SENSOR NETWORK PARTICLE FILTERS: MOTES AS PARTICLES

Mark Coates and Garrick Ing

Department of Electrical Engineering
McGill University
3480 University St, Montreal, QC, Canada H3A 2A7

ABSTRACT

We describe an algorithm for tracking an object using particle filtering in a sensor network comprised of smart dust-type motes. We investigate the situation where the motes are equipped with binary proximity sensors, low-power lasers and optical receivers for communication with nearby motes, and corner-cube arrays for communication with a central transceiver. The particle filter we describe is largely decentralized; a central transceiver performs no processing beyond a summation and weighted average. Individual motes act as the particles in that they represent candidate positions of the object. Propagation of the particle filter is performed through activation of appropriate neighbouring nodes with "weighted" messages. We provide simulation results of tracking a maneuvering object, comparing performance with a centralized particle filter.

1. INTRODUCTION

Smart dust networks, proposed in [1, 2], consist of many millimetre-scale motes, each equipped with a small power supply and capable of sensing, communication and limited computation. Due to the extreme constraints on energy availability, optical line-of-sight communication is attractive in such networks, because it can be designed to consume much less energy than radio-frequency wireless communication. An external querying transceiver, consisting of steerable lasers and a directionally sensitive optical receiver array, can further reduce mote power requirements, because motes can reflect light in a controlled fashion for long-range communication, rather than generating it themselves. Networking procedures such as localization, routing, synchronization and clustering can be very difficult to perform in a smart dust environment, so any distributed signal processing algorithm should avoid them as much as possible.

In this paper, we focus on the problem of using a smart dust-type network to track an object maneuvering through a region. Each mote of the network is equipped with a proximity sensor with relatively weak detection capability. We develop a tracking algorithm that is a type of (approximate) distributed particle filter [3] and explore the performance of the algorithm relative to a centralized particle filter. Our objective is to achieve reasonably accurate estimation whilst minimizing the amount of network configuration overhead and maximizing network lifetime by reducing the number of active sensors and the network communication.

The paper is structured as follows. In Section 2 we describe the architecture of the sensor network and the communication model. Section 3 specifies the dynamic model for object motion and the measurement framework. Section 4 describes the tracking problem we address and provides a high-level review of particle filters. Section 5 discusses issues with distributed particle filtering in sensor networks and then details our proposed approach. Section 6 describes the simulations we have performed to explore the performance of the algorithm and examines the results. Finally, Section 7 discusses related work, limitations of the approach and intended extensions.

2. NETWORK ARCHITECTURE AND COMMUNICATION

We consider a sensor network comprised of a set of motes, each equipped with a binary proximity sensor, a controllable reflective device (a corner-cube array) for communication with a global querying transceiver, and low-power lasers and direction-sensitive optical receivers for communication with neighbouring motes. Using such a network, we design an architecture and algorithm for tracking an object moving through a monitored region. Figure 1 depicts the main properties of the global architecture. The binary proximity sensor attached to each mote has a detection region; there is an associated probability of detection when the object is in this region and a probability of false-alarm when the object is outside the region.

Each mote has a set of low-power lasers for communication with neighbouring motes; these can be steered in any of eight directions. Motes transmit at different wavelengths; the number of unique wavelengths is such that within any communication neighbourhood there is no duplication. This

This research was supported by NSERC.

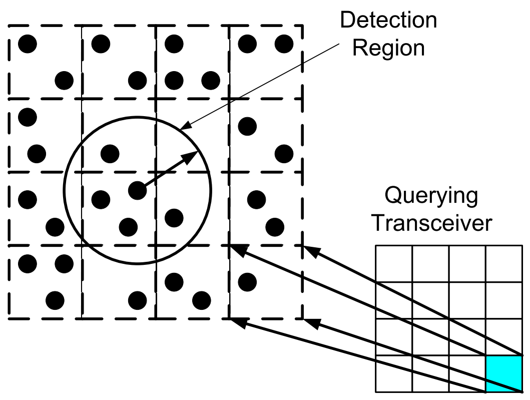

Fig. 1. Architecture of the sensor network. The binary sensor of each mote has a detection region, and an associated probability of detection and false-alarm. Communication between motes is accomplished through the use of a small set of lasers and directionally-sensitive optical receivers. The central querying transceiver communicates with the network by projecting light onto the network; very low bit-rate signals are embedded by modulation. Communication back to the transceiver is performed by controlled reflection at the motes.

prevents interference of transmissions and removes the need for accurate synchronization. The motes are also equipped with optical receivers that can determine the direction of incident communication (to within a forty-five degree region that corresponds to one of the eight directions). During the particle filter operation, a mote must be capable of restricting its communication to a small region of the sensor field (see Figure 2). This is achieved by setting minimum and maximum thresholds on the power level at a receiver for recognition of a valid transmission. The communication rate between motes is very low, since a mote must only transmit a few weights (each weight is several bytes) at any measurement instant. Communication with the central transceiver is achieved solely through reflection. Each mote modulates the power of the reflected light according to its current total weight.

We do not assume any alignment of the motes or knowledge of position. We do implicitly assume some level of coarse synchronization in this paper, primarily for ease of exposition, but the algorithm can be readily implemented in an asynchronous fashion with some minor changes.

3. OBJECT DYNAMICS AND MEASUREMENT

In this paper, we consider the situation where the dynamics of the object can be adequately captured by a jump Markov model [4], but the observations (binary proximity measurements) are non-linear functions of the object state. Our approach is generalizable to more other Markovian dynamic models (those that have sufficient mixing properties [5]).

Let $r_t, t = 1, 2, \dots$ denote a discrete-time Markov chain with a finite set of states and known transition probabilities. The object dynamics of a general jump Markov linear system can be modelled as:

$$x_{t+1} = A(r_{t+1})x_t + B(r_{t+1})\eta_{t+1} + F(r_{t+1})u_{t+1} \quad (1)$$

Here u_t denotes an exogenous input and η_t is an independent white Gaussian noise sequence. In the particular case we consider in this paper, we use r_t to model the maneuvering of the object (straight, left-turn, right-turn) and x_t to model the position.

Our observations consist of binary variables y_t collected at a set N_t of active sensor motes. We model the observations as conditionally independent given object position, so for any active mote j:

$$p(y_t^{(j)} = 1) = \begin{cases} p_d & \text{if } x_t \in D^{(j)}, \\ p_f & \text{if } x_t \notin D^{(j)}. \end{cases} \quad (2)$$

where $D^{(j)}$ is the detection region of the sensor of mote j.

We denote the mote positions as $\{v^{(j)}; j = 1, \dots, N\}$. When the mote measurements are modelled as independent, the likelihood function is equal to the product of the observation probabilities. For a position x_t, the likelihood function is:

$$\mathcal{L}(y_t|x_t) = p_d^{n_t}(1-p_d)^{s_t}(1-p_f)^{g_t}p_f^{f_t} \quad (3)$$

Here $n_t + s_t$ is the number of active sensors whose detection region includes the position x_t; n_t is the number of these which record a positive detection. In a similar fashion, $g_t + f_t$ is the number of active sensors whose detection regions exclude x_t, and f_t is the number recording a positive detection. If there are N_t active sensors at time t, $N_t = n_t + s_t + f_t + g_t$.

In this paper, we focus on the case where the detection capability of the sensor has been maximized at the expense of a high false alarm rate. In particular, we assume that when the object is not in range, a sensor has equal probability of reporting either result, i.e. $p_f = 0.5$. This model implies that a sensor can calculate the likelihood of the object being at its location (to within a proportionality constant) based solely on neighbourhood measurements. It does not need to know the number of active motes, but requires a (conservative) upper bound $n_{\max}$ on the maximum number of motes in the detection region of any position. The likelihood evaluation is:

$$\widehat{\mathcal{L}}(y_t|x_t = v^{(j)}) = p_d^{n_t^{(j)}}(1-p_d)^{s_t^{(j)}}0.5^{n_{\max}-n_t^{(j)}-s_t^{(j)}} \quad (4)$$

where $n_t^{(j)}$ and $s_t^{(j)}$ are, respectively, the number of sensors in the detection neighbourhood of sensor j that report positive and negative responses.

4. PARTICLE FILTERS AND TRACKING

Our goal is to sequentially estimate (or track) the position of the object. The state of the object includes its position, direction and current maneuver. The non-linear nature of the measurements means that a decentralized Kalman filter [6] is an inappropriate choice for performing this task. Alternative approaches include extended Kalman filters, grid-based methods and Gaussian-sum filters [7, 8, 9], but these all have serious limitations on estimation performance, and information exchange between motes is not a simple matter. Particle filtering methods [10, 11] are attractive because of their tracking power and modelling flexibility, but they have substantial computational requirements, impose a potentially high communication overhead, and can require careful sensor network configuration.

Particle filtering methods keep track of a set of candidate state trajectories (particles) [11]. There is a weight associated with each of these particles; note that these weights are not necessarily normalized. When a new measurement becomes available, each particle's state trajectory is augmented based on the dynamic model (and possibly the measurement) in a *propagation* step. Its weight is then updated according to how well it conforms to the dynamic model and the likelihood of it generating the current measurement (the *update* step). The set of weighted particles forms a weighted pointwise approximation to the filtering distribution, and this can be used to form estimates of the current state of the object (the *estimation* step). In practice, an additional *resampling* step must be incorporated in the particle filter; this step eliminates particles with very low weight and duplicates those with large weight. In this manner, a reasonable number of particles is sufficient to focus on the likely region of the state-space.

5. SEMI-DISTRIBUTED PARTICLE FILTERING ALGORITHM

There are a number of difficulties associated with a distributed implementation of particle filtering methods in sensor networks. In this paper, we interpret a distributed implementation as meaning that disjoint subsets of the particle set are maintained at different motes. By distributing the particles amongst different motes, we can reduce the computational burden to within mote capability (computation is linear in the number of particles). At any point in time there is a relatively small number of active motes that maintain particle sets, so we can conserve system energy by deactivating the majority of motes.

In the sensor network, each mote only has access to its own sensor measurements. In the propagation step of the particle filter, this is not critical if augmentation is based solely on the dynamic model (although imposing this limitation can have a significant impact on the performance of the particle filter). However, the update step requires evaluation of the likelihood, which is in general dependent on all measurements. Our model in this paper results in a likelihood function that is only dependent on the binary measurements in the local neighbourhood of the mote, so we can achieve evaluation of the likelihood by requiring all active sensors to broadcast their binary proximity measurement to their neighbours. It is possible to evaluate (approximately) the likelihood in this local fashion for a reasonably broad range of measurement models, but we do not elaborate on the model requirements here.

The estimation and resampling steps both require knowledge of *all* particle weights. A minimum mean-squared error estimate of the object position is formed by the weighted average of the particle positions. The resampling step requires normalization of the weights of all particles in the system. This implies that each active sensor mote must be informed of the total particle weight. We do not propose a method for distributed implementation of these steps in this paper. The global transceiver is used to perform the weighted averaging for estimation and also transmits the total weight to all active sensor motes whenever resampling is performed.

We must also address the challenge of changing the set of active motes over time. We want the active motes to be clustered around the true position of the object, so that the most informative sensor measurements are made. When the object is about to leave the detection region of a currently active mote, that mote should transfer its particle set to another mote and deactivate.

5.1. Implementation

We adopt a semi-distributed approach in the particle filtering algorithm we develop. Disjoint sets of particles are independently maintained at a small set of active motes, but a global transceiver performs a few simple averaging operations to facilitate resampling and extract the estimate from the network. An important innovation in our tracking algorithm is that all the particles maintained at any given sensor j have position $x_t = v^{(j)}$, where $v^{(j)}$ is the position of the sensor. This restricts the flexibility and accuracy of the particle filter, but with reasonable density of motes, it does not impose a substantial tracking penalty (see Section 6).

At time t_0, the system is initialized by activating all motes in the system. In this initial phase, each mote is allotted eight particles, one for each of the directions it is capable of signalling. The x_0 value for each of these particles is equal to the (uncalibrated) position of the sensor. The r_0 value of all particles is set to "straight". The weight of each particle is initialized to one.

Propagation step: Particles are propagated by activating neighbouring motes in the direction of motion; a message is sent containing the particle weight and discrete state. As an example, if a given mote received a particle with a message containing $\{straight, 0.2\}$ as the state and weight from the west (where west is simply a label that the mote associates with one of its receivers), then on the next time-step the mote may generate two particle descendants, one with message $\{left, 0.15\}$ that it transmits via its north-east laser, and one with message $\{straight, 0.1\}$ that it transmits via its east laser.

These generated messages are sent to a region (see Figure 2) and are thus not necessarily confined to a single mote. The number of descendants of a given particle in the representation is dependent on the number of motes that lie within the communication region. If motes are uniformly distributed in the region of observation, this does not present a problem, aside from imparting some additional variance to the estimate.

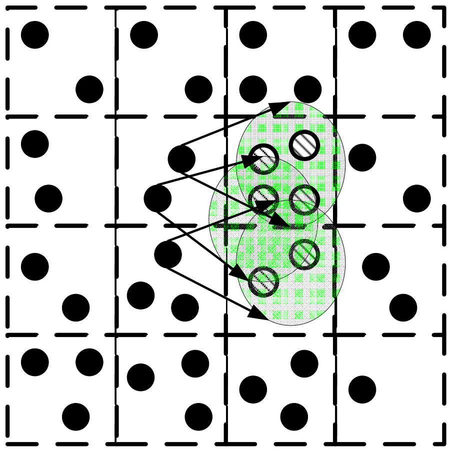

Fig. 2. Particle propagation in the sensor network. Motes activate new motes in the predicted direction of object travel. These become the new particles in the filter.

When choosing the size and location of the communication region, we attempt to match the possible region of movement determined by the dynamic model. For instance, if there is the potential for substantial variation in the direction or distance moved then the region will be larger. If we achieve a perfect match between the probability of communication to a mote and the probability of movement to that mote according to the dynamic model, then the importance sampling distribution (see [11]) is the prior. In our setup, the match is only very approximate, because the communication cannot mimic any complicated distribution, but given the limitations of the measurements, this approximation does not impart unduly large errors relative to other aspects of the monitoring system. If the dynamic model does not have substantial uncertainty relative to the density of the motes, then the communication region needs to be larger than the region dictated by the prior, in order to ensure that at least one mote is contacted.

Update step: The update step is performed in three stages. First, a mote with particles broadcasts a message to activate all motes in its detection neighbourhood. Second, these motes perform a proximity measurement and broadcast the result. Third, each particle-maintaining mote calculates its likelihood according to (4) and updates the weight of its particles by multiplying by the likelihood (in the standard manner of particle filters where the importance sampling distribution is the prior [11]).

Estimation step: The global transceiver sends an inquiry beam requesting all active motes to respond. Each active mote modulates the reflected power according the total weight of all its particles. The global transceiver is equipped with a receiver array and can perform an averaging of the received power to form an estimate of object location.

Resampling step: The global transceiver sends a message to every active mote containing the total weight of the system. Each mote can then normalize the weights of its particles and perform resampling. Each individual mote performs systematic resampling [12] on its own particle set. This results in a variable number of total particles in the system, but the propagation step is the dominant source of variability in the number of particles.

6. SIMULATIONS

We conducted Matlab simulations to compare the performance of the distributed algorithm relative to a centralized particle filter. The simulations used a sensor network occupying a 128×128 metre plane. The number of motes in the sensor network was varied to explore the effect of mote density on tracking performance.

The dynamic system of the object's movement employs a jump-state Markov model, described by an initial distribution $p(u_0, \theta_0, \mathbf{x}_0)$ and update equations

$$u_t \sim p(u_t|u_{t-1}), \tag{5}$$

$$\theta_t = \theta_{t-1} + c(u_t) + \epsilon_t, \tag{6}$$

$$x_t = x_{t-1} + m[\cos\theta_t, \sin\theta_t], \tag{7}$$

where $u_t \in \{0, 1, 2\}$ represents a discrete motion state of the object (continuing straight, making a 0.3 radian left turn or making a 0.3 radian right turn, respectively). $c(u_t)$ represents the angle of turn in radians. The angle of the motion is represented by θ_t, which has a zero-mean Gaussian innovation noise ϵ_t of variance 0.001. The object's position is x_t and its speed is constant at $m = 0.5$. The update probability matrix for the discrete state is

$$p(u_t|u_{t-1}) = \begin{pmatrix} 0.75 & 0.65 & 0.65 \\ 0.125 & 0.3 & 0.05 \\ 0.125 & 0.05 & 0.03 \end{pmatrix}$$

Each sensor has a detection radius of 8 metres and a probability of detection $p_d = 0.7$. We do not model any errors in communication. We use a communication region of radius 2 metres for the propagation of particles, which is substantially larger than the region of motion determined by the dynamic model, but the increase is necessary because of the sparse mote density.

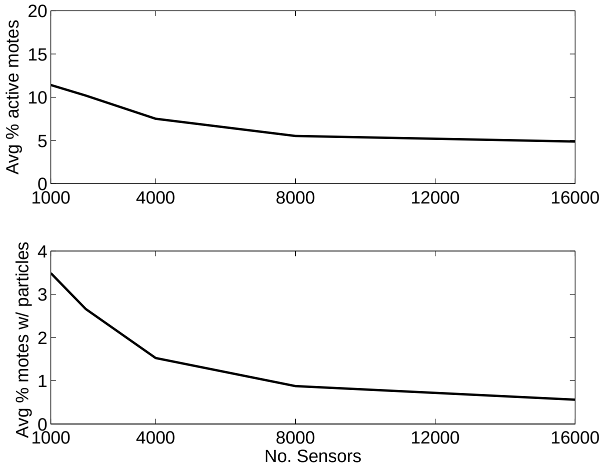

Fig. 3. The average percentage of active motes per time interval as a function of the number of sensors in the network. Top panel: the average percentage of motes that make a measurement. Bottom panel: the average percentage of motes maintaining particles.

As in the algorithmic description, a mote is activated whenever it is required to make a measurement. In our simulations, this activation persists for 5 time intervals if no subsequent activation messages are received. This determines the set of active motes at any time instant. A subset of these motes maintain particles.

We conducted simulation experiments with 4000, 8000 and 16000 sensors, performing 25 trials in each case. These values represent densities of approximately 0.25, 0.5 and 1 mote per square metre. We compared the tracking performance of the distributed particle filtering algorithm described in the previous section with a centralized particle filter. The centralized particle filter propagates particles according to the dynamic model and uses 2000 particles. The distributed filter uses approximately 2000 particles, but this number is subject to the variability discussed above, and there is substantial duplication of particles because of quantization effects.

Figure 3 displays the average percentage of active motes in the system as a function of the number of sensors. Over

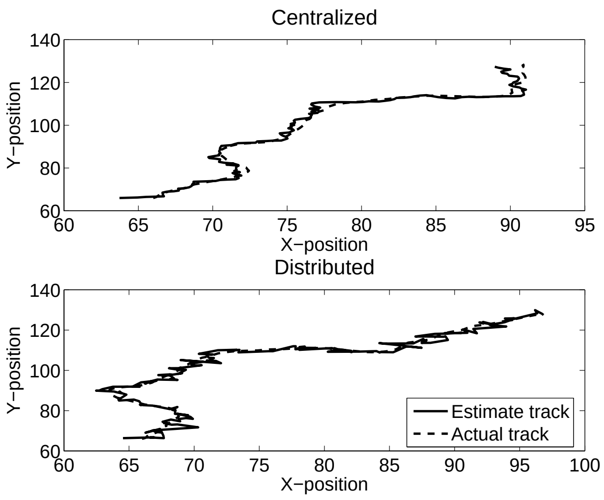

Fig. 4. Examples of tracking performance for 16,000 sensors. Top panel: centralized particle filter; bottom panel: distributed particle filter.

the range from 4000 to 16000 sensors, the percentage is reasonably constant, slowly decaying from approximately 7 to 5 percent. This corresponds to approximately 300, 400 and 800 motes respectively. The average percentage of motes maintaining particles at any instant ranges from 1.5 to 0.5 percent, corresponding to approximately 60-80 motes in each case. Each one of these motes is thus responsible (on average) for approximately 25 particles, although it should be noted that there are only 24 possible particle-types at any sensor (8 directions and 3 maneuvers). It is clear that the distributed particle filter approach achieves a dramatic reduction in the number of active motes (and hence the energy expenditure of the system). It also provides an effective distribution of the computational requirements of the particle filtering algorithm.

Figure 4 shows examples of tracking behaviour for the centralized and distributed algorithms in the case of 16000 motes. The distributed filter is much less smooth in its estimates, primarily due to the sparsity of motes. However, in terms of average mean squared error, the performance of the two algorithms for this density of motes is comparable, as indicated by Figure 5. The relative performance of the distributed algorithm deteriorates as the number of sensors is decreased. Particle positions must correspond to the associated mote positions, and if there are too few motes, the resultant particle distribution is a poor approximation to the filtering distribution.

7. DISCUSSION

7.1. Related Work

The problem of tracking in sensor networks has been addressed by numerous authors, although most work has ei-

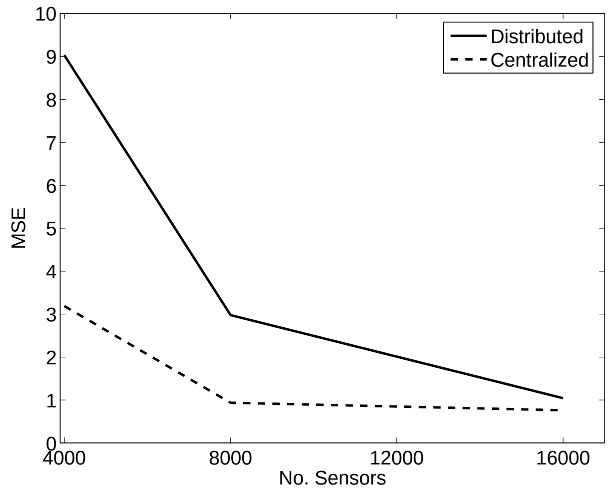

Fig. 5. A comparison of the average mean-squared error in position estimation as a function of the number of network motes.

ther proposed accumulation of the data at a central site or investigated the case of linear dynamics and observations [13, 14, 15, 16]. Distributed particle filters for sensor networks have been proposed in [3, 17, 18], although these approaches require substantial computational power and memory at some motes. Distributed implementation of particle filters and resampling algorithms has been explored in [19], but the focus was on distributing computation; the communication overhead of the proposed approach is unreasonable for a sensor network application.

7.2. Conclusion, Limitations and Future Work

We have described a smart-dust type sensor network and an associated distributed particle filtering algorithm for tracking an object maneuvering through the monitored region. The approach is particularly novel in its strong linkage between motes and particles and particle propagation and communication. The algorithm achieves good tracking performance in relatively sparse networks with weak sensors and requires only local communication and minimal network configuration. Currently, the main disadvantage of the algorithm is the need for a central transceiver for resampling. Ideally, the network should perform tracking autonomously and only respond to a mobile, querying transceiver periodically for estimation. This requires the development of completely distributed resampling, possibly through in-network algorithms for weight aggregation and (approximate) normalization. More extensive simulations and prototyped systems are required to properly examine system performance when detection and communication behaviour do not match the idealized models explored here.

8. REFERENCES

[1] V. Hsu, J. M. Kahn, and K. S. J. Pister, “Wireless communications for smart dust,” Tech. Rep., Electronics Research Laboratory Tech. Memo. M98/2, Feb. 1998.

[2] J.M. Kahn, R.H. Katz, and K.S.J. Pister, “Mobile networking for smart dust,” in *Proc. IEEE Intl. Conf. MobiCom*, Seattle, WA, Aug. 1999.

[3] M.J. Coates, “Distributed particle filtering for sensor networks,” in *Proc. Int. Symp. IPSN*, Berkeley, CA, Apr. 2004.

[4] A. Doucet, N.J. Gordon, and V. Krishnamurthy, “Particle filters for state estimation of jump Markov linear systems,” *IEEE Trans. Signal Proc.*, vol. 49, pp. 613–624, Mar. 2001.

[5] D. Crisan and A. Doucet, “A survey of convergence results on particle filtering for practitioners,” *IEEE Trans. Signal Proc.*, vol. 50, pp. 736–746, Mar. 2002.

[6] A. Mutambara, *Decentralized estimation and control for multisensor systems*, CRC Press, Boca Raton, FL, 1998.

[7] B.D. Anderson and J.B. Moore, *Optimal Filtering*, Prentice-Hall, New Jersey, 1979.

[8] D. Alspach and H. Sorenson, “Nonlinear Bayesian estimation using Gaussian sum approximations,” *IEEE Trans. Automatic Control*, vol. 17, pp. 439–448, Aug. 1972.

[9] R.S. Bucy and K.D. Senne, “Digital synthesis of nonlinear filters,” *Automatica*, vol. 7, pp. 287–298, 1971.

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

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

[12] G. Kitagawa, “Monte Carlo filter and smoother for nonlinear non-Gaussian state space models,” *J. Comp. Graph. Statist.*, vol. 5, pp. 1–25, 1996.

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

[14] D. McErlean and S. Narayanan, “Distributed detection and tracking in sensor networks,” in *Proc. Asilomar Conf. Signals, Systems and Computers*, Nov. 2002.

[15] R. R. Brooks, P. Ramanathan, and A.M. Sayeed, “Distributed target classification and tracking in sensor networks,” *Proc. IEEE*, vol. 91, pp. 1163–1171, Aug. 2003.

[16] C. Gui and P. Mohapatra, “Power conservation and quality of surveillance in target tracking sensor networks,” in *Proc. Intl. Conf. MobiCom*, Philadelphia, PA, Sept. 2004.

[17] G. Ing and M.J. Coates, “Parallel particle filters for tracking in wireless sensor networks,” in *Proc. SPAWC*, New York, NY, June 2005.

[18] X. Sheng and Y-H. Hu, “Distributed particle filter with GMM approximation for multiple target localization and tracking in wireless sensor network,” in *Proc. IEEE/ACM Int. Symp. IPSN*, Los Angeles, CA, Apr. 2005.

[19] M. Bolic, P. M. Djuric, and S. Hong, “Resampling algorithms and architectures for distributed particle filters,” to appear, *IEEE Trans. Signal Proc.*, 2005.