TRACKING WITH PARTICLE FILTERING IN TERTIARY WIRELESS SENSOR NETWORKS

Petar M. Djurić, Mahesh Vemula, and Mónica F. Bugallo

Department of Electrical and Computer Engineering
Stony Brook University
Stony Brook, NY, 11794-2350
djuric, vema, monica@ece.sunysb.edu

ABSTRACT

Recent advances of wireless sensor networks have presented some very interesting problems for signal processing. For practical reasons, many networks are composed of simple sensors that use very little power and do not consume much communication bandwidth. A class of sensors that satisfy these requirements are the tertiary sensors. They report an approaching event with one signal and a receding event with another signal. When the event is out of their range, they do not report anything. In this paper, we apply particle filtering for processing signals from tertiary sensor networks with the purpose of tracking events (targets) within the field of the sensor network. We present an algorithm for tracking and demonstrate its performance by computer simulations.

1. INTRODUCTION

The most recent progress in low-power micro sensors, actuators, embedded sensors and radios have made wireless sensor networks one of the most exciting technological developments. The potential of wireless sensor networks is vast and can include applications in fields as diverse as environment monitoring (traffic, habitat, security), infrastructure integrity (e.g. power grids), battlefield tactical applications (target tracking), and medical applications [7]. There is no doubt that they will be deployed with increased success and their importance for homeland security, health care, manufacturing, monitoring of disaster areas, and meteorology may become unprecedented.

Research on signal processing for wireless sensor networks has also gained momentum because of the many unique signal processing challenges posed in the framework of these networks [7, 8]. One such class of sensor networks is known as tertiary sensor networks. They are designed to minimize the power needed for operation of the sensors as well as the communication bandwidth needed for signal transmission.

In brief, tertiary sensor networks are composed of sensors that emit information based on the strength of the sensed signal in consecutive instants and a central unit that collects the information from all the sensors and fuses it to produce estimates of a monitored event. A tertiary sensor operates as follows [1]: if the sensed signal is below a preset threshold, the sensor does not report anything (and thereby saves power), if the sensed signal is above the threshold and has increased in two consecutive time instants, it transmits a 1, and if it is above the threshold and has decreased in two consecutive time instants, it transmits a -1. The operation of such sensor is displayed in Fig. 1. When the sensed target is far from the sensor, the sensed signal is below the threshold and the sensor does not send any signals (instants t_1 and t_2). When the target enters the range of the sensor and approaches it, the sensor transmits a 1 (instants t_3 and t_4), and when it recedes from the sensor, the sensor transmits a -1 (instant t_5). As soon as the sensor leaves the range of the sensor, again this sensor does not transmit any signals (instant t_6).

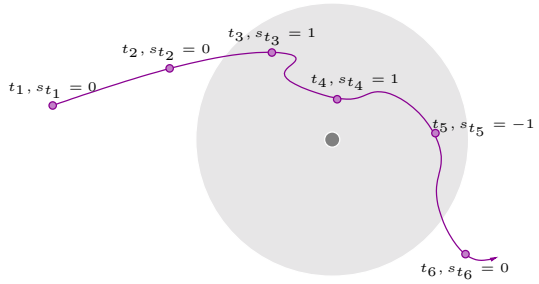

Fig. 1. A tertiary sensor with a target passing nearby. The signals transmitted by the sensors are denoted by s_{t_k}.

The central unit has considerable computing power and can run sophisticated algorithms for sequential estimation of the various unknowns of an observed event using the received sensor signals. In this paper, for sequential estimation of the unknowns we propose a particle filtering algorithm. Particle filtering is a sequential methodology for estimating unobserved states of an evolving system by using discrete random measures composed of particles (samples in the space of unknowns) and their associated weights [3, 4]. It is well known that particle filtering is particularly useful in scenarios where the studied system is described by nonlinear models and where the noise in the system may be non-Gaussian. This is precisely a setup that arises with tertiary wireless sensor networks. In this paper we present a particle filtering algorithm that processes signals from tertiary sensors and tracks a moving target. We also present simulation results that demonstrate the performance of the proposed particle filter.

2. PROBLEM STATEMENT

The wireless sensor network is composed of deterministically or randomly deployed sensors whose positions are known to the fu-

This work has been supported under Award CCR-0082607.

sion center. The fusion center also knows the model of the target movement, which is given by [5]

$$\mathbf{x}_t \quad = \quad \mathbf{G}_x \mathbf{x}_{t-1} + \mathbf{G}_u \mathbf{u}_t. \tag{1}$$

Here $\mathbf{x}_t = [x_{1,t},\ x_{2,t},\ \dot{x}_{1,t},\ \dot{x}_{2,t}]^\top \in \mathbb{R}^4$ is a state vector whose elements are the position and the velocity of the target in a two-dimensional Cartesian coordinate system, $\mathbf{G}_x$ and $\mathbf{G}_u$ are known matrices defined by

$$\mathbf{G}_x = \begin{pmatrix} 1 & 0 & T_s & 0 \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ and } \mathbf{G}_u = \begin{pmatrix} \frac{T_s^2}{2} & 0 \\ 0 & \frac{T_s^2}{2} \\ T_s & 0 \\ 0 & T_s \end{pmatrix}$$

where T_s denotes the sampling period, and $\mathbf{u}_t$ is a 2×1 vector that represents a Gaussian noise process with zero mean and known covariance matrix $\mathbf{C}_u = \mathrm{diag}(\sigma_{u1}^2, \sigma_{u2}^2)$, which accounts for the acceleration of the target.

The sensors measure the generated signal in log scale and the model representing the measurement of the $n-$th sensor, $n = 1, \ldots, N$ is

$$y_{n,t} \quad = \quad g_n(\mathbf{x}_{n,t}) + v_{n,t} \tag{2}$$

$$\quad = \quad \Psi_0 - 10\ \alpha_n \log_{10}\left(\frac{|\mathbf{r}_n - \mathbf{l}_t|}{d_0}\right) + v_{n,t} \tag{3}$$

where $g_n(\cdot)$ is a known function, Ψ_0 is the unknown emitted power of the target measured at a reference distance d_0, $\mathbf{r}_n \in \mathbb{R}^2$ is the position of the n-th sensor, $\mathbf{l}_t^\top = [x_{1,t}\, x_{2,t}]$ denotes the location of the target at time t, $|\cdot|$ denotes norm (length) of a vector, α_n is a known attenuation parameter which depends on the transmission medium, and $v_{n,t} \sim \mathcal{N}(0, \sigma_v^2)$ is a noise process, where the variance σ_v^2 is assumed known.

The n-th sensor ($n = 1, \ldots, N$) measures the received power $y_{n,t}$, processes it locally and generates a signal if $y_{n,t} > \gamma$, where γ is a threshold of the sensor. The signal is generated according to

$$s_{n,t} = \begin{cases} 1, & \text{if} \quad y_{n,t} - y_{n,t-1} > 0 \ \text{ and } \ y_{n,t} > \gamma \\ -1, & \text{if} \quad y_{n,t} - y_{n,t-1} < 0 \ \text{ and } \ y_{n,t} > \gamma \end{cases}. \tag{4}$$

If $y_{n,t} < \gamma$, the sensor does not emit anything. When $|s_{n,t}| = 1$, the central unit receives a signal of the form

$$z_{n,t} = \beta_n s_{n,t} + \epsilon_{n,t} \tag{5}$$

where $\epsilon_{n,t} \sim \mathcal{N}(0, \sigma_\epsilon^2)$, and β_n is an attenuation coefficient associated with the $n-$th sensor. Otherwise,

$$z_{n,t} = \epsilon_{n,t}. \tag{6}$$

We assume that the noise parameters σ_ϵ^2 and the coefficients β_n are known.

The main task is the tracking of the evolving state $\mathbf{x}_{0:t}$ using the observations $\mathbf{z}_{1:t} = [z_{1,1:t}, \ldots, z_{N,1:t}]^\top$.

3. PARTICLE FILTERING

In the past decade, particle filtering has received considerable attention in the engineering literature [2, 3, 4]. Particle filtering is based on the concept of importance sampling and Bayesian theory. Here we do not provide any review of the methodology, and instead, we refer the reader to the cited items and the references therein. In this section we only provide the terminology and the notation.

The sought information about the unknown state $\mathbf{x}_{0:t}$ is completely captured by the posterior $p(\mathbf{x}_{0:t} \mid \mathbf{z}_{1:t})$. With particle filtering, the posterior, $p(\mathbf{x}_{0:t} \mid \mathbf{z}_{1:t})$, is approximated by a set of particle streams $\mathbf{x}_{0:t}^{(m)}$, $m = 1, 2, \cdots, M$ and their weights $w_t^{(m)}$. As soon as a new observation $\mathbf{z}_{t+1}$ is available, the stream of particles $\mathbf{x}_{0:t}^{(m)}$ is expanded to $\mathbf{x}_{0:t+1}^{(m)}$, $m = 1, 2, \cdots, M$ and the weights are recursively updated to $w_{t+1}^{(m)}$. The steps of implementing particle filtering algorithms are known as particle generation (drawing of $\mathbf{x}_{t+1}^{(m)}$ using an importance function and appending it to $\mathbf{x}_{0:t}^{(m)}$), weight computation, and resampling. The last step, resampling, is needed for avoiding degeneracy of the random measure. With resampling, streams of particles that have large weights are replicated, and those with negligible weights are removed.

4. TRACKING ALGORITHM

In this section we show the development of a particle filtering algorithm for tracking of one target. First, we form particles of the form

$$\tilde{\mathbf{x}}_t^{(m)} = [x_{1,t}^{(m)},\ x_{2,t}^{(m)},\ \dot{x}_{1,t}^{(m)},\ \dot{x}_{2,t}^{(m)}\ \Psi_{0,t}^{(m)}\ y_{n,t}^{(m)}]^\top$$

where $\Psi_{0,t}^{(m)}$, although constant, is represented as a time-varying parameter. In simplifying the implementation of the algorithm, we use the *priors* $p\,(\tilde{\mathbf{x}}_t \mid \tilde{\mathbf{x}}_{t-1})$ as importance functions.

The computation of the weights is then obtained from

$$w_t^{(m)} \propto w_{t-1}^{(m)} \prod_{n=1}^{N} p\,(z_{n,t} \mid \tilde{\mathbf{x}}_t^{(m)}). \tag{7}$$

For the factors $p(z_{n,t} \mid \tilde{\mathbf{x}}_t^{(m)})$, we can write

$$p(z_{n,t} \mid \tilde{\mathbf{x}}_t^{(m)}) \quad = \quad \sum_{k=-1}^{1} p(z_{n,t} \mid s_{n,t} = k, \tilde{\mathbf{x}}_t^{(m)}) \times P(s_{n,t} = k \mid \tilde{\mathbf{x}}_t^{(m)})$$

$$\quad = \quad \sum_{k=-1}^{1} p(z_{n,t} \mid s_{n,t} = k) P(s_{n,t} = k \mid y_{n,t}^{(m)}) \tag{8}$$

where

$$p(z_{n,t} \mid s_{n,t} = k) = \mathcal{N}(\beta_n k, \sigma_\epsilon^2) \tag{9}$$

and

$$P\left(s_{n,t} = k \mid y_{n,t}^{(m)}\right) = \begin{cases} 1, & \text{if } \mathcal{C}_k \text{ holds} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where $k = -1, 0, 1$, and

$$\mathcal{C}_{-1} \quad \equiv \quad \{y_{n,t} - y_{n,t-1} \leq 0 \ \text{ and } \ y_{n,t} > \gamma\} \tag{11}$$

$$\mathcal{C}_0 \quad \equiv \quad \{y_{n,t} \leq \gamma\} \tag{12}$$

$$\mathcal{C}_1 \quad \equiv \quad \{y_{n,t} - y_{n,t-1} > 0 \ \text{ and } \ y_{n,t} > \gamma\}. \tag{13}$$

It is obvious that the three conditions describe events that are mutually exclusive.

The steps of the particle filtering algorithm can be implemented as follows:

1. Initialization:

The particles $\tilde{\mathbf{x}}_0^{(m)}$, $m = 1, 2, \cdots, M$, are initially drawn from a prior distribution $\pi(\mathbf{x}_0)$, and the weights of the particles are set to $\frac{1}{M}$.

2. New particle generation:

The first two elements of $\tilde{\mathbf{x}}_t$ are the location of the target in a two-dimensional space, and the next two elements represent the velocity in this space. In accordance with the movement model, we only have to generate the location or velocity components, and obtain the rest deterministically. For example, first we can generate the velocity components by $p(\dot{x}_{1,t}, \dot{x}_{2,t}|\dot{x}_{1,t-1}, \dot{x}_{2,t-1})$ or $p(\dot{x}_{1,t}, \dot{x}_{2,t}|\dot{x}_{1,t-1}, \dot{x}_{2,t-1}, \mathbf{z}_t)$ and second, compute the locations from

$$x_{1,t}^{(m)} = x_{1,t-1}^{(m)} + \frac{T_s}{2}\left(\dot{x}_{1,t}^{(m)} + \dot{x}_{1,t-1}^{(m)}\right) \quad (14)$$

$$x_{2,t}^{(m)} = x_{2,t-1}^{(m)} + \frac{T_s}{2}\left(\dot{x}_{2,t}^{(m)} + \dot{x}_{2,t-1}^{(m)}\right). \quad (15)$$

For the element $\Psi_{0,t}$, we employ the following scheme [6]. Let

$$\Psi_{0,t} = \Psi_{0,t-1} \quad (16)$$

and thus the prior proposal density of $\Psi_{0,t}$ should be

$$p(\Psi_{0,t} \mid \Psi_{0,t-1}^{(m)}) = \delta(\Psi_{0,t} - \Psi_{0,t-1}^{(m)}) \quad (17)$$

where $\delta(\cdot)$ is the Dirac delta function. At time $t-1$, we approximate $p(\Psi_{0,t-1} \mid \mathbf{z}_{1:t-1})$ with a Gaussian density (other densities may also be used). We then compute the mean and variance of $\Psi_{0,t-1}$, from the particles $\Psi_{0,t-1}^{(m)}$, $m = 1, 2, \cdots, M$ by

$$\mu_{\Psi_{0,t-1}} = \sum_{m=1}^{M} w_{t-1}^{(m)} \Psi_{0,t-1}^{(m)}$$

$$\sigma^2_{\Psi_{0,t-1}} = \sum_{m=1}^{M} w_{t-1}^{(m)} \left(\Psi_{0,t-1}^{(m)} - \mu_{\Psi_{0,t-1}}\right)^2. \quad (18)$$

Then we resample from $\mathcal{N}\left(\mu_{\Psi_{0,t-1}}, \sigma^2_{\Psi_{0,t-1}}\right)$, i.e.,

$$\Psi_{0,t-1}^{(m)} \sim \mathcal{N}(\mu_{\Psi_{0,t-1}}, \sigma^2_{\Psi_{0,t-1}}). \quad (19)$$

The drawn $\Psi_{0,t-1}^{(m)}$'s remain unchanged at time instant t, that is,

$$\Psi_{0,t}^{(m)} = \Psi_{0,t-1}^{(m)}. \quad (20)$$

Finally, the elements $y_{n,t}^{(m)}$ are easily drawn from

$$y_{n,t}^{(m)} \sim \mathcal{N}(g_n(x_{1,t}^{(m)}, x_{2,t}^{(m)}), \sigma_v^2).$$

3. Weight computation:

The computation of the weights is carried out recursively by (8). The factors $p(z_{n,t} \mid \tilde{\mathbf{x}}_t^{(m)})$ are given by (8)–(13) . When the weights are normalized, we use them and the particles to find desired estimates of the unknown states. For instance, if the MMSE estimates are needed, we compute them by

$$\widehat{\tilde{\mathbf{x}}}_t = \sum_{m=1}^{M} w_t^{(m)} \tilde{\mathbf{x}}_t^{(m)}. \quad (21)$$

Before propagating the particles for the next time instant $t+1$, we may resample them.

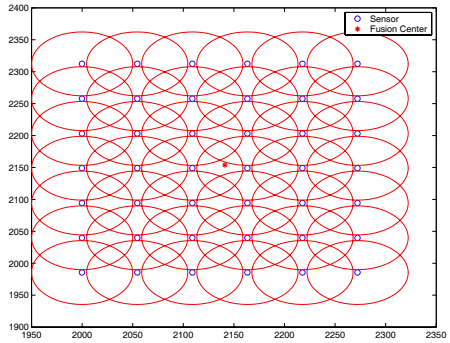

Fig. 2. Layout of the deployed sensor network. The position of the sensors is indicated by small circles and the sensing boundaries of the sensors are depicted by larger circles. The fusion center is represented by an asterisk. The distance units are meters.

5. SIMULATION RESULTS

In this section we present some simulations that illustrate the performance of the proposed algorithm for target tracking. In the simulations we considered a sensor network with sensors uniformly placed on a grid. Other possible sensor placements may include random and other deterministic constellations. The deployed sensors had a sensing radius of 50m. The ability of the sensor to detect the presence of targets depended on the received power and the sensing radius. If the sensed signals were above a preset threshold, the sensors reported to the fusion center (placed in the middle of the field) with a signal according to (4). The sensor network is shown in Fig. 2.

The state vector particles involving the kinematic parameters of the target were initially drawn from a Gaussian distribution function with a known mean $\bar{\mathbf{x}}_0$ and covariance matrix $\boldsymbol{\Xi}$

$$\bar{\mathbf{x}}_0 = \begin{bmatrix} 2000 \\ 2000 \\ 5 \\ 5 \end{bmatrix} \text{ and } \boldsymbol{\Xi} = \begin{bmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix} \quad (22)$$

while $\Psi_0^{(m)}$ was generated from a uniform distribution $\mathcal{U}(\Psi_a, \Psi_b)$, with $\Psi_a = -80$ and $\Psi_b = 0$.

The simulation parameters were set as follows: the state covariance matrix $\mathbf{C}_u$ = diag$(0.5, 0.5)$, the variance of the noise process $v_{n,t}$ was $\sigma_v^2 = 1$, the sampling period $T_s = 1$, and the attenuation parameter α = 2.3, which was the same for all the sensors. Recall that the fusion center receives a highly attenuated information signal from the sensors. The probability of false alarm in detecting a signal from the sensor was $P_{\text{FA}} = 0.015$. In the implementation of the particle filtering algorithm, $M = 2000$ particles were used and resampling was performed at every step. Since at a particular instant of time only very few sensors, which are in the vicinity of the target, contain meaningful information about the target, only a subset of the sensors were used in calculating the likelihoods.

In Figs. 3 and 4 we show the root mean square errors of the estimated kinematic and power parameters of the target. The errors were computed over 500 runs with 100 different target trajectories drawn from the same distribution $p(\mathbf{x}_{0:t})$. It can be seen that a mean square error of about 8.5m in the target's position is attained using this algorithm. A target trajectory and its estimates

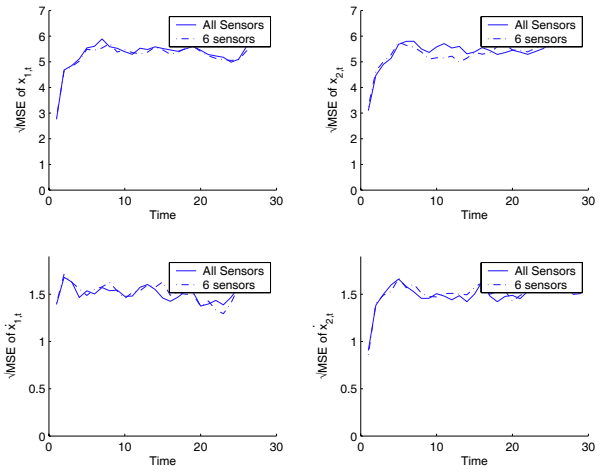

Fig. 3. Root mean square errors of the kinematic parameters.

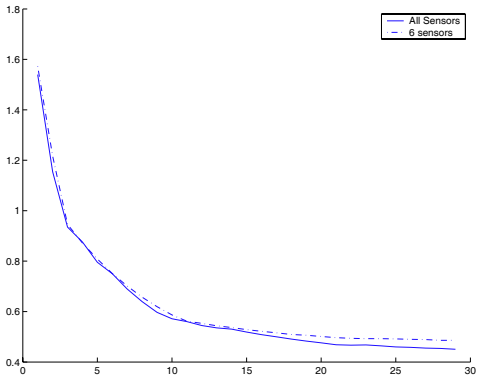

Fig. 4. Root mean square error of Ψ_0.

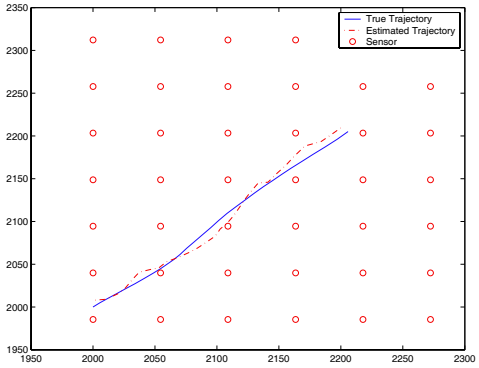

Fig. 5. A single run of a system trajectory estimation.

are shown in Fig. 5. As can be seen in Figs. 3 and 4, the mean square error performance is almost the same when one considers 6 sensors around the vicinity of the target and all the sensors in the network. This suggests that a reduction of computational complexity of the algorithm by several orders of magnitude is possible. Fig. 6(a) shows the evolution of Ψ_0 with time, where the true value of Ψ_0 is represented with a dotted line. The other three subplots show the posterior density of Ψ_0 which is approximated by a Gaussian kernel. It can be seen that the conventional particle

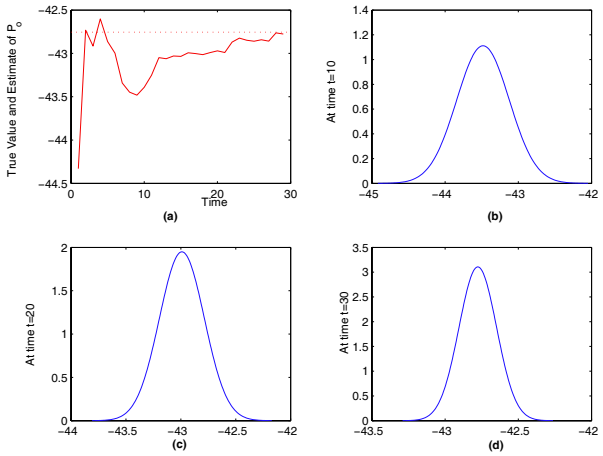

Fig. 6. (a) Evolution of Ψ_0 with time. (b,c,d) represent the posterior densities of Ψ_0 at t=10,20,30s

filtering algorithm provides good tracking results.

6. CONCLUSIONS

In this paper we presented a particle filtering algorithm for a tertiary wireless sensor network for tracking a moving target in a two-dimensional plane. Even though the sensors provide highly compressed information about the moving target, the central unit of the sensor network was capable of estimating the targets trajectory and its velocity with good accuracy.

7. REFERENCES

[1] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," in *the Proceedings of the First International Conference on Embedded Networked Sensor Systems*, Los Angeles, CA, 2003, pp. 150–161.

[2] P. M. Djurić and S. J. Godsill, Eds., *Special Issue on Monte Carlo Methods for Statistical Signal Processing*, vol. 50, IEEE Transactions on Signal Processing, 2002.

[3] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Míguez, "Particle filtering," *IEEE Signal Processing Magazine*, vol. 20, no. 5, pp. 19–38, 2003.

[4] A. Doucet, N. de Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*, Springer, New York, 2001.

[5] F. Gustaffson, F. Gunnarsson, N. Bergman, U. Forssel, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filtering for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.

[6] J. Kotecha and P. M. Djurić, "Gaussian particle filtering," *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2592–2601, 2003.

[7] Z. Kumar, F. Zhao, and D. Shepherd, Eds., *Collaborative Information Processing, IEEE Signal Processing Magazine*, March 2002.

[8] K. Yao, D. Estrin, and Y. H. Hu, Eds., *Special Issue on Sensor Networks, EURASIP Journal on Applied Signal Processing*, vol. 2003, 2003.