Cluster Based Sensor Scheduling in a Target Tracking Application with Particle Filtering

Özgür Özfidan
R&D Department
PAVO Electronic
Gebze, Kocaeli, Turkey
ozgur.ozfidan@pavo.com.tr

Uluğ Bayazıt
Electronics Engineering
Işık University
Istanbul, Turkey
bayazit@isikun.edu.tr

Hakan A. Çırpan
Electrical-Electronics Engineering
Istanbul University
Istanbul, Turkey
hcirpan@istanbul.edu.tr

***Abstract*— In multi-sensor applications management of sensors is necessary for the classification of data they produce and for the efficient use of sensors as well. One of the important aspects in sensor management is the sensor scheduling. By scheduling the sensors, serious reductions can be achieved in the cost of bandwidth, power, and computation. In this work a simple solution for the problem of sensor scheduling in a multi-sensor target tracking application is presented. Due to non-linearity of the problem itself, proposed solution is presented in the framework of non-linear Bayesian estimation.**

I. Introduction

In the last decades there has been a considerable growth in the semiconductor technology. With the advances in this field, the size of measuring devices are now smaller and smaller and their prices are quite low when compared to the past. Consequently, these devices can now be deployed in various places where their small form factor is a major advantage. This give rise to a big interest to the sensor networks and technology. Sensor networks have found application areas both in military and civil environments. In multi-sensor networks, multiple measurement devices, namely sensors, are employed in an area of interest to collect data, and share this data with a data fusion center directly, or by forwarding through their neighbors. Thus, we need to effectively control the sensors and interpret the information they send us. Effective utilization of sensors brings some other issues into our consideration: Operational costs and sensor lifetime. Operational costs include cost of bandwidth, power and computation while sensor lifetime is an issue directly related to the power consumption of the device. A powerful method for the effective utilization of sensors is to schedule them in a smart way. Sensor scheduling is performed to save the resources and improve the overall system performance.

Bayesian techniques have lately been extensively used in Target tracking applications[1]. Due to non-linearity and possibly non-gaussianity of the problem, Sequential Monte Carlo (SMC) Methods [5], [6], are widely used for target tracking applications in Multi-Sensor systems. In a multi sensor environment, different approaches in sensor management such as information driven [2] and Cluster based techniques[3], [4], have previously been proposed.

In this paper, we present a novel sensor scheduling method based on partitioning the sensors into clusters. Our clustering algorithm depends on associating each slave sensor to its nearest master node. Initially, we assumed a totally random deployment strategy into the region of interest. After grouping the sensors, we used the information collected by the sensors in the same cluster to track the position of our target.Block diagram in Fig. 1 describes such a tracking scenario.

II. Description of Tracking Scenario

In this section, we introduce a general model for our tracking scenario. We consider the task of tracking a moving vehicle through our two dimensional stationary sensor field

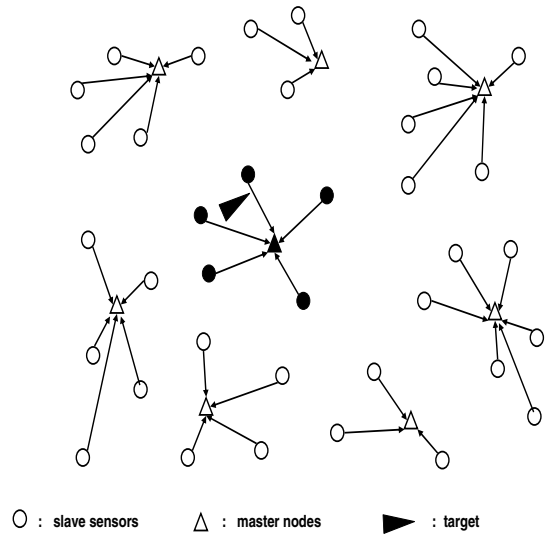

Fig. 1. *General Tracking Scenario: Target moves in the region of interest. Black triangle and circles represent the sensors in the active cluster. Arrows indicate the master node that each slave report.*

This research has been funded by the The Scientific & Technological Research Council of Turkey (TÜBİTAK), Project No: 104E130 and also supported in part by the Research Fund of the University of Istanbul. Project number: 513/05052006.

under surveillance while conserving power by minimizing the number of active sensors. Before we run our tracking algorithm there is a set-up procedure which works as follows : First, we randomly distribute both the slave sensors and the master nodes into our region of interest. Master nodes are basically responsible for communicating with the data fusion center. Remaining sensors will be called slaves. Slave sensors report the position of the target to their master periodically or if there is no target detected, they report this situation as well . After randomly distributing both type of sensors, we associate each slave with a master by running our master-slave association algorithm. Basic criteria for this process is the Cartesian distance between the master nodes and the slave sensors. Each slave is associated with its closest master. Another practical real world constraint that we take into account at this point is the service capacity of a master node. Maximum number of slaves that we can associate with each master is defined. During the set-up process if this limit is exceeded for a master, than the remaining slaves are associated with another master. If we summarize our assumptions:

- Target moves with a constant velocity
- Initial position of the target is known
- Target state to be tracked consists of its two dimensional position and velocity
- There are randomly distributed M stationary master nodes
- There are randomly distributed S stationary slave sensors
- Each slave sensor is associated with a master
- Maximum number of slaves that can be associated with a master is C
- At each time step there is only one active master communicating with the sink
- All masters can communicate with each other
- Slaves can communicate with their associated master node only

The driving force behind this scenario is the limiting conditions of the physical world. Above scenario, for instance, can be exactly achieved by throwing the sensors away from a plane. Since the average communication time and the power consumption of the master nodes will be much more than the slaves, they can be equipped with longer life batteries and higher output transmit power RF communication ICs. That means we can differentiate master and slave sensors before the setup process and distribute both type of sensors in a completely random fashion. Number of master nodes and the number of slaves that each master can give service is a matter of optimization.

Another advantage of this scenario is the efficient use of bandwidth when compared with the case each sensor sends the target position individually to the sink. By limiting the output transmission power of the slaves, possible interference problem between the neighboring clusters can be overcome.

As mentioned before, our main objective is to accurately track the target while minimizing the number of active sensors. Only the active sensors provide observation about target position, otherwise they are configured to remain in sleep mode to reduce the power consumption. Thus, activation of sensors within a specified distance from the current target position estimate is quite important. Several different formulations of this problem are possible as target of interest moves through our randomly distributed sensors. Our approach at this point is simply to compare the current position estimate of the target with the position of each master node at every time step and to activate the associated slaves of the closest master for the next epoch. Here, we are using the assumption that master nodes are always active and thus leadership can be immediately transferred from one master to another. Every master can activate its own slaves whenever needed.

III. Tracking and Scheduling

In this section, we introduce a general model for our multi-sensor, single target system. Target velocity is assumed to be constant during the tracking phase. Sensors are assumed to be range only sensors. After running our clustering algorithm we employ particle filter to estimate the position of our target based on reported range information by the slave sensors in the same cluster.

A. System Dynamics

Now we define the system and observation models for our target in a detailed manner. For the 2-dimensional case, state vector $\mathbf{X}_k$ at time step k contains four elements: positions in the x and y directions and velocities in the x and y directions:

$$\mathbf{X}_k = [x_k, y_k, \dot{x}_k, \dot{y}_k]^T \tag{1}$$

Kinematics for the target can be written as

$$x_k = x_{k-1} + \dot{x}_{k-1}\Delta t + \frac{1}{2}\ddot{x}_{k-1}\Delta t^2 + \frac{1}{3}\dddot{x}_{k-1}\Delta t^3 \tag{2}$$

$$y_k = y_{k-1} + \dot{y}_{k-1}\Delta t + \frac{1}{2}\ddot{y}_{k-1}\Delta t^2 + \frac{1}{3}\dddot{y}_{k-1}\Delta t^3 \tag{3}$$

where Δt is the time difference between state transitions or simply the sampling period. The parameters $\ddot{x}_k$ and $\ddot{y}_k$ represent the acceleration in the x and y directions, respectively. Finally, $\dddot{x}_k$ and $\dddot{y}_k$ are to represent the variations in the acceleration in two directions again. We model the acceleration components using random noise. Assuming target moves with a constant velocity, using (2) and (3), the state equation can be written as

$$\mathbf{X}_k = F\mathbf{X}_{k-1} + Q^{1/2}\mathbf{V}_{k-1} \tag{4}$$

where

$$F = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5}$$

$$Q = q \begin{pmatrix} \Delta t^3/3 & 0 & \Delta t^2/2 & 0 \\ 0 & \Delta t^3/3 & 0 & \Delta t^2/2 \\ \Delta t^2/2 & 0 & \Delta t & 0 \\ 0 & \Delta t^2/2 & 0 & \Delta t \end{pmatrix} \tag{6}$$

where Q is the state error covariance matrix and models the acceleration terms in the x and y directions. The vector $\mathbf{V}_k$ is a Gaussian random vector of zero mean, unit variance and independent components. Finally, q is used to control the intensity of the process noise.

The observation vector can simply be related to the state vector as

$$\mathbf{Z}_k = \mathbf{d}_k + R^{1/2}\mathbf{n}_k. \tag{7}$$

$\mathbf{d}_k$ is an L x 1 vector whose elements are the Cartesian distance between the target and the position of each slave sensor used to generate observations at time step k, where L is the number of slaves in the corresponding cluster. The ith element of $\mathbf{d}_k$ is

$$d_{ik} = ||\boldsymbol{x}_k - s_i||, i = 1, ...L. \tag{8}$$

where $\boldsymbol{x}_k$ is the target position and s_i is the position of ith sensor in the corresponding cluster. R denotes the measurement error covariance matrix and $\mathbf{n}_k$ is an L x 1 vector whose elements are generated by a Gaussian random variable of zero mean and unit variance.

At this point, detection quality of our range only Infrared (IR) sensors need to be examined carefully. Intuitively, it can be said that Signal to Noise Ratio (SNR) will decrease with increasing slave-target distance. Defining Received and Transmitted Signal Powers as P_r and P_t respectively, we can write

$$P_r = KP_t. \tag{9}$$

K represents both the attenuation due to the channel characteristics and the reflection depending on the target material. Then, SNR can simply be expressed as

$$SNR = P_r/\sigma^2. \tag{10}$$

We emphasize that ambient light and the environmental thermal fluctuations are the major noise sources together with the shot noise for the IR sensors in our application. This situation can be modeled in the observation vector above, by increasing the variance of measurement error with sensor-target distance. Furthermore, if we assume that the noise components for each sensor are independent, R becomes an LxL diagonal matrix whose elements are directly proportional to the slave-target distance, where L is the number of slaves in the active cluster.

$$R_{ii} \propto ||\boldsymbol{x}_k - s_i|| \tag{11}$$

B. Clustering

For the maneuvering target tracking application we can assume our region of interest under surveillance is quite a large area and only a small portion of deployed sensors can provide useful information at a specific time instance. That is why, clustering the range only sensors is one of the most reasonable strategy that can be applied. By clustering, we can reduce the spatial coverage of sensors considerably, which means higher quality of data reported by each sensor.

Now we present unbalanced clustering algorithm for the efficient data collection. As mentioned before, initially both Master nodes and Slave sensors are distributed into the region of interest randomly.

$S \equiv$ **Number of slave sensors**
$M \equiv$ **Number of master nodes**
$C \equiv$ **Max service capacity of a master**
$m \equiv$ **Master nodes**
$s \equiv$ **Slave sensors**

Calculate distance from all slaves to all masters
for $i = 1, 2, \cdots, M$
  for $j = 1, 2, \cdots, S$
    $D[i, j] = ||m_i - s_j||$
  end
end

Assign each slave to its closest master
for $j = 1, 2, \cdots, S$
  $[mindist, i] = min(D[:, j])$
  if capacity of ith master is smaller than C
    Assign slave j to master i
  else
    Assign slave j to another master
  end
end

TABLE I
MASTER-SLAVE ASSOCIATION ALGORITHM

C. Particle Filtering

Since we know the initial position of our target, we start filtering with the data reported by the slaves in the cluster whose master is closest to this initial position. Given $p(\mathbf{X}_k|\mathbf{X}_{k-1})$ and $p(\mathbf{Z}_k|\mathbf{X}_k)$ Generic Particle Filter and Resampling algorithms [1] are used to recursively estimate the state of moving target.

for $i = 1, 2, \cdots, N$
  -Draw $\boldsymbol{x}_k^i \sim q(\boldsymbol{x}_k|\boldsymbol{x}_{k-1}^i, \mathbf{z}_k)$
  -Assign the particle a weight ω_k^i
end
Calculate total weight: $t = sum_{i=1}^{N}\omega_k^i$
for $i = 1, 2, \cdots, N$
  -Normalize: $\omega_k^i = t^{-1}\omega_k^i$
end
Calculate $\widehat{N_{eff}}$ using (14)
if $\widehat{N_{eff}} < N_T$
  -Resample the particles
end

TABLE II
GENERIC PARTICLE FILTER ALGORITHM

We approximate the posterior density $p(\mathbf{X}_k|\mathbf{Z}_{1:k})$ at time step k by a set of particles $\{\boldsymbol{x}_k^i, i = 1, 2, ...N\}$ and associated weights $\{\omega_k^i, i = 1, 2, ...N\}$ where $\sum_{i=1}^{N}\omega_k^i = 1$. We use the prior distribution as the importance density thus, draw the particles from a proposal distribution $q(\boldsymbol{x}_k^i|\boldsymbol{x}_{k-1}^i) = p(\boldsymbol{x}_k^i|\boldsymbol{x}_{k-1}^i)$ and assign each particle a weight using the weight update equation $\omega_k^i \propto \omega_{k-1}^i p(\mathbf{z}_k|\boldsymbol{x}_k^i)$. Approximation to the posterior density is then

$$p(\boldsymbol{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N}\omega_k^i\delta(\boldsymbol{x}_k - \boldsymbol{x}_k^i) \tag{12}$$

Initialize the CDF: $c_1 = 0$
for $i = 2, \cdots, N$
-Construct CDF: $c_i = c_{i-1} + \omega_k^i$
end
Start at the bottom of the CDF: $i = 1$
Draw a starting point: $u_1 \sim U[0, N^{-1}]$
for $j = 1, 2, \cdots, N$
-Move along the CDF: $u_j = u_1 + N^{-1}(j-1)$
-while $u_j > c_i$
$*i = i + 1$
-end
-Assign sample: $\boldsymbol{x}_k^{j*} = \boldsymbol{x}_k^i$
-Assign weight: $\omega_k^j = N^{-1}$
-Assign parent: $i^j = i$
end

TABLE III
RESAMPLING ALGORITHM

And finally the state estimation is

$$\hat{\boldsymbol{x}}_k \approx \sum_{i=1}^{N} \omega_k^i \boldsymbol{x}_k^i. \tag{13}$$

Since variance of weights increase continuously, resampling is necessary in order to avoid degeneracy. Measure of degeneracy can be approximated by

$$\widehat{N_{eff}} = \frac{1}{\sum_{i=1}^{N} (\omega_k^i)^2}. \tag{14}$$

(13) can be thought of as an MMSE Estimator which is optimum for gaussian densities. The estimation error covariance matrix is obtained by

$$\hat{\mathbf{P}}_k = \sum_{i=1}^{N} \omega_k^i (\boldsymbol{x}_k^i - \hat{\boldsymbol{x}}_k)(\boldsymbol{x}_k^i - \hat{\boldsymbol{x}}_k)^T \tag{15}$$

D. Sensor Scheduling

In this section, we develop a sensor scheduling method for our randomly distributed sensors. Note that sensors were initially partitioned into clusters by running Master-Slave association algorithm given in Table 1. In our work we apply the Closest Point Activate (CPA) policy by comparing the Cartesian distance between the Master of each cluster and the estimated target position at each time step k and activate the corresponding cluster for the next time epoch.

Cost function is

$$C_k^i = \|\hat{\boldsymbol{x}}_k - m_i\| \tag{16}$$

where $\hat{\boldsymbol{x}}_k$ is the estimated target position at time step k. Finally our scheduling decision is that we choose the master node for which the cost function is minimized

$$m_{opt} = argmin_i C_k^i \tag{17}$$

Then, the corresponding master node takes the leadership and activates its associated slaves immediately. For the next time step, we use the observation vector obtained by this new sensor set.

IV. SIMULATIONS AND RESULTS

In this section, we discuss an example of target tracking using our proposed sensor scheduling algorithm. For the same scheduling algorithm, we also compared the results of proposed non-linear Bayesian estimation method described in section 3.3, Particle Filter (PF), with the method of Unscented Kalman Filter (UKF)[7]. For the simulations, the trajectory for a target was generated in a 2-dimensional cartesian coordinate system. Initially, 64 master nodes and 256 slave sensors were distributed randomly in the area x = (-8000,8000) and y = (-8000,8000). Then, our Master-Slave Association algorithm was applied. Maximum number of slaves that a master can give service was assumed to be 5. Sampling period, Δt was

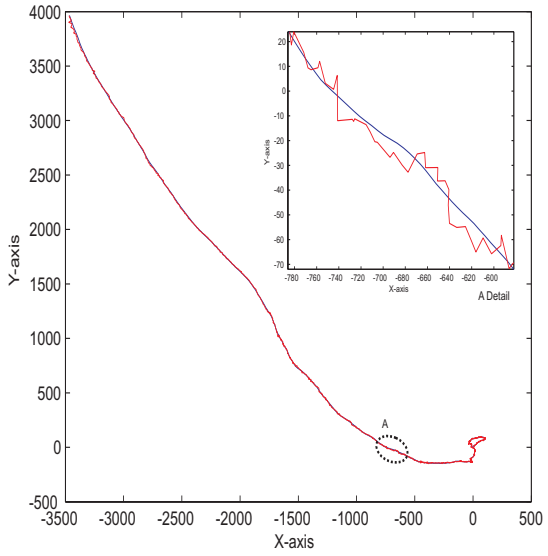

Fig. 2. *True and Estimated trajectories with PF. Blue and red lines represent the true and estimated trajectories*

chosen to be 2 seconds, which generates the state covariance matrix in (6) as.

$$Q = \begin{pmatrix} 2.67 & 0 & 2 & 0 \\ 0 & 2.67 & 0 & 2 \\ 2 & 0 & 2 & 0 \\ 0 & 2 & 0 & 2 \end{pmatrix} \tag{18}$$

The process noise intensity factor q was taken as 0.01 and the initial position of target was taken to be $(x, y) = (0, 0)$. For the particle filter algorithm we used a total of 200 particles and 1000 time steps. Resampling applied when $\widehat{N_{eff}}$ is below the threshold 40. UKF parameters α and β were set to be 0.05 and 2, respectively.

During the tracking phase, at each time step k, we have updated measurement error covariance matrix R by calculating the distance between each slave and the target position.

$$R = R_{coeff} R' \tag{19}$$

$$R' = \frac{1}{d_{max}} diag([d_1, d_2, ...d_L]^T) \tag{20}$$

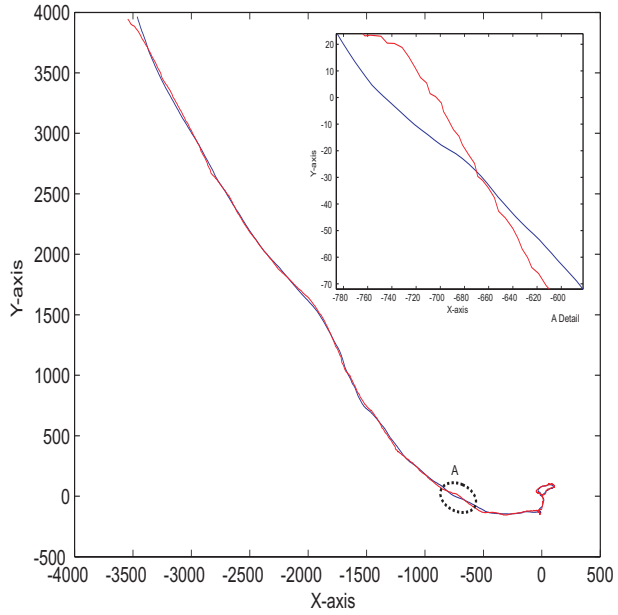

Fig. 3. *True and Estimated trajectories with UKF. Blue and red lines represent the true and estimated trajectories*

where

$$d_i = \|\hat{\boldsymbol{x}}_k - s_i\|, i = 1, 2, ...L \tag{21}$$

d_{max} is the normalizing constant and L is the number of slaves in the active cluster. Constant R_{coeff} was set to 100.

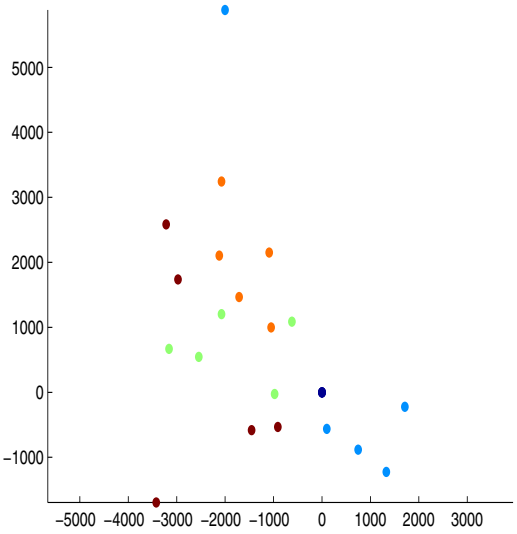

Fig. 4. *Activated sensors.*

True and Estimated target trajectories using Particle Filter and Unscented Kalman Filter methods are shown in Figures 2 and 3. Activated sensors throughout the tracking phase with two different Bayesian methods are same and shown in Figure 4. Rms position errors in x and y directions for two methods are shown in Figures 5 and 6.

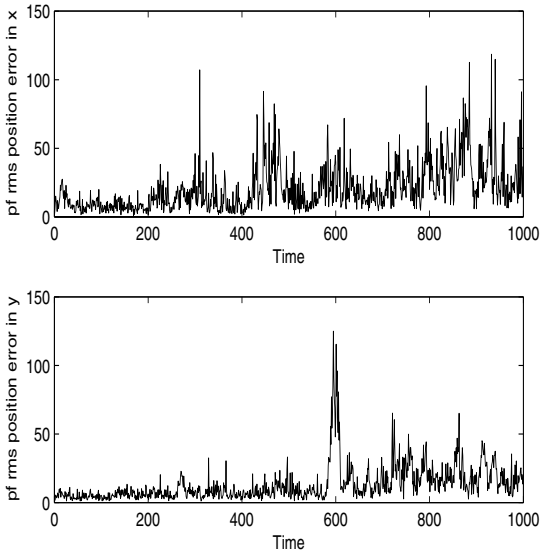

Fig. 5. *rms position errors in x and y directions for PF.*

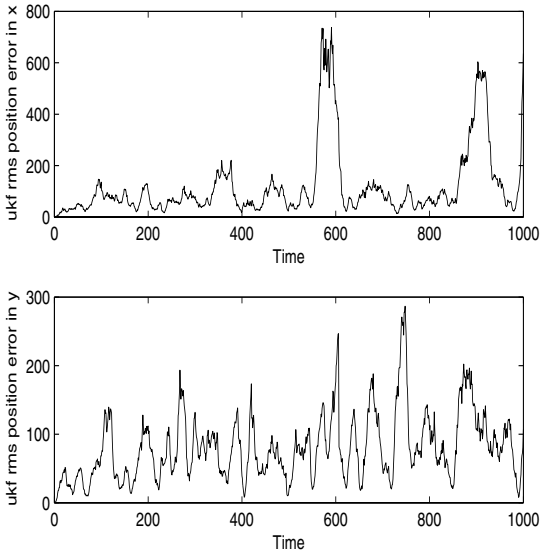

Fig. 6. *rms position errors in x and y directions for UKF.*

V. Conclusion

We have presented a recursive Bayesian formulation for target tracking and proposed a simple sensor scheduling technique in order to reduce power consumption of the system. In particular, we have formulated the target tracking problem using state-space equations. Tracking was considered as a sequential estimation problem and particle filtering algorithm was implemented. In order to schedule the sensors in our region of interest we have compared the position estimate of our target and the master nodes. We observed that our scheduling results are still quite satisfactory when we take

into account the decreasing detection quality of range only IR sensors with increasing distance. It is evident that, over all power consumption of the system is extremely low when compared to the case where no scheduling is done.

ACKNOWLEDGMENT

The authors would like to thank Habib Şenol for his contribution to the paper in the theory of Particle Filter.

REFERENCES

[1] M. S. Arulampalam, 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, Feb. 2002.

[2] F. Zhao, J. Shin, and J. Reich, “Information-Driven Dynamic Sensor Collaboration for Tracking Applications," *IEEE Signal Processing Magazine*, vol. 15, March 2002.

[3] S. Ghiasi, A. Srivastava, X. Yang, and M. Sarrafzadeh, “Optimal Energy Aware Clustering in Sensor Networks," *Sensors 2002,2,258-269*, July 2002.

[4] X. Sheng, Y-H. Hu, “Distributed Particle Filter with GMM Approximation for multiple target localization and tracking in wireless sensor networks," *Proc. IEEE/ACM Int. Symp. IPSN, Los Angeles, CA* , Apr 2005.

[5] C. Kreucher, K. Kastella, and A. O. Hero, “Multitarget Tracking using the Joint Multitarget Probabilty Density," *IEEE Trans. on Aerospace and Electronic Systems* , vol. 41, No. 4, October 2005.

[6] C. Hue, J.P. Le Cadre, and P. Pérez, “Sequential Monte Carlo Methods for Multiple Target Tracking and Data Fusion," *IEEE Trans. on Signal Processing.*, Vol 50,No. 2, Feb. 2002.

[7] S.J.Julier, J.K. Uhlmann, “A new extension of the Kalman Filter to Nonlinear Systems," *Proceedings of Aerosense: The 11th International Symposium on Aerospace/Defense, Sensing, Simulation and Controls, Multi Sensor Fusion, Tracking and Resource Management II, SPIE*, 1997.