SVM compression scheme, in which it is observed that they have similar performance as in Figure 2. On the other hand, the number of particles transmitted using SVM scheme at one time is shown in Figure 5. It can be seen that there are three cases, namely,

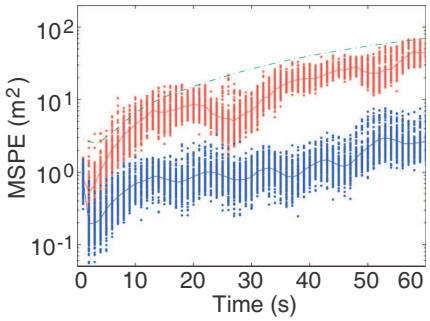

Figure 2. Mean square position error (MSPE) of different methods at $\sigma_\varepsilon^2 = \sigma_q^2 = 1$ with SVM. The upper part refers to centralized PF, middle part refers to gossip-DPF, and lower part corresponds to consensus-DPF.

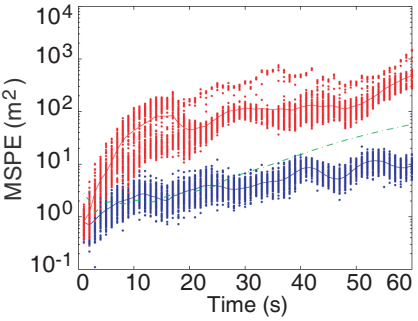

Figure 3. Mean square position error (MSPE) of different methods at $\sigma_\varepsilon^2 = \sigma_q^2 = 5$ with SVM. The upper part refers to gossip-DPF, middle part refers to centralized PF, and lower part corresponds to consensus-DPF.

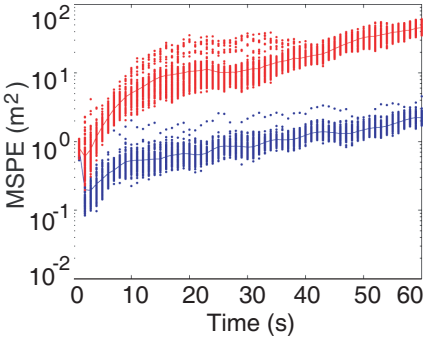

Figure 4. Mean square position error (MSPE) of different methods at $\sigma_\varepsilon^2 = \sigma_q^2 = 1$ without SVM scheme. The upper part refers to gossip-DPF and lower part corresponds to consensus-DPF.

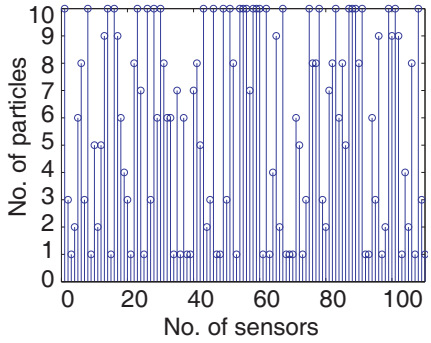

Figure 5. Number of particles transmitted using SVM scheme.

only one particle is being transmitted in some sensors, all particles are exchanged in some sensors, and portion of the particles is transmitted in other sensors. However, on average less than 60% percentage of particles is transmitted compared to transmitting all particles, that is, without SVM scheme, which demonstrates the effectiveness of the SVM scheme in compressing particles.

6. CONCLUSION

A distributed particle filter (DPF) for target tracking in sensor networks is developed in this paper. The proposed DPF includes two major steps, namely, the use of support vector machine to compress the particles and utilizing distributed averaging algorithms to fuse the information between sensors. This DPF is robust because it does not require a central unit and it is scalable because the information exchange only takes place between neighboring sensors. Computer simulation results demonstrate the validity of the proposed methods. It is expected that the performance can be further improved if extended Kalman filter or unscented Kalman filter is used as importance sampling function at the expense of higher computational requirement [16, 17].

ACKNOWLEDGMENT

The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [Project No. CityU 122307].

REFERENCES

1. Patwari, N., J. N. Ash, S. Kyperountas, A. O. Hero III, R. L. Moses, and N. S. Correal, "Locating the nodes — Cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, Vol. 22, No. 4, 54–69, Jul. 2005.
2. Guo, D. and X. Wang, "Dynamic sensor collaboration via sequential Monte Carlo," *IEEE Journal on Selected Areas in Communications*, Vol. 22, No. 6, 1037–1047, Aug. 2004.
3. Denantes, P., F. Benezit, P. Thiran, and M. Vetterli, "Which distributed averaging algorithm should I choose for my sensor network?," *Proc. 27th IEEE Conf. Computer Communications and Networks*, 986–994, St. Thomas, U.S. Virgin Islands, Apr. 2008.
4. Olfati-Saber, R., "Distributed Kalman filter with embedded consensus filters," *Proc. 44th IEEE Conf. Decision and Control and the European Control Conference*, 8179–8184, Seville, Spain, Dec. 2005.
5. Coates, M. J., "Distributed particle filtering for sensor networks," *Proc. Int. Symp. Information Processing in Sensor Networks*, 99–107, Berkeley, CA, Apr. 2004.
6. Ristic, B., A. Arulampalam, and N. Gordon, *Beyond the Kalman Filter-particle Filters for Tracking Applications*, Artech House, Boston, 2004
7. Arulampalam, M. S., S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. on Signal Processing*, Vol. 50, No. 2, 174–188, Feb. 2002.
8. Sheng, Y., X. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor networks," *Proc. 4th Int. Symposium on Information Processing in Sensor Networks*, 181–188, Los Angeles, California, USA, Apr. 2005.
9. Gu, D., "Distributed particle filter for target tracking," *Proc. IEEE International Conference on Robotics and Automation*, 3856–3861, Roma, Italy, Apr. 2007.
10. Weston, J., A. Gammerman, M. Stitson, V. Vapnik, V. Vovk, and C. Watkins, "Support vector method for multivariate density estimation," *Advances in Kernel Methods: Support Vector Machines*, 293–306, MIT Press, Cambridge, MA, 1998.
11. Vapnik, V. N. and S. Mukherjee, "Support vector method for multivariate density estimation," Tech. Rep., No. 1653, A.I.

Memo, MIT AI Lab., 1999.

12. Burges, C. J. C., "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, Vol. 2, 121–167, Jun. 1998.
13. Smola, A. J. and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, Vol. 14, No. 3, 199–222, Aug. 2004.
14. Boyd, S., A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. on Information Theory*, Vol. 52, No. 6, 2508–2530, Jun. 2006.
15. Aysal, T. C., M. E. Yildiz, and A. Scaglione, "Broadcast gossip algorithms," *Proc. IEEE Information Theory Workshop*, 343–347, Porto, Portugal, May 2008.
16. Liu, H. Q., H. C. So, K. W. K. Lui, and F. K. W. Chan, "Sensor selection for target tracking in sensor networks," *Progress In Electromagnetics Research*, PIER 95, 267–282, 2009.
17. Liu, H. Q. and H. C. So, "Target tracking with line-of-sight identification in sensor networks under unknown measurement noises," *Progress In Electromagnetics Research*, PIER 97, 373–389, 2009.
18. Chen, J. F., Z. G. Shi, S. H. Hong, and K. S. Chen, "Grey prediction based particle filter for maneuvering target tracking," *Progress In Electromagnetics Research*, PIER 93, 237–254, 2009.
19. Khodier, M. M. and M. Al-Aqeel, "Linear and circular array optimization: A study using particle swarm intelligence," *Progress In Electromagnetics Research B*, Vol. 15, 347–373, 2009.
20. Vapnik, V. N., *Statistical Learning Theory*, John Wiley, 1998
21. Olfati-Saber, R. and J. S. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," *Proc. 44th IEEE Conference on Decision and Control, and the European Control Conference*, 6698–6703, Seville, Spain, Dec. 2005.