DISTRIBUTED AUXILIARY PARTICLE FILTERS USING SELECTIVE GOSSIP

Deniz Üstebay, Mark Coates, and Michael Rabbat

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

ABSTRACT

This paper introduces a distributed auxiliary particle filter for target tracking in sensor networks. Nodes maintain a shared particle filter by coming to a consensus about the likelihoods associated with each particle using the selective gossip procedure. Selective gossip provides a mechanism to efficiently identify the particles with largest weights and focus communication on sharing these important weights. We demonstrate through simulations that the algorithm performs well; compared to state-of-the-art approaches it either significantly improves the accuracy at the expense of a small increase in communication overhead, or achieves comparable accuracy with much lower communication overhead.

***Index Terms*—** Particle filters, gossip algorithms, target tracking, distributed computation

1. INTRODUCTION

Target tracking is an important task for sensor networks and particle filtering becomes an attractive approach when the dynamics are highly non-linear or the noise distribution is non-Gaussian. One approach to implementing a particle filter is the leader-node framework [1], in which one node is appointed as leader (the selected node may change over time). All measurements are relayed to this node so that it can run a centralized particle filter using all data. This has a few disadvantages: only one node can be queried; there is a single point of failure; and the leader node must be aware of the observation models (including calibration parameters and possibly node locations) of all sensors. In addition, since only the leader node has access to an approximation of the posterior, it has to make all sensor management decisions (whether nodes take measurements; which sensing modality they use).

An alternative approach is to distribute the computation. Each node calculates its local likelihood and the information is fused to form a global posterior. Virtually all such distributed filters rely on an assumption of conditional independence of the measurements made at each node. Several of these distributed particle filters require a spanning tree or Hamiltonian cycle for communication [2,3]. Construction and maintenance of such routes can be very challenging when nodes are mobile or wireless conditions are adverse and the algorithms are thus highly vulnerable to link and node failures.

Recently, several more robust approaches have been introduced, which involve an increased communication overhead [4–7]. These algorithms employ gossip (consensus) methods to distribute the information. The algorithm in [4] uses a gossip-based expectation-maximization (EM) algorithm to estimate the parameters of a mixture approximation to the global posterior, but it imposes undesirable constraints on the structure of the likelihood function. In the procedure in [7] each node forms a Gaussian approximation to a (weighted) local posterior and then the gossip algorithm is used to fuse these to construct a Gaussian approximation of the global posterior. This algorithm has a much reduced communication overhead, but its accuracy diminishes when posteriors cannot be adequately captured by the Gaussian approximations.

The algorithms in [5, 6] are more closely related to the one we present here. Rather than forming parametric approximations to the posterior, these algorithms share particles among different nodes. In [5], particles undergo a random walk through the sensor network, and their weights are successively multiplied by a function of the local likelihood. The function is carefully chosen so that the particle weights converge to the same values that a centralized particle filter would calculate. This algorithm has attractive properties, but it only supports importance-sampling from the prior, which can lead to poor performance of a particle filtering algorithm [8]. The algorithm also has no mechanism for eliminating particles with small weights, leading to wasteful communication.

The algorithm in [6] was designed to allow sampling from a better importance distribution (one that better matches the posterior). It estimates regions of concentrated mass in the global posterior by calculating the intersection of the regions of concentration in the local posteriors. The importance sampling function is then constructed to focus on the calculated region, and the gossip procedure is used to calculate the global likelihoods and hence the particle weights. This procedure achieves high accuracy, but the communication cost (in terms of number of values exchanged) is high because the weights of all particles must be calculated, even if many are very small.

In this paper, we present a distributed version of the (two-stage) auxiliary particle filter [9]. In most practical cases, the auxiliary particle filter (APF) significantly outperforms the sequential importance resampling (SIR) filter, especially when the SIR filter uses the prior as the importance function. Our method employs the selective gossip procedure from [10] to calculate both the first- and second-stage weights of the APF. The important aspect of selective gossip is that it can automatically identify the largest values in the vector of particle weights and then only gossips on these values. Our algorithm thus avoids wasteful communication by focusing only on particles with large global weight. We demonstrate through numerical simulations that the algorithm can be significantly more accurate than the distributed particle filter of [7] at the expense of an increased communication overhead. It achieves comparable accuracy to the algorithm in [6] but has significantly less communication overhead.

The remainder of the paper is organized as follows. Section 2 provides a problem statement and Section 3 presents background material on the auxiliary particle filter and selective gossip. Section 4 describes the proposed distributed auxiliary particle filter and Section 5 presents simulation results. Finally, Section 6 makes concluding remarks.

This work was supported by the National Science and Engineering Research Council of Canada, through the Discovery Grants program and the MITACS Networked Centre of Excellence.

2. PROBLEM STATEMENT

We consider a network of geographically distributed (potentially mobile) sensor nodes. Our goal is to perform discrete-time tracking of a state $\mathbf{x}_t$, which may represent a target's position and velocity or a set of environmental conditions (e.g., soil moisture, temperature). We abstract the network as a graph, with a set of vertices $\mathcal{V} = \{1, \ldots, N\}$ and a set of edges $\mathcal{E}_t$, which constitute unordered pairs $u, v \in \mathcal{V}$. The presence of an edge at time t indicates that the two nodes can perform bidirectional wireless communication with high probability. We assume that the network is connected at each time t, and that although nodes are unaware of the global topology, they do have knowledge of their local neighbours.

At time t each node makes a noisy measurement $\mathbf{y}_t^u$ using one of its measurement modalities. The likelihood of the observation is given by the function $p_t(\mathbf{y}_t^u|\mathbf{x}_t)$ where p is time- and sensor-dependent because the choice of sensor(s) may vary. Nodes are unaware of the measurement modalities and observation models employed by other nodes. We assume conditional independence given the state, which implies that the joint likelihood can be factorized:

$$p(\mathbf{y}_t^{\mathcal{V}}|\mathbf{x}_t) = \prod_{u \in \mathcal{V}} p_t(\mathbf{y}_t^u|\mathbf{x}_t). \tag{1}$$

Here $\mathbf{y}_t^{\mathcal{V}} = \{\mathbf{y}_t^u : u \in \mathcal{V}\}$ is the complete set of measurements acquired by the network at time t. The time history of such measurements is denoted $\mathbf{y}_{1:t}^{\mathcal{V}} = \{\mathbf{y}_j^{\mathcal{V}} : 1 \leq j \leq t\}$. We assume that there is coarse-grained synchronization between nodes so that measurements with the same time-index are conducted at approximately the same time (so they are affected by the same state).

3. BACKGROUND

We provide a very limited discussion of particle filtering and focus on the auxiliary particle filter. For a more detailed tutorial with excellent discussion, see [8].

3.1. Auxiliary Particle Filter

The sequential importance resampling (SIR) particle filter strives to maintain a weighted particle approximation $\{\mathbf{x}_{1:t}^{(i)}, w_t^{(i)}\}_{i=1}^K$ to a posterior of interest $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$. Assuming it has a weighted particle approximation at time $t-1$, it does this by propagating the particles to time t by sampling from an importance function q, evaluating the likelihoods of the extended particles, and updating the weights accordingly. Then there is then an optional resampling step to construct a set of particles with more evenly distributed weights. Resampling replicates particles with high weights and discards particles with low weights. In almost all distributed particle filters, the prior is used as the importance function, i.e., $q = p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$. The exception is the algorithm in [6]. Particle filters perform much better if q takes into account information provided by the measurements $\mathbf{y}_t$.

The auxiliary particle filter, introduced in [9], strives to do this by modifying the sampling step. It calculates a *first-stage weight* $\rho_t^{(i)}$ for each particle based on the particle's compatibility with the observation at the next time-step. A good choice for the weight $\rho_t^{(i)}$ is an approximation $\widehat{p}_t(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)})$ to the likelihood $p_t(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}) = \int p_t(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})\, d\mathbf{x}_t$. The APF performs a resampling step according to these weights, prior to conducting the actual importance sampling process. Subsequent to the resampling, the particles are sampled according to an importance function $q(x_t|x_{t-1}^{(i)})$, and the weights associated with the samples are updated. There is then an optional second resampling procedure.

3.2. Distributed Approximation via Selective Gossip

Randomized gossip algorithms are an attractive approach to decentralized computation over wireless networks because they are robust to changing topologies and to unreliable communications. They have simple implementations and they do not have a single point of failure. See [11] for a recent survey. The prototypical version of randomized gossip solves the *average consensus problem*. Let $G = (V, E)$ be a graph representing the communication topology of the network. The graph has an edge $(u, v) \in E$ if and only if nodes u and v communicate directly. Each node initially has a vector, denoted $\gamma^u(0) \in \mathbb{R}^L$, and the goal is to compute the element-wise average, $\bar{\gamma} = |V|^{-1} \sum_{u \in V} \gamma^u(0)$. In randomized gossip, each node u iteratively and asynchronously computes $\bar{\gamma}$ by averaging a local estimate, $\gamma^u(k)$, with one of its neighbors. In particular, at iteration k, a node u is chosen randomly and it contacts a random neighbor v (for which $(u, v) \in E$); then u and v exchange estimates $\gamma^u(k)$ and $\gamma^v(k)$, and compute updates $\gamma^u(k+1) = \gamma^v(k+1) = \frac{1}{2}\big(\gamma^u(k) + \gamma^v(k)\big)$. Standard results for gossip algorithms show that, under mild conditions on the connectivity of G, the estimates converge $\gamma^u(k) \to \bar{\gamma}$ as $k \to \infty$ at every node $u \in V$, and rates of convergence depends on the network topology (see [11] and references therein for details). Note that the version of gossip described above can easily be modified to compute any linear function of the initial vectors by appropriate weighting. In particular, if each node re-weights its initial vector by $|V|$, then randomized gossip will compute the sum rather than the average. The overall communication overhead of randomized gossip depends on the rate of convergence—and thus, the network topology—since two messages are transmitted each iteration, and the size of these messages depends on L, the dimension of the vectors being averaged.

Selective gossip, proposed in [10], is specifically geared toward decentralized approximation of high-dimensional vectors. In particular, selective gossip seeks to approximate $\bar{\gamma}$ at every node by only computing the largest elements of $\bar{\gamma}$. The indices of these elements are not known in advance, since the values of elements of $\bar{\gamma}$ are functions of the initial conditions at all nodes. To save on communication cost, in selective gossip nodes only exchange information and update those elements they believe to be significant. Specifically, each node maintains a threshold, $\tau^u(k)$, which induces a list of significant indices, $\mathcal{I}^u(k) = \{i : [\gamma^u(k)]_i \geq \tau^u(k)\}$. Then, when two nodes u and v gossip, they only exchange information and update the elements in the union $\mathcal{I}^u(k) \cup \mathcal{I}^v(k)$, and the other elements remain fixed. When $|\mathcal{I}^u(k) \cup \mathcal{I}^v(k)| \ll L$, this can result in significant savings in communication resource usage. If an appropriate value for the threshold is known in advance, such that one wishes to only compute elements i for which $[\bar{\gamma}]_i \geq \tau^*$, then the thresholds $\tau^u(0) = \tau^*$ can be initialized and held constant throughout computation. It is more common that one does not know τ^* initially, but instead would only like to compute the largest M out of L elements of $\bar{\gamma}$. In this case, node u adapts $\tau^u(k)$ using stochastic approximation-like updates, as its estimate $\gamma^u(k)$ evolves, so that $\tau^u(k)$ tracks the midpoint between the Mth and $(M+1)$st element of $\gamma^u(k)$.

4. DISTRIBUTED APF ALGORITHM

We strive to construct a distributed APF algorithm such that every node has a copy of the same filter (the same weights and particles).

To do this, the nodes must execute a synchronization routine so that their random number generators have the same seeds; in this way, they always sample the same values.

The challenge in implementing a distributed auxiliary particle filter lies in the fact that the global first-stage weights $\rho_t^{\mathcal{Y},(i)}$ must depend on the measurements $\mathbf{y}_t^{\mathcal{Y}}$, but each node only has access to its local measurement $\mathbf{y}_t^u$. We address this challenge by identifying local first-stage weights $\rho_t^{u,(i)}$ and two (non-linear) functions g and f_u such that the global first-stage weight $\rho_t^{\mathcal{Y},(i)} = g(\frac{1}{N}\sum_{u=1}^N f_u(\rho_t^{u,(i)}))$. With this form, we can employ selective gossip to perform the averaging operation, and to identify and estimate the largest global first-stage weights.

Although it is not an ideal choice because it can lead to unbounded variance in the estimates [8], one choice that satisfies these criteria is the following: $\rho_t^{u,(i)} = p_t(\mathbf{y}_t|\mu_t^{(i)})$, $f_u = N\log(\cdot)$ and $g = \exp(\cdot)$. Here μ_t^i is the mean, mode, or a random draw from $p_t(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$; this was one of the suggested approaches in [9].

Selective gossip identifies a set $\mathcal{I}$ of the particles with the largest global first-stage weights and provides each node with estimates of these weights. We then run a max gossip procedure to ensure that all nodes have exactly the same value (in max gossip, the local update operation is a maximum rather than an average). We then perform the APF resampling step using the particles in $\mathcal{I}$. In the second stage of the algorithm, the nodes need to calculate the global likelihood $p_t(\mathbf{y}_t^{\mathcal{Y}}|\mathbf{x}_t^{(i)})$. This is done in the same fashion — each node calculates $N\log p_t(\mathbf{y}_t^u|\mathbf{x}_t^{(i)})$ and then selective gossip is used to identify the largest average values and perform the averaging for these largest values. Once these likelihoods have been calculated, the weights can be updated and resampling conducted to generate K particles. The complete algorithm is described in Algorithm 1.

5. SIMULATIONS

In this section we investigate the performance of our algorithm through numerical experiments. In our simulation setup we have N sensor nodes distributed uniformly at random on unit square. The communication topology is a random geometric graph, i.e., nodes within radius of $\sqrt{(2\log N/N)}$ are connected. The sensor nodes track a single target for a duration of 50 seconds during which the target makes a full clockwise turn. We model the target motion by a nearly coordinated turn model [12] which assumes that the turn rate is constant and known whereas the cartesian velocity is unknown. The sampling interval is 1 second.

In our simulations, the sensors are capable of measuring four different features: bearing, received signal strength, range and radial velocity. The measurement models are the same as those in [7] — see this paper for more detail. Each measurement includes Gaussian noise with the standard deviations of $0.6981, 8, 0.04, 0.016$, respectively. Each sensor measures one of these features, with the sensing modality chosen randomly.

To compare the performance of our proposed scheme, we implemented a centralized particle filter as well as the distributed Gaussian approximation particle filter scheme of [7]. For the centralized particle filter implementation, we gather all the sensor measurements at one point and calculate the particles centrally. Hence this filter represents the best performance we can hope to achieve with distributed filters. The number of particles for the centralized filter is fixed at 10,000. For the other two schemes, we use 1000 particles at each sensor. The network size is 50 throughout the simulations.

Algorithm 1: Distributed Auxiliary Particle Filter

// Initialization at time $t = 1$
for $u = 1, \ldots, N$ **do**
  for $i = 1, \ldots, K$ **do**
    Sample $\mathbf{x}_1^{(i)} \sim q_1(\cdot)$;
    Set $\gamma_i^u = N\log p_1(\mathbf{y}_1^u|\mathbf{x}_1^{(i)})$;
  endfor
endfor
$[\tilde{\gamma}^u, \mathcal{I}]$ = Selective gossip$(\{\gamma^u\}_{u=1}^N)$;
$\widehat{\gamma}[\mathcal{I}]$ = Max gossip$(\{\tilde{\gamma}^u[\mathcal{I}]\}_{u=1}^N)$;
for $u = 1, \ldots, N$ **do**
  for $i \in \mathcal{I}$ **do**
    Set $\widetilde{w}_1^{(i)} = \frac{\exp(\widehat{\gamma}_i)p(\mathbf{x}_1^{(i)})}{q_1(\mathbf{x}_1^{(i)})}$;
  endfor
  Normalize weights $\widetilde{w}_1^{(i)}$ so that $\sum_{i\in\mathcal{I}}\widetilde{w}_1^{(i)} = 1$;
  Resample $\left\{\mathbf{x}_1^{(i)}, \widetilde{w}_1^{(i)}\right\}_{i\in\mathcal{I}}$ to obtain $\left\{\mathbf{x}_1^{(i)}, \frac{1}{K}\right\}_{i=1}^K$;
endfor

// For times $t > 1$:
for $u = 1, \ldots, N$ **do**
  for $i = 1, \ldots, K$ **do**
    Calculate local first-stage weight $\rho_t^{u,(i)}$;
    Set $\gamma_i^u = f_u(\rho_t^{u,(i)})$;
  endfor
endfor
$[\tilde{\gamma}^u, \mathcal{I}]$ = Selective gossip$(\{\gamma^u\}_{u=1}^N)$;
$\widehat{\gamma}[\mathcal{I}]$ = Max gossip$(\{\tilde{\gamma}^u[\mathcal{I}]\}_{u=1}^N)$;
// Resample
for $u = 1, \ldots, N$ **do**
  for $i \in \mathcal{I}$ **do**
    Set $\rho_t^{(i)} = g(\widehat{\gamma}_i)$;
    Set $w_t^{(i)} = \widetilde{w}_{t-1}^{(i)} \times \rho_t^{(i)}$;
  endfor
  Resample $\left\{\mathbf{x}_{t-1}^{(i)}, w_t^{(i)}\right\}_{i\in I}$ to obtain $\left\{\mathbf{x}_{t-1}^{'(i)}, \frac{1}{K}\right\}_{i=1}^K$;
  for $i = 1, \ldots, K$ **do**
    Set $\mathbf{x}_{1:t-1}^{(i)} = \mathbf{x}_{1:t-1}^{'(i)}$ and $\rho_t^{(i)} = \rho_t^{'(i)}$;
    Sample $\mathbf{x}_t^{(i)} \sim q_t(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$;
    Set $\gamma_i^u = N\log p_t(\mathbf{y}_t^u|\mathbf{x}_t^{(i)})$;
  endfor
endfor
$[\tilde{\gamma}^u, \mathcal{I}]$ = Selective gossip$(\{\gamma^u\}_{u=1}^N)$;
$\widehat{\gamma}[\mathcal{I}]$ = Max gossip$(\{\tilde{\gamma}^u[\mathcal{I}]\}_{u=1}^N)$;
for $u = 1, \ldots, N$ **do**
  for $i \in \mathcal{I}$ **do**
    Set $\widetilde{w}_t^{(i)} = \frac{\exp(\widehat{\gamma}_i)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{\rho_t^{(i)}q(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}$;
  endfor
  Resample $\left\{\mathbf{x}_{1:t}^{(i)}, \widetilde{w}_t^{(i)}\right\}_{i\in\mathcal{I}}$ to obtain $\left\{\mathbf{x}_{1:t}^{(i)}, \frac{1}{K}\right\}_{i=1}^K$;
endfor

Here we provide the results of 1000 Monte-Carlo trials. For each trial we use a random node distribution on the unit square, hence the network connectivity is different each time. In addition, the sequence of gossiping pairs of nodes changes for each trial. Figure 1

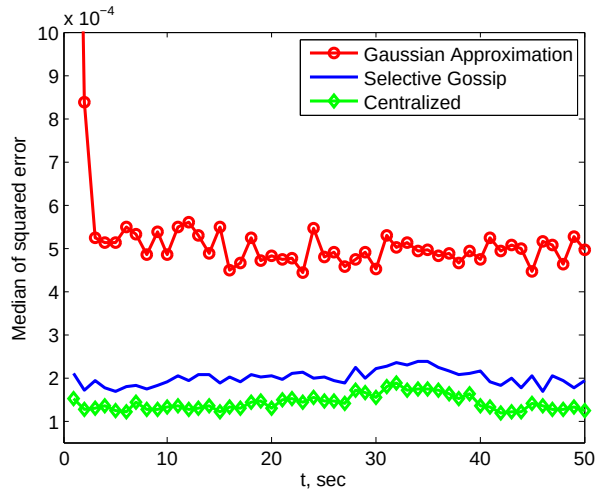

Fig. 1. Position error performance comparison for the centralized filter, proposed filter and the filter of Oreshkin et. al. [7].

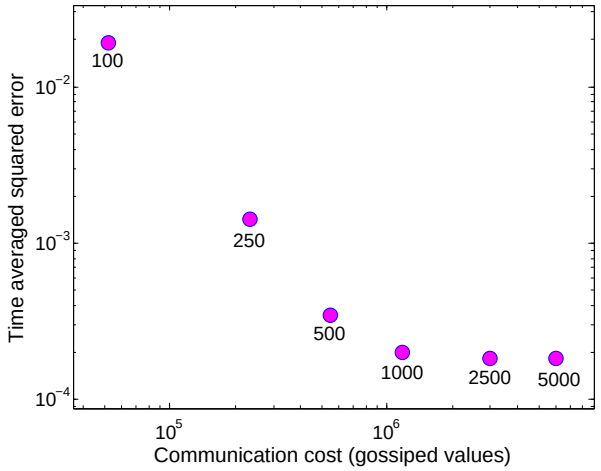

Fig. 2. Position error - communication cost trade off for the proposed particle filter. The numbers below the markers indicate the number of selective gossip iterations.

illustrates the median of squared position error performances over the Monte-Carlo trials. For the selective gossip approach, we gossip over only the 50 largest terms. We observe that, for this example problem, our proposed selective gossip based approach gives an error performance close to the centralized filter.

Next we investigate the communication cost of our algorithm. Figure 2 shows the position error versus associated communication overhead. The communication overhead is calculated in terms of gossiped values. Numbers on the plot indicate the number of gossip iterations. Hence this plot illustrates how error and required communication overhead varies as we change the number of selective gossip iterations. In [7] the communication overhead is given as $50 \times N^2$, hence it is 1.25×10^5 for this setup. We conclude that selective gossip approach achieves substantial improvement in tracking performance compared to the Gaussian approximation approach. The trade off is the increased communication cost.

6. CONCLUSIONS

We have presented a distributed implementation of the auxiliary particle filter. In this algorithm, nodes maintain a common particle filter. Selective gossip is used to efficiently calculate the first-stage and second-stage weights of the filter. Numerical simulations indicate that the algorithm provides advantages to other state-of-the-art approaches; either it provides a substantial improvement in accuracy at the expense of a small increase in communication overhead or it provides comparable accuracy with much less communication overhead. In future work, we will explore strategies for identifying better first-stage weights and mechanisms that allow selective gossip to automatically determine how many terms should be included in $\mathcal{I}$.

7. REFERENCES

[1] F. Zhao, J. Liu, J. Liu, L. Guibas, and J. Reich, "Collaborative signal and information processing: an information-directed approach," *Proc. IEEE*, vol. 91, no. 8, pp. 1199–1209, Aug. 2003.

[2] M. Coates, "Distributed particle filters for sensor networks," in *Proc. Int. Symposium Info. Process. Sens. Netw. (IPSN)*, Berkeley, CA, USA, Apr. 2004.

[3] X. Sheng, Y.-H. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network," in *Proc. Int. Symp. Info. Process. Sens. Netw. (IPSN)*, Los Angeles, CA, USA, Apr. 2005.

[4] D. Gu, "Distributed particle filter for target tracking," in *Proc. IEEE Int. Conf. Robot. and Automation*, Rome, Italy, Apr. 2007.

[5] S.H. Lee and M. West, "Markov Chain Distributed Particle Filters (MCDPF)," in *Proc. IEEE Conf. Decision and Control*, Shanghai, China, Dec. 2009.

[6] S. Farahmand, S.I. Roumeliotis, and G.B. Giannakis, "Particle filter adaptation for distributed sensors via set membership," in *Proc. IEEE. Int. Conf. Acoustics Speech and Sig. Proc. (ICASSP)*, Dallas, TX, USA, Mar. 2010.

[7] B.N. Oreshkin and M.J. Coates, "Asynchronous distributed particle filter via decentralized evaluation of Gaussian products," in *Proc. ISIF Int. Conf. Information Fusion*, Edinburgh, UK, July 2010.

[8] A. Doucet and M. Johansen, *Oxford Handbook of Nonlinear Filtering*, chapter A tutorial on particle filtering and smoothing: fifteen years later, Oxford University Press, 2010, to appear.

[9] M.K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, Jun. 1999.

[10] D. Üstebay, R. Castro, and M. Rabbat, "Selective gossip," in *Proc. IEEE Comp. Advances in Multi-Sensor Adaptive Processing*, Aruba, Dec. 2009.

[11] A.G. Dimakis, S. Kar, J.M.F. Moura, M.G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, Nov. 2010.

[12] Y. Bar-Shalom, X. Rong Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation*, Athena Scientific, Belmont, MA, 2001.