We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



25

Monte Carlo Methods for Node Self-Localization and Nonlinear Target Tracking in Wireless Sensor Networks

Joaquín Míguez, Luis Arnaiz and Antonio Artés-Rodríguez
Department of Signal Theory and Communications
Universidad Carlos III de Madrid
Spain

1. Introduction

Most applications of wireless sensor networks (WSN) rely on the accurate localization of the network nodes [Patwari et al., 2005]. In particular, for network-based navigation and tracking applications it is usually assumed that the sensors, and possibly any data fusion centers (DFCs) in charge of processing the data collected by the network, are placed at *a priori* known locations. Alternatively, when the number of nodes is too large, WSNs are usually equipped with beacons that can be used as a reference to locate the remaining nodes [Sun et al., 2005]. In both scenarios, the accuracy of node localization depends on some external system that must provide the position of either the whole set of nodes or, at least, the beacons [Patwari et al., 2005]. Although beacon-free network designs are feasible [Sun et al., 2005, Ihler et al., 2005, Fang et al., 2005, Vemula et al., 2006], they usually involve complicated and energy-consuming local communications among nodes which should, ideally, be very simple.

In this paper, we address the problem of tracking a maneuvering target that moves along a region monitored by a WSN whose nodes, including both the sensors and the DFCs, are located at unknown positions. Therefore, the target trajectory, its velocity and all node locations must be estimated jointly, without assuming the availability of beacons. We advocate an approach that consists of three stages: initialization of the WSN, target and node tracking, and data fusion. At initialization, the network collects a set of data related to the distances among nodes. These data can be obtained in a number of ways, but here we assume that each sensor node is able to detect, with a certain probability of error, other nodes located nearby and transmit this information to the DFCs. These data are then used by the DFCs to acquire initial estimates of the node positions. An effective tool to perform this computation is the accelerated random search (ARS) method of [Appel et al., 2003], possibly complemented with an iterated importances sampling procedure [Cappé et al., 2004] to produce a random population of node positions approximately distributed according to their postrior probability distribution given the available data. This approach is appealing because it couples naturally with the algorithms in the tracking phase.

We propose to carry out target tracking by means of sequential Monte Carlo (SMC) methods, also known as particle filters (PFs) [Doucet et al., 2000, 2001, Crisan & Doucet,

Open Access Database www.intechweb.org

Source: Sensor and Data Fusion, Book edited by: Dr. ir. Nada Milisavljević,
ISBN 978-3-902613-52-3, pp. 490, February 2009, I-Tech, Vienna, Austria

2002, Djurić et al., 2003, Ristić et al., 2004, Bolić et al., 2005], which recursively track the target position and velocity, as well as improve node positioning, with the generation of new data by the WSN. We should remark that the treatment of unknown (random) fixed parameters, such as the sensor positions are in our framework, using PFs is still an open problem. We propose two algorithms that tackle this dificulty. The first one is based on the auxiliary particle filtering (APF) methodology [Pitt & Shephard, 2001, Liu & West, 2001] and the second one follows the density-assisted strategy of [Djurić et al., 2004].

The data fusion stage deals with the combination of the outputs produced by different DFCs in order to produce improved estimates of both the target state and the WSN node locations. Again, we investigate two approaches. Both of them are aimed at the coherent combination of the estimates produced by individual DFCs but differ in the amount of information that they use and the requirements imposed on the WSN communication capabilities. The most complex technique theoretically yields asymptotically optimal Bayesian estimation of the target state and the node locations (hence, optimal fusion) with distributed computations, but at the expense of making *all* data available to *all* DFCs. It is based on the parallelization of PFs as addressed in [Bolić et al., 2005, Míguez, 2007].

The remaining of the paper is organized as follows. After a brief comment on notation, Section 2 is devoted to a formal description of the system moel. A general outline of the proposed scheme is given in Section 3. Sections 4, 5 and 6 are devoted to the procedures proposed for initialization, tracking and data fusion, respectively. In Section 7 we show some illustrative simulation results. Finally, the main results are summarized in Section 8.

1.1 Notation

Scalar magnitudes are denoted as regular letters, e.g., x, N. Vectors and matrices are denoted as lower-case and upper-case bold-face letters, respectively, e.g., vector x and matrix $\mathbf{X}$. We use $p(\cdot)$ to denote the probability density function (pdf) of a random magnitude. This is an argument-wise notation, i.e., $p(x)$ denotes the pdf of x and $p(y)$ is the pdf of y, possibly different. The conditional pdf of x given the observation of y is written as $p(x \mid y)$. Sets are denoted using calligraphic letters, e.g., $\mathcal{Q}$. Specific sets built from sequences of elements are denoted by appropriate subscripts, e.g., $x_{1:N} = \{x_1, \ldots, x_N\}$.

2. System model

We assume that the target moves along a 2-dimensional region $\mathcal{C} \subseteq \mathbb{C}^2$ (a compact subset of the complex plane) according to the linear model Gustafsson02

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{u}_t, \quad t \in \mathbb{N} \tag{1}$$

where $\mathrm{x}_t = [r_t, v_t]^{\mathrm{T}} \in \mathbb{C}^2$ is the target state, which includes its position and its velocity at time t (r_t and v_t, respectively); $\mathbf{A} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$ is a transition matrix that depends on the observation period, T, and

$$\mathbf{u}_t = [u_{r,t}, u_{v,t}]^{\mathrm{T}} \sim CN(\mathbf{u}_t \mid 0, \mathbf{C}_u) \tag{2}$$

is a complex Gaussian noise term, with zero mean and known covariance matrix

$$\mathbf{C}_u = \sigma_u^2 \begin{bmatrix} \frac{1}{4}T^4 & 0 \\ 0 & T^2 \end{bmatrix}, \tag{3}$$

that accounts for unknown acceleration forces. The initial target state, x_0, has a known prior probability density function (pdf), $p(\mathbf{x}_0) = p(r_0)p(v_0)$, and we assume $p(v_0) = CN(0, \sigma_{v,0}^2)$, i.e., the prior pdf of the velocity random process is complex Gaussian with zero mean and variance, $\sigma_{v,0}^2$.

The network consists of N_s sensors and N_c DFCs. Sensors are located at random unknown positions $s_{1:N_s} = \{s_1, s_2, \ldots, s_{N_s}\}$, $s_i \in \mathbb{C}$, with independent and identical uniform prior pdf's, $p(s_i) = U(\mathcal{C})$, $i = 1, \ldots, N_s$, on the 2-dimensional region monitored by the WSN. During the network startup, each sensor detects any other nodes located within a certain range, $S_u > 0$. In particular, the *n*-th sensor builds up an $N_s \times 1$ vector of decisions, $\mathbf{b}_n = [b_{n,1}, \ldots, b_{n,N_s}]^{\mathrm{T}}$, where (deterministically) $b_{n,n} = 1$ while $b_{n,k} \in \{1,0\}$, $n \neq k$, is a binary random variable with probability mass function (pmf)

$$p(b_{n,k} = 1 \mid s_{1:N_s}) = p_d(d_{n,k}^s, S_u), \tag{4}$$

where $d_{n,k}^s = \mid s_n - s_k \mid$ is the distance between the *n*-th and *k*-th sensors and $p_d(\cdot,\cdot)$ is the function that yields the probability of detection. At time 0, these decisions are broadcast to the DFCs and we collect them altogether in the $N_s \times N_s$ matrix $\mathbf{B} = [\mathrm{b}_1, \ldots, \mathrm{b}_{N_s}]$ for notational convenience.

The locations of the N_c DFCs are denoted as $c_1, \ldots, c_{N_c}$, with $c_i \in \mathcal{C} \ \ \forall i$. By convention, the first DFC is assumed to be located at the origin of the monitored region, i.e., $c_1 = 0$. The positions of the remaining DFCs are assumed random and unknown, with complex Gaussian prior pdf's $p(c_i) = CN(c_i \mid \mu_i^c, \sigma_c^2)$, $i = 2, \ldots, N_c$. The physical implication of this model is that DFCs are deployed at locations which are only roughly known. The variance σ_c^2 indicates the uncertainty in this prior knowledge.

During the normal operation of the network, the *n*-th sensor periodically measures some distance-dependent physical magnitude related to the target. The measurement obtained by the n -th sensor at discrete-time $t \geq 1$ is denoted as $y_{n,t} = f_s(d_{n,t}, \varepsilon_{n,t}^y)$, where $d_{n,t} = \mid r_t - s_n \mid$ is the distance between the target and the sensor, $\varepsilon_{n,t}^y$ is a random perturbation with known pdf and $f_s(\cdot,\cdot)$ is the measurement function. We assume that not every sensor necessarily transmits its observation, $y_{n,t}$, at every time. Indeed, it is often convenient (in order to reduce energy consumption) that only a subset of sensors become active and transmit their measurements. The local decision of a sensor to transmit its data or

not depends on the comparison of the measurement, $y_{n,t}$, with some reference value, S_y. We also introduce a certain probability of transmission failure, β. A failure can be caused, e.g., by a strong interference in the channel that prevents adequate reception of the communication signal at the DFC. Thus, at time t only an $N_t \times 1$ vector of observations, $\mathbf{y}_t = [y_{\kappa(1),t}, \ldots, y_{\kappa(N_t),t}]^{\mathrm{T}}$, where $0 \le N_t \le N_s$ and $\kappa(i) \in \{1, \ldots, N_s\}$, $\forall i$, is effectively broadcast to the DFCs (note that all DFCs collect the same data from the sensors). We assume that the likelihood $p(\mathbf{y}_t \mid r_t, s_{1:N_s}, c_{1:N_c})$ can be evaluated up to a proportionality constant.

Each DFC has the capability to extract some distance-related magnitude from the communication signals transmitted by the sensors. For simplicity, we consider the same type of measurement carried out at the sensors, hence the n -th DFC also has available, at time $t \ge 0$, the $N_t \times 1$ data vector $\mathbf{z}_{n,t} = [z_{\kappa(1),n,t}, \ldots, z_{\kappa(N_t),n,t}]^{\mathrm{T}}$, where $\mathbf{z}_{i,n,t} = f_s(d^c_{i,n,t}, \varepsilon^z_{i,n,t})$, $d^c_{i,n,t} = \| s_i - c_n \|$ and $\varepsilon^z_{i,n,t}$ is a random perturbation with known pdf, so that the likelihood $p(z_{n,t} \mid s_{1:N_s}, c_n)$ can be computed. Note that $\mathbf{z}_{n,0}$ is defined (unlike y_0), and has dimension $N_0 = N_s$, because during the network startup all sensors broadcast signals to the DFCs.

We assume that the DFCs are equipped with communication devices more sophisticated than those at the sensor nodes and, as a consequence, it is feasible to exchange data among the DFCs. In particular, during network startup one DFC collects a set of $N_c(N_c - 1)$ observations, $q_{i,n} = f_s(d^0_{i,n}, \varepsilon^0_{i,n})$, $i, n \in \{1, \ldots, N_c\}$ (but $i \ne n$), where $d^0_{i,n} = \| c_i - c_n \|$ and $\varepsilon^0_{i,n}$ is a random perturbation with known pdf, so that $p(q_{i,n} \mid c_{1:N_c})$ can be evaluated. For conciseness, we define the set $\mathcal{Q} = \{q_{i,n}\}^{i \ne n}_{i,n \in \{1,\ldots,N_c\}}$. Moreover, during normal operation of the WSN, each DFC may receive sufficient information from the other fusion nodes to build the $N_t \times N_c$ matrix of observations $\mathbf{Z}_t = \lfloor \mathbf{z}_{1,t}, \ldots, \mathbf{z}_{N_c,t} \rfloor$. Essentially, this means that the DFCs may be capable of sharing data.

The goal is to jointly estimate the target states $\mathbf{x}_{0:t} = \{\mathbf{x}_0, \ldots, \mathbf{x}_t\}$, the sensor locations, $s_{1:N_s}$ and the uknown DFC positions, $c_{2:N_c}$, from the decisions in B, the data in $\mathcal{Q}$ and the sequences of observation vectors $\mathbf{y}_{1:t} = \{\mathbf{y}_1, \ldots, \mathbf{y}_t\}$ and $\mathbf{Z}_{0:t} = \{\mathbf{z}_0, \ldots, \mathbf{z}_t\}$.

3. Proposed scheme

The proposed method consists of three stages, that we outline below.

Stage 1. Initialization: Using the data generated during the network startup, at this stage we compute maximum *a posteriori* (MAP) estimates of the DFC locations, $c_{2:N_c}$, that will be kept fixed during the WSN operation (including the tracking and fusion stages). We also compute marginally MAP estimates of the sensor locations, $s_{1:N_s}$. The latter point estimates are employed to initializae an iterated importance sampling procedure that generates a

population of candidate positions at time $t=0$, denoted $s_{0,1:N_s}^{(i)}$, $i=1,...,M$, approximately distributed according to the posterior pdf $p(s_{1:N_s} | \mathcal{Q}, \mathbf{B}, \mathbf{Z}_0, c_{1:N_c})$. This posterior sample is needed to start the tracking algorithms in stage 2.

Stage 2. Tracking: We investigate two techniques. The first one is an APF algorithm for state estimation in dynamic systems with unknown fixed parameters derived according to [Liu & West, 2001]. A special feature of this technique is that it adaptively approximates the high dimensional posterior pdf of the sensor positions using a sequence of kernel mixtures.

The second algorithm relies on the density-assisted (DA) approach of [Djurić et al., 2004] to approximate the posterior density of the sensor positions using a parametric familiy of pdf's. An advantage of this procedure is that it enables the analytical integration of the target velocity, thus reducing the sampling space dimension [Chen & Liu, 2000].

Stage 3. Fusion of estimates: Both the APF and the DA tracking algorithms can either be run independently in separate DFCs (an using only the data available at each DFC) or be designed as a single global algorithm (that needs to use all the data available through all DFCs) implemented in a distributed way. The computational complexity is similar in both cases but the second approach imposes stringent communication requirements on the network.

If the tracking algorithms are run independently, using different observation sets at each DFC, we propose a mechanism for fusing the resulting N_c estimates that combines them *coherently* into a single point estimate. We must note that estimates produced by different DFCs with different data cannot by simply averaged because the available observations (both $\mathbf{y}_{1:t}$ and $\mathbf{Z}_{0:t}$) are insensitive to the angles between the target and the sensors or the sensors and the DFCs. Therefore, it is necessary to take one track estimate as a reference (e.g., the one produced by DFC 1) and adjust the others *by means of rotations only* to minimize the mismatch.

If the tracking algorithms use the same set of observations in all DFCs, then it is possible to apply methods por the implementation of a single PF whose computations are distributed (parallelized) among the available DFCs. This is done by applying the techniques in [Bolić et al., 2005] and [Míguez, 2007]. With this approach each DFC transmits its $\mathbf{z}_{n,t}$, $n \in \{1,..., N_c\}$, to all other DFCs but, in exchange, it theoretically guarantees an asymptotically optimal Bayesian estimation of the target state and sensor positions by a simple linear combination of the estimates produced by the N_c DFCs.

4. Initialization

4.1 Point estimation

As a first step, we obtain point estimates of the node locations, including the DFCs, $c_{2:N_c}$ (note that we assume c_1=0), and the sensors, $s_{1:N_s}$, given the information available at time t=0. An accurate estimation of the DFC positions is of utmost importance, since they will be kept fixed in subsequent stages. Therefore, we propose to compute MAP estimates of the locations $c_{2:N_c}$ by solving the nonlinear optimization problem

$$\hat{c}_{2:N_c} = \arg\max_{c_{2:N_c}} \{ p(c_{2:N_c} | \mathcal{Q}, \mathbf{Z}_0) \}$$

$$= \arg\max_{c_{2:N_c}} \left\{ \prod_{i \neq k} p(q_{i,k} \mid c_i, c_k) \prod_{j=2}^{N_c} CN(c_j \mid \mu_j^c, \sigma_c^2) \right\}. \quad (5)$$

Similarly, marginal MAP estimates of the sensor locations, conditional on $c_{2:N_c} = \hat{c}_{2:N_c}$, is achieved by solving

$$\hat{s}_\ell = \arg\max_{s_\ell} \left\{ p(s_\ell \mid Z_0, \hat{c}_{2:N_c}) \right\}$$

$$= \arg\max_{s_\ell} \left\{ \prod_{n=1}^{N_c} p(z_{\ell,n,0} \mid s_\ell, \hat{c}_n) \right\}, \quad (6)$$

for $\ell = 1, \ldots, N_s$ and $\hat{c}_1 = c_1 = 0$ is *a priori* known. We point out that addressing the estimation of each sensor individually we neglect the information in matrix **B**, but the dimensionality reduction accounts for this loss (searching for a global MAP solution in $\mathcal{C}^{N_s}$ turns out practically much harder).

Problems (5) and (6) do not have closed form solutions in general. However, they can be numerically solved, with high accuracy, using the accelerated random search (ARS) algorithm [Appel et al., 2003]. ARS is a Monte Carlo technique for global optimization that enjoys a fast convergence rate and a simple implementation. The algorithm is described, for a general setup, in Table 1.

Problem: $\hat{\alpha} = \arg\max_{\alpha \in \mathcal{A}} g(\alpha)$ for some function g.
Denote: $R_{min} > 0$, the "minimum radius"; $R_{max} > R_{min}$, the "maximum radius"; $R_{max} \geq R_n \geq R_{min}$ the radius at the n-th iteration; $\nu > 1$ the "contraction" factor; α_n the solution obtained after the n-th iteration; and
$\mathcal{B}_n = \{\tilde{\alpha} \in \mathcal{A} : ||\tilde{\alpha} - \alpha_n||_2 < R_n\}$,
where $||\cdot||_2$ indicates the Euclidean norm.
Algorithm: given R_n and α_n,
(1) Draw $\tilde{\alpha} \sim U(\mathcal{B}_n)$.
(2) If $g(\tilde{\alpha}) > g(\alpha_n)$ then $\alpha_{n+1} = \tilde{\alpha}$ and $R_{n+1} = R_{max}$, else $\alpha_{n+1} = \alpha_n$ and $R_{n+1} = R_n/\nu$.
(3) If $R_{n+1} < R_{min}$, then $R_{n+1} = R_{max}$.
(4) Go back to (1)

Table 1. Iterative ARS algorithm for a maximization problem. Parameter α is possibly multidimensional (typically, $\alpha \in \mathbb{C}^n$). The algorithm is usually stopped after a given number of iterations without going changing α_n.

We assume the DFC positions known for all subsequent derivations, i.e., we treat $\hat{c}_{2:N_c}$ as the true values and skip them from notation for conciseness.

4.2 Population Monte Carlo

The PFs applied in the tracking stage need a sample of sensor positions drawn from the posterior pdf at time *t*=0, $p(s_{1:N_s} \mid \mathbf{B}, \mathbf{Z}_0)$ in order to start running. To generate this initial population, we propose an iterated importance sampling[1] method called PMC [Cappé et al., 2004]. In the first iteration, particles are drawn from independent complex Gaussian proposals built from the ARS estimates and a fixed variance, $\sigma_s^2(0)$, i.e.,

$$s_n^{(i)}(0) \sim CN(s_n \mid \hat{s}_n, \sigma_s^2(0)), \quad n = 1, \ldots, N_s, \quad i = 1, \ldots, M, \tag{7}$$

with weights

$$w^{(i)}(0) = \frac{p(\mathbf{Z}_0 \mid s_{1:N_s}^{(i)}(0)) p(\mathbf{B} \mid s_{1:N_s}^{(i)}(0))}{\prod_{n=1}^{N_s} CN(s_n^{(i)}(0) \mid \hat{s}_n, \sigma_s^2(0))}. \tag{8}$$

After the (k-1)-th iteration, the weighted particles are $\Omega_{k-1}^{pmc} = \{s_{1:N_s}(k-1)^{(i)}, w^{(i)}(k-1)\}_{i=1}^{M}$ and importance sampling for the *k*-th iteration is performed as

$$s_n^{(i)}(k) \sim CN(s_n \mid \bar{s}_n^{(i)}(k-1), \sigma_{s,n}^2(k-1)), \tag{9}$$

where

$$\bar{s}_n(k-1) = \sum_{i=1}^{M} w^{(i)}(k-1) s_n^{(i)}(k-1),$$

$$\bar{s}_n^{(i)}(k-1) = a s_n^{(i)}(k-1) + (1-a)\bar{s}_n(k-1), \quad \textit{and}$$

$$\sigma_{s,n}^2(k-1) = (1-a^2) \sum_{i=1}^{M} w^{(i)}(k-1) \mid s_n^{(i)}(k-1) - \bar{s}_n(k-1) \mid^2$$

for some 0<*a*<1, i.e., we build the (*k*-1) -th kernel approximation of $p(s_{1:N_s} \mid \mathbf{Z}_0, \mathbf{B})$ with shrinkage [Liu & West, 2001] for variance reduction. The corresponding weights are

$$w^{(i)}(k) = \frac{p(\mathbf{Z}_0 \mid s_{1:N_s}^{(i)}(k)) p(\mathbf{B} \mid s_{1:N_s}^{(i)}(k))}{\prod_{n=1}^{N_s} CN(s_n^{(i)}(k) \mid \bar{s}_n^{(i)}(k-1), \sigma_{s,n}^2(k-1))}. \tag{10}$$

If the algorithm is iterated *N* times, we obtain a sample of equally-weighted particles $\{s_{0,1:N_s}^{(i)}\}_{i=1}^{M}$, with approximate pdf $p(s_{1:N_s} \mid \mathbf{Z}_0, \mathbf{B})$ by resampling from Ω_N^{pmc}.

[1]See, e.g., [DeGroot & Schervish, 2002] for a brief review of the importance sampling principle.

5. Tracking

In the second stage, the aim is to track the sequence of target states, $\mathbf{x}_t$, and improve the estimation of the sensor positions, $s_{1:N_s}$, recursively, as new observations $\mathbf{y}_t$ and $\mathbf{Z}_t$ are received. We introduce two PFs that attain this goal. We assume that each DFC runs a PF, but the algorithms are derived for the scenario where all data at time t, including both $\mathbf{y}_t$ and $\mathbf{Z}_t = [\mathbf{z}_{1,t}, \ldots, \mathbf{z}_{N_c,t}]$, are available for the algorithm. This means that $\mathbf{z}_{n,t}$ must be transmitted from DFC n, where it is collected, to all other DFCs. Note, however, that we can derive the proposed algorithms in the alternative scenario in which only $\mathbf{z}_{n,t}$ is available at the n -th DFC by simply substituting $\mathbf{Z}_{0:t}$ by $\mathbf{z}_{n,0:t}$ in the proposed procedures. Indeed, we will assess the performance of the PFs in the two scenarios in Section 7.

5.1 Auxiliary particle filter

As a first approach, we propose to use an APF algorithm based on [Liu & West, 2001]. The APF is a recursive algorithm that generates a sequence of discrete probability measures, denoted $\Omega_t = \{(\mathbf{x}_t, s_{t,1:N_s}, w_t^{(i)}\}_{i=1}^M$, that approximate the posterior pdf's of the unknowns, i.e., for $n \in \{1, \ldots, N_c\}$,

$$p(\mathbf{x}_t, s_{1:N_s} \mid \mathbf{y}_{1:t}, \mathbf{Z}_{0:t}, \mathbf{B}) \approx \sum_{i=1}^{M} \delta_i(\mathbf{x}_t, s_{1:N_s}) w_t^{(i)}, \tag{11}$$

where $\delta_i(\mathbf{x}_t, s_{1:N_s})$ is a delta measure centered at $\{\mathbf{x}_t^{(i)}, s_{t,1:N_s}^{(i)}\}$. The samples, $\mathbf{x}_t^{(i)}$ and $s_{t,1:N_s}^{(i)}$, for $i = 1,...,M$, are called particles and the importance weights $w_t^{(i)}$ are normalized to yield $\sum_{i=1}^{M} w_t^{(i)} = 1$. When at time t observations $\mathbf{y}_t$ and $\mathbf{Z}_t$ become available, Ω_t is recursively computed from Ω_{t-1} as described in Table 2.

The proposed APF algorithm is based on the relationship

$$\begin{aligned} p(\mathbf{x}_t, s_{1:N_s} \mid \mathbf{y}_{1:t}, \mathbf{Z}_{0:t}, \mathbf{B}) \propto\ & p(\mathbf{y}_t, \mathbf{Z}_t \mid \mathbf{x}_t, s_{1:N_s}) p(\mathbf{x}_t \mid s_{1:N_s}, \mathbf{y}_{1:t-1}) \\ & \times p(s_{1:N_s} \mid \mathbf{y}_{1:t-1}, \mathbf{Z}_{0:t-1}, \mathbf{B}) \end{aligned} \tag{12}$$

and the approximations

$$p_M(\mathbf{x}_t \mid s_{1:N_s}, \mathbf{y}_{1:t-1}) = \sum_{i=1}^{M} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}^{(i)}) \delta_i(s_{1:N_s}) w_{t-1}^{(i)} \tag{13}$$

$$p_M(s_{1:N_s} \mid \mathbf{y}_{1:t-1}, \mathbf{Z}_{0:t-1}, \mathbf{B}) = \sum_{i=1}^{M} w_{t-1}^{(i)} K_i(s_{1:N_s}), \tag{14}$$

where $K_i(\cdot)$ is a symmetric kernel. For the latter, we have chosen

$$K_i(s_{1:N_s}) = CN(s_{1:N_s} \mid \boldsymbol{\mu}_{t-1}^{(i)}, h^2\Sigma_{t-1}) \tag{15}$$

where

$$\boldsymbol{\mu}_{t-1}^{(i)} = [\bar{s}_{t-1,1}^{(i)}, \ldots, \bar{s}_{t-1,N_s}^{(i)}]^{\mathrm{T}} \tag{16}$$

$$\boldsymbol{\Sigma}_{t-1} = diag\{\sigma_{t-1,1}^2, \ldots, \sigma_{t-1,N_s}^2\} \tag{17}$$

and $h \in (0,1)$ is a bandwidth factor. The kernel modes are calculated as

$$\bar{s}_{t-1,k}^{(i)} = a s_{t-1,k}^{(i)} + (1-a)\bar{s}_{t-1,k}, \tag{18}$$

for $a = \sqrt{1-h^2}$ and

$$\bar{s}_{t-1,k} = \sum_{k=1}^{M} w_{t-1}^{(i)} s_{t-1,k}^{(i)}. \tag{19}$$

$t = 0$: Draw $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$ and $s_{0,1:N_s}^{(i)} \sim p(s_{1:N_s}|\mathbf{B}, \mathbf{Z}_0)$

t: Given $\Omega_{t-1} = \left\{(\mathbf{x}_{t-1}, s_{t-1,1:N_s})^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{M}$:

(1) Compute $\tilde{\mathbf{x}}_t^{(i)} = \mathbf{A}\mathbf{x}_{t-1}^{(i)}$, $i = 1, ..., M$.

(2) Draw indices $\ell^{(i)} \sim q_t(\ell)$, $i = 1, ..., M$, where $q_t(\ell) \propto w_{t-1}^{(\ell)} p(\mathbf{y}_t, \mathbf{Z}_t|\tilde{\mathbf{x}}_t^{(\ell)}, s_{t-1,1:Ns}^{(\ell)})$.

(3) Draw $s_{t,1:N_s}^{(i)} \sim q_t(s_{1:N_s}|\ell^{(i)})$, where $q_t(s_{1:N_s}|\ell^{(i)}) = N(s_{1:N_s}|\boldsymbol{\mu}_{t-1}^{(\ell^{(i)})}, h^2\boldsymbol{\Sigma}_{t-1})$, for $i = 1, ..., M$.

(4) Draw target states $\mathbf{x}_t^{(i)} \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(\ell^{(i)})})$, $i = 1, ...M$, and build the trajectory $\mathbf{x}_{0:t}^{(i)} = \{\mathbf{x}_{0:t-1}^{(\ell^{(i)})}, \mathbf{x}_t^{(i)}\}$.

(5) Update the weights, for $i = 1, ..., M$, $w_t^{(i)} \propto \frac{p\left(\mathbf{y}_t, \mathbf{Z}_t|\mathbf{x}_t^{(i)}, s_{t,1:N_s}^{(i)}\right)}{p\left(\mathbf{y}_t, \mathbf{Z}_t|(\tilde{\mathbf{x}}_t, s_{t-1,1:N_s})^{(\ell^{(i)})}\right)}$.

(6) MMSE estimation: $\hat{\mathbf{x}}_t = \sum_{i=1}^{M} \mathbf{x}_t^{(i)} w_t^{(i)}$ and $\hat{s}_{1:N_s} = \sum_{i=1}^{M} s_{1:N_s}^{(i)} w_t^{(i)}$.

Table 2. Liu and West's APF algorithm for joint estimation of the target trajectory, $x_{0:t}$, and the fixed node locations, $s_{1:N_s}$, from the observations available at time t.

The variances, in turn, are found as

$$\sigma_{k,t-1}^2 = \sum_{l=1}^{M} w_{t-1}^{(l)} \left(s_{t-1,k}^{(l)} - \bar{s}_{t-1,k}\right)^2. \tag{20}$$

This choice of $\boldsymbol{\mu}_{t-1}^{(i)}$ and $\boldsymbol{\Sigma}_{t-1}$ ensures that the mean and marginal variance of every fixed parameter given by the kernel approximation (14) is equal to the corresponding mean and marginal variance given by the weights [Liu & West, 2001]..
One difficulty with the approximations (13) and (14) is that they involve mixtures of a typically large number (*M*) of pdf's. We avoid this limitation by incorporating a discrete auxiliary random variable $\ell \in \{1,...,M\}$ that indicates the terms in (13) and (14) to be selected. In particular, we define

$$p(x_t, s_{1:N_s}, \ell \mid y_{1:t}, Z_{0:t}, B) \propto p(y_t, Z_t \mid x_t, s_{1:N_s}) p(x_t \mid x_{t-1}^{(\ell)}) w_{t-1}^{(\ell)} K_\ell(s_{1:N_s}). \tag{21}$$

Using (21) we can easily draw particles and compute weights by applying the principle of importance sampling (IS) [DeGroot & Schervish, 2002].. In particular, we define a suitable importance function, or proposal pdf,

$$q_t(x_t, s_{1:N_s}, \ell) = q_t(\ell) q_t(s_{1:N_s} \mid \ell) p(x_t \mid x_{t-1}^{(\ell)}) \tag{22}$$

(see Table 2 for the details) that we use for drawing new particles and then update the weights as

$$w_t^{(i)} \propto \frac{p((\mathbf{x}_t, s_{t,1:N_s}, \ell)^{(i)} \mid \mathbf{y}_{1:t}, \mathbf{Z}_{0:t}, \mathbf{B})}{q_t((\mathbf{x}_t, s_{t,1:N_s}, \ell)^{(i)})}, i = 1, \ldots, M. \tag{23}$$

The auxiliary variables are discarded before proceeding to time *t*+1.
We finally note that, given Ω_t, it is straightforward to produce estimates of the target trajectory and the node locations. In particular, we can approximate the minimum mean square error (MMSE) estimate of $\mathbf{x}_t$ or $s_{1:N_s}$ at time *t* by simply computing the weighted mean of the particles in Ω_t, as shown also in Table 2.

5.2 Mixture Kalman filter

For convenience of exposition, let us begin with the case in which the locations of the sensors, $s_{1:N_s}$, are known. If we aim at the Bayesian estimation of the sequence of target positions $r_{0:t}$ conditional on the observations $\mathbf{y}_{1:t}$ (given $c_{1:N_c}$ and $s_{1:N_s}$, $\mathcal{Q}$, $\mathbf{B}$ and $\mathbf{Z}_{0:t}$ are not relevant for the estimation problem), all statistical information is contained in the posterior pdf $p(r_{0:t} \mid \mathbf{y}_{1:t}, s_{1:N_s})$, which can be approximated by means of a particle filter. Specifically, the dynamic model (1) is linear in r_t conditional on v_t, hence the recursive decomposition

$$p(r_{0:t} \mid \mathbf{y}_{1:t}, s_{1:N_s}) \propto p(\mathbf{y}_t \mid r_t, s_{1:N_s}) p(r_t \mid r_{0:t-1}) p(r_{0:t-1} \mid \mathbf{y}_{1:t-1}, s_{1:N_s}) \tag{24}$$

enables the application of the mixture Kalman filter (MKF) technique [Chen & Liu, 2000] to build a point-mass approximation of the posterior pdf,

$$p(r_{0:t-1} \mid \mathbf{y}_{1:t-1}, s_{1:N_s}) \approx \sum_{i=1}^{M} w_{t-1}^{(i)} \delta_i(r_{0:t-1}), \tag{25}$$

where δ_i is the delta measure centered at $r_{0:t-1}^{(i)}$ and $\{w_t^{(i)}\}_{i=1}^{M}$ are normalized importance weights [Doucet et al., 2000]. Given the set $\Omega_{t-1} = \{r_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{M}$, we can apply the sequential importance sampling (SIS) [Doucet et al., 2000]. algorithm to recursively compute Ω_t. For $i = 1, \ldots, M$, the following steps are recursively applied:

1. Importance sampling: Draw $r_t^{(i)} \sim p(r_t \mid r_{0:t-1}^{(i)})$.
2. Weight update:

$$w_t^{(i)^*} = w_{t-1} p(\mathbf{y}_t \mid r_t^{(i)}, s_{1:N_s}) \quad and \quad w_t^{(i)} = w_t^{(i)^*} / \sum_{k=1}^{M} w_t^{(k)^*}. \tag{26}$$

Resampling steps must also be applied (although not necessarily at each *t*) to avoid weight degeneracy [Doucet et al., 2000]. Since the likelihood $p(\mathbf{y}_t \mid r_t^{(i)}, s_{1:N_s})$ can be computed, by assumption of the model, and several methods are available for resampling, the only difficulty in the application of this algorithm is sampling from the prior $p(r_t \mid r_{0:t-1})$. The latter can be obtained from the Kalman filter (KF) equations [Kalman, 1960]. To be specific, let us note that the pair of equations jointly given by (1),

$$v_t = v_{t-1} + u_{v,t} \tag{27}$$

$$\Delta_t = r_t - r_{t-1} = T v_{t-1} + u_{r,t}, \tag{28}$$

where $[u_{r,t}, u_{v,t}]^{\mathrm{T}} = \mathbf{u}_t$ in (1), form a linear-Gaussian system. Hence, the posterior pdf of v_t, given a specific sequence $r_{0:t}$, is complex Gaussian, $p(v_t \mid \Delta_{1:t}, r_0) = CN(v_t \mid \mu_t^v, \sigma_{v,t}^2)$, with posterior mean and variance, μ_t^v and $\sigma_{v,t}^2$, respectively, that can be recursively computed using the KF recursion [Haykin, 2001],

$$\sigma_{v,t|t-1}^2 = \sigma_{v,t-1}^2 + T^2, \tag{29}$$

$$g_t = T^2 \sigma_{v,t|t-1}^2 + \frac{1}{4} T^4, \tag{30}$$

$$\mu_t^v = \mu_{t-1}^v + \sigma_{v,t|t-1}^2 \frac{T\left(\Delta_t - T\mu_{t-1}^v\right)}{g_t}, \tag{31}$$

$$\sigma_{v,t}^2 = \sigma_{v,t|t-1}^2 \left(1 - \sigma_{v,t|t-1}^2 \frac{T^2}{g_t}\right), \tag{32}$$

where g_t is the Kalman gain and $\sigma^2_{v,t|t-1}$ is the variance of v_t conditional on $\Delta_{1:t-1}$. Moreover, the normalization constant of $p(v_t \mid \Delta_{1:t}, r_0)$ is $p(\Delta_t \mid \Delta_{1:t-1}, r_0) = p(r_t - r_{t-1} \mid r_{0:t-1})$, which is also complex Gaussian and can be analytically found. In our specific model we obtain

$$p(r_t \mid r_{0:t-1}) = CN\left(r_t \mid r_{t-1} + T\mu^v_{t-1}, \sigma^2_{r,t|t-1}\right), \tag{33}$$

where $\sigma^2_{r,t|t-1} = T^2\sigma^2_{v,t-1} + \frac{5}{4}T^4$.

Therefore, the outlined MKF algorithm can be implemented using a bank of M KFs, one per particle. Given $r^{(i)}_{0:t-1}$, it is possible to (recursively and analytically) obtain $p(r_t \mid r^{(i)}_{0:t-1})$, which can be easily sampled to draw $r^{(i)}_t$ in the importance sampling step. Moreover, both r_t and v_t can be estimated (in the MMSE sense),

$$\hat{r}_t = \sum_{i=1}^{M} w^{(i)}_t r^{(i)}_t, \quad \hat{v}_t = \sum_{i=1}^{M} w^{(i)}_t \mu^{v^{(i)}}_t, \tag{34}$$

by combining the outputs of the KF's, hence the name MKF. This methodology was applied to generic tracking problems in [Gustafsson et al., 2002].

Unfortunately, the standard MKF algorithm does not provide means to properly handle the unknown sensor positions $s_{1:N_s}$. To overcome this limitation, we resort to the DA-PF scheme of [Djurić et al., 2004]. The basic probabilistic relationship that we exploit to derive the new algorithm is obtained by means of the Bayes theorem and the repeated decomposition of conditional probabilities, namely

$$\begin{aligned} p(r_{0:t}, s_{1:N_s} \mid \mathbf{y}_{1:t}, \mathbf{Z}_{0:t}, \mathbf{B}) &\propto p(\mathbf{y}_t \mid r_t, s_{1:N_s})\, p(\mathbf{Z}_t \mid s_{1:N_s})\, p(r_t \mid r_{0:t-1}) \\ &\quad \times \rho_t(s_{1:N_s})\, p(r_{0:t-1} \mid \mathbf{y}_{1:t-1}, \mathbf{Z}_{0:t-1}, \mathbf{B}) \end{aligned} \tag{35}$$

$$\begin{aligned} &= \prod_{k=1}^{t} p(\mathbf{y}_k \mid r_k, s_{1:N_s})\, p(\mathbf{Z}_k \mid s_{1:N_s})\, p(r_k \mid r_{0:k-1}) \\ &\quad \times p(r_0)\, p(s_{1:N_s} \mid \mathbf{Z}_0, \mathbf{B}), \end{aligned} \tag{36}$$

where

$$\rho_t(s_{1:N_s}) = p(s_{1:N_s} \mid r_{0:t-1}, \mathbf{y}_{1:t-1}, \mathbf{Z}_{0:t-1}, \mathbf{B}) \tag{37}$$

is the posterior pdf of $s_{1:N_s}$ at time t-1.

Assume that we are able to draw samples from $\rho_t(s_{1:N_s})$. Then, (35) shows that a SMC algorithm can be used to recursively approximate $p(r_{0:t}, s_{1:N_s} \mid \mathbf{y}_{1:t}, \mathbf{Z}_{0:t}, \mathbf{B})$. Indeed, if at time t-1 the set of particles $\Omega_{t-1} = \{r^{(i)}_{0:t-1}, s^{(i)}_{t-1,1:N_s}, w^{(i)}_t\}_{i=1}^{M}$ is available, then we can compute a point-mass approximation of the last factor in (35),

$$p_M(r_{0:t-1} \mid \mathbf{y}_{1:t-1}, \mathbf{Z}_{0:t-1}, \mathbf{B}) = \int \sum_{i=1}^{M} w_{t-1}^{(i)} \delta_i(r_{0:t-1}, s_{1:N_s}) ds_{1:N_s} \tag{38}$$

$$= \sum_{i=1}^{M} w_{t-1}^{(i)} \delta_i(r_{0:t-1}), \tag{39}$$

where the integrand in (38) is the approximation of $p(r_{0:t-1}, s_{1:N_s} \mid \mathbf{y}_{1:t-1}, \mathbf{Z}_{0:t-1}, \mathbf{B})$ built from Ω_{t-1}. Eq. (39) implies that we can start from the set $\Omega_{t-1} = \{r_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{M}$ and exploit (35) to build Ω_t via the MKF algorithm. Specifically,

$$r_t^{(i)} \sim p(r_t \mid r_{0:t-1}^{(i)}) \tag{40}$$

$$s_{t,1:N_s}^{(i)} \sim \rho_t(s_{1:N_s}) \tag{41}$$

$$w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t \mid r_t^{(i)}, s_{t,1:N_s}^{(i)}) p(\mathbf{Z}_t \mid s_{t,1:N_s}^{(i)}). \tag{42}$$

Moreover, (36) explicitly shows that, in order to start the recursion, we need to draw initial populations not only from the prior $p(r_0)$, but also from the posterior $p(s_{1:N_s} \mid \mathbf{Z}_0, \mathbf{B})$. MMSE estimates of r_t and v_t are computed in the same way shown by Eq. (34), while $\hat{s}_{t,1:N_s} = \sum_{i=1}^{M} s_{t,1:N_s}^{(i)} w_t^{(i)}$.

To apply the MKF algorithm (40)-(42) we only need to specify how to approximate $\rho_t(s_{1:N_s})$. One of the simplest choices is to assume a Gaussian distribution built from the particles and weights at time *t*-1, i.e.,

$$\rho_t(s_{1:N_s}) \approx \prod_{n=1}^{N_s} CN(s_n \mid \bar{s}_{t-1,n}, \sigma_{t-1,s,n}^2), \tag{43}$$

where $\bar{s}_{t-1,n} = \sum_{i=1}^{M} w_{t-1}^{(i)} s_{t-1,n}^{(i)}$ and $\sigma_{t-1,s,n}^2 = \sum_{i=1}^{M} w_{t-1}^{(i)} \left(s_{t-1,n}^{(i)} - \bar{s}_{t-1,n}\right)^2$. This approximation is easy to sample and still provides an acceptable performance, as will be numerically shown in Section 7.

6. Fusion

6.1 Point estimation

Assume that the tracking algorithms, either the APF or the DA-MKF, run independently in the separate DFCs, using the same sequence of sensor observations[2], $\mathbf{y}_{0:t}$, but different DFC

[2]This is not a requirement for fusion based on point estimation. The same technique could be applied if each DFC collected a different sequence $\mathbf{y}_{n,1:t}$, $n = 1,...,N_c$.

observations, $\mathbf{z}_{n,0:t}$, $n=1,...,N_c$, instead of the whole set $\mathbf{Z}_{0:t}$. At any time t, the PF can yield an MMSE point estimate of the target and sensor positions. Let $\hat{r}_{0:t}(n)$ and $\hat{s}_{1:N_s}(n)$ denote the estimates computed at the n-th DFC. We wish to combine all available estimates coherently to obtain an improved estimator. However, we must take into account the possibility that n -th DFC estimates may be rotated around the location c_n, as a consequence of the insensitivity of the observations $\mathbf{y}_t$ and $\mathbf{z}_{n,t}$ to the angle between r_t and s_k, k=1,..., N_s, and the angle between s_k and c_n, respectively.
In order to correct any possible rotation and gurantee the computation of a coherent average of the available estimates, we propose the following fusion rule for the overall target and sensor position estimator,

$$\hat{r}_t = \frac{1}{N_c}\sum_{n=1}^{N_c}\hat{c}_n + \left(\hat{r}_t(n) - \hat{c}_n\right)e^{j\phi_n} \tag{44}$$

$$\hat{s}_{t,k} = \frac{1}{N_c}\sum_{n=1}^{N_c}\hat{c}_n + \left(\hat{s}_{k,t}(n) - \hat{c}_n\right)e^{j\phi_n}, \tag{45}$$

where k=1,..., N_s and the correction angles $\phi_1,...,\phi_{N_c}$ are selected to minimize the mismatch with respect to the estimates produced by DFC 1, which is (arbitrarily) chosen as a reference. Specifically,

$$\phi_n = \arg\min_{\phi\in[0,2\pi)}\left\{\sum_{i=1}^{N_s}\left|\hat{s}_i(1) - \left[\hat{c}_n + (\hat{s}_i(n) - \hat{c}_n)e^{j\phi}\right]\right|^2\right.$$

$$\left. + \sum_{k=t-L_t}^{t}\left|\hat{r}_k(1) - \left[\hat{c}_n + (\hat{r}_k(n) - \hat{c}_n)e^{j\phi}\right]\right|^2\right\}, \tag{46}$$

where L_t>0 is a delay lag and $j=\sqrt{-1}$. Problem (46) can be solved using the ARS algorithm described in Section 4.

6.2 Distributed implementation of PFs

In this section we describe how a centralized PF can be implemented in a distributed manner, such that each DFC updates a distinct subset of particles and generates individual estimates that can subsequently be combined optimally. For this approach to be formally sound, we require that all observations (in particular, the sequence $\mathbf{Z}_{0:t}$) be available to all DFCs. If this is not the case, the fusion tchnique described here can still be used, but it becomes an approximation and optimality cannot be claimed.
We propose to use the resampling with non-proportional allocation (RNA) method of [Bolić et al., 2005, Míguez, 2007] 7 to distribute the MKF tracking algorithm over the N_c DFCs. Let us split the overall particle set Ω_t into N_c subsets, one per DFC, denoted as $\Omega_{n,t} = \{r_{0:t}^{(n,i)}, s_{t,1:N_s}^{(n,i)}, w_t^{(n,i)}, W_t^{(n)^*}\}_{i=1}^{M_n}$, $n=1,\ldots,N_c$, and such that $\sum_n M_n = M$. The

weights in $\Omega_{n,t}$ are normalized locally, i.e., $\sum_{i=1}^{M_n} w_t^{(n,i)} = 1$, and the sum of the unnormalized weights, $W_t^{(n)^*} = \sum_{i=1}^{M_n} w_t^{(n,i)^*}$ are also kept in order to assess the relative value of each subset (the subsets $\Omega_{n,t}$ are not equally "good" in general).

In the basic RNA scheme, an independent MKF algorithm (40)-(41) is run for each subset $\Omega_{n,t}$ (i.e., for each DFC). This means that resampling is carried out locally (using any desired method) at each DFC and estimates are also computed locally,

$$\hat{r}_t(n) = \sum_{i=1}^{M_n} w_t^{(n,i)} r_t^{(n,i)} \quad and \quad \hat{s}_{1:N_s}(n) = \sum_{i=1}^{M_n} w_t^{(n,i)} s_{t,1:N_s}^{(n,i)}. \tag{47}$$

Optimal fusion is performed by combining the local estimates according to the sum-weights, $W_t^{(n)^*}$, i.e., global MMSE estimates are computed as

$$\hat{r}_t = \frac{\sum_{n=1}^{N_c} W_t^{(n)^*} \hat{r}_t(n)}{\sum_{k=1}^{N_c} W_t^{(k)^*}} \quad and \quad \hat{s}_{1:N_s} = \frac{\sum_{n=1}^{N_c} W_t^{(n)^*} \hat{s}_{1:N_s}(n)}{\sum_{k=1}^{N_c} W_t^{(k)^*}}, \tag{48}$$

in such a way that only the local estimates and the sum-weights need to be transmitted to the DFC in charge of the fusion stage.

One limitation of this approach is that when the subset sizes, M_n, n=1,..., N_c, are not large enough, some particle filters may get relatively impoverished [Míguez,2007], i.e., it may eventually happen that, for some n, $W_t^{(n)^*} << W_t^{(k)^*}$, for all $k \neq n$. In such a case, the corresponding n -th DFC becomes 'useless', since its local estimates are essentially irrelevant for the computation of the global estimates. A solution to this phenomenon (equivalent to the weight degeneracy in standard particle filters [Doucet et al., 2000]) is to periodically perform a local exchange (LE) of a small number of particles between pairs of DFCs. We propose a simple implementation of LE in which L<min$_n${M_n} particles from DFC n are transmitted to DFC n+1, for n=1,..., N_c -1 and L particles from DFC N_c are transmitted to DFC 1, i.e., particles are exchanged in a ring configuration.

7. Simulations

In order to provide illustrative numerical results, we have particularized the model of Section 2 to a network of power-aware sensors. Specifically, the measurement function $f_s(\cdot,\cdot)$ has the form

$$f_s(d,\varepsilon) = 10\log_{10}\left(\frac{1}{d^2} + \eta\right) + \varepsilon, \quad (dB) \tag{49}$$

where $\eta = 10^{-6}$ accounts for the sensitivity of the measurement device (-60 dB). The n-th sensor transmits its measurement, $y_{n,t}$, only if it corresponds to a distance $d_{n,t} < S_y = 50$

m (i.e., $y_{n,t} > -33.97$ dB) and otherwise remains silent. A transmission failure can also occur, with probability $\beta = 10^{-4}$. The observational noise, ε, is zero mean Gaussian but, depending on whether the power observation is carried out at a sensor node or at a DFC node, its variance is assumed different. In particular $\varepsilon_{n,t}^{y} \sim N(0,2)$, for sensors, and, for DFCs, $\varepsilon_{i,n,t}^{z}$ and $\varepsilon_{i,n}^{0}$ are identically distributed according to the Gaussian pdf $N(0,10^{-2})$. Therefore, the likelihoods, namely,

$$p(y_t \mid r_t^{(m)}, s_{1:N_s}^{(m)}) = \prod_{l=1}^{N_t} p(y_{\kappa(l),t} \mid r_t^{(m)}, s_{\kappa(l)}^{(m)}) \tag{50}$$

$$p(\mathbf{Z}_t \mid s_{1:N_s}^{(m)}, c_1, \hat{c}_{2:N_c}) = \prod_{i=1,n=1}^{N_t,N_c} p(z_{\kappa(i),n,t} \mid s_{t,\kappa(i)}^{(m)}, \hat{c}_n) \tag{51}$$

are Gaussian with known mean and variance.

At time zero, the sensors detect all other nodes which are closer than S_u=50 m. Since observations are obtained from function $f_s(\cdot,\cdot)$, with the parameters already described for the sensors, the probability of detection is

$$p_d(d_{n,k}^{s}, S_u) = \Phi_N\left(\frac{\overline{P}_{n,k} - P_u}{\sqrt{10^{-2}}}\right), \tag{52}$$

where $\Phi_N(\cdot)$ is the standard (zero mean, unit variance) Gaussian cumulative distribution function, $\overline{P}_{n,k} = f_s(d_{n,k}^{s},0)$, $P_u = f_s(S_u,0) = -33.97$ dB and 10^{-2} is the variance of the observational noise.

The state priors are $p(r_0) = CN(r_0 \mid 0,5)$ and $p(v_0) = CN(v_0 \mid 0,10^{-2})$ and the state equation parameters are $T = \frac{1}{2}$ s and $\sigma_u^2 = \frac{1}{5}$. There are $N_c = 4$ DFCs and $N_s = 23$ sensors in the network. We assume $c_1 = 0$, while the others have complex Gaussian priors with equal variance $\sigma_c^2 = 25$ and means $\mu_2^c = -50 + j35$, $\mu_3^c = 45 - j37$ and $\mu_4^c = 36 + j45$ (where $j = \sqrt{-1}$), respectively. This prior pdf's are used to randomly draw initial estimates of $c_{2:4}$ which are used as inputs to the ARS algorithm that solves (5), the other parameters being $\nu = 2$, $R_{max} = 15$, $R_{min} = 10^{-4}$. The ARS algorithm for problem (6) receives as inputs a sensor position drawn from $U(\mathcal{C})$ (where $\mathcal{C}$ is the square centered at 0 with sides of length 160 m), $R_{max} = 200$, $r_{min} = 10^{-4}$ and $\nu = 2$. The ARS procedures are iterated 3000 times for (5) and 300 times for each (6). The estimates $\hat{s}_{1:N_s}$ are then used to build the first proposal pdf in the PMC procedure. The corresponding variance is $\sigma_s^2(0) = \frac{1}{2}$ and the subsequent proposals are computed by shrinkage, with parameter $a = 0.7$. We iterate the PMC algorithm 200 times with 1600 particles.

Resampling, via the RNA scheme, is performed every 5 time steps of the tracking algorithms. The latter are run with M=1600 particles and each DFC is assigned $M_n = M/N_c = 400$ particles (for n=1,2,3,4). We assume a local exchange of particles every 4 resampling steps, with L=8 particles being transmitted from DFC n to DFC n+1 and from DFC N_c to DFC 1.

Figure 1 shows an example of estimation of the DFC and sensor positions, $c_{2:N_c}$ and $s_{1:N_s}$, respectively, using the available data at time t=0 and the ARS algorithm. It is observed that MAP estimation using the obervations in the initialization stage can be very accurate. However, the simulation results also illustrate the ambiguity in position estimation that arises due to the insensitivity of the available measurements to rotations. The plot shows that the estimation of node locations near the origin (where angle errors have little effect) is clearly more accurate than for nodes placed far away from the 0 point, where rotation errors cause an apparent shift of the estimates with respect to the true values.

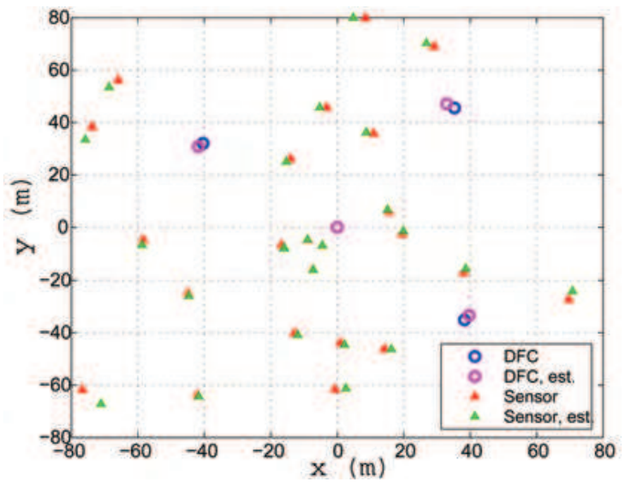


Fig. 1. Example of node MAP position estimation at the initialization stage using the ARS algorithm. The phenomenon of rotation ambiguity (due to the angle insensitivity of the available measurements) can be clearly observed for nodes located far from the 0 (origin) point.

Next we turn attention to the performance of the schemes that use independent PFs at separate DFCs and employ the proposed angle correction method to fuse the N_c available estimates coherently, as described in Section 6.1. We recall that, in this scenario, the observations available to the PF in the n-th DFC during the tracking stage are $y_{1:t}$ and $z_{n,1:t}$.

The ARS algorithm that approximates the solutions (correction angles) in problem 46 is iterated 100 times, with initial values $\phi_{n>1} = 0$ and time lag L_t=20 (for t>20, and L_t=t otherwise).

Figure 2 (left) shows the mean absolute target position error attained with the APF and MKF algorithms in this scenario. These results are the average of 500 independent simulation trials. The APF technique turns out to be clearly superior and outperforms the MKF method by ≈0.6 m of accuracy. This means that the improved importance function employed by the APF algorithm (which takes into account $\mathbf{y}_t$ and $\mathbf{z}_{n,t}$ when drawing the particles at time t)

provides an error reduction superior to the analytical integration of the velocity process carried out by the MKF method. We must remark that the position errors shown in this plot are corrected to remove the insensitivity, inherent to the proposed model, to rotations of the complete system (including the target and the nodes) around point 0, where DFC 1 is assumed to be located.

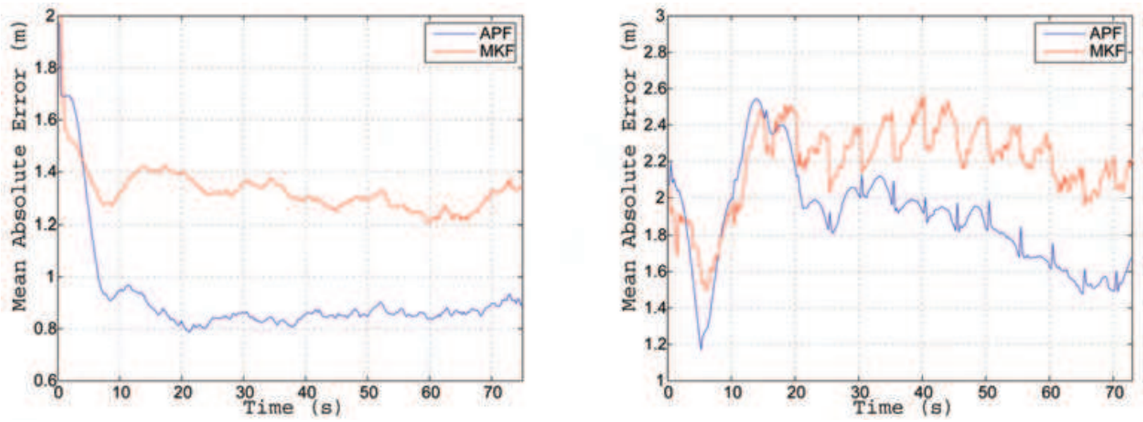


Fig. 2. Averaged absolute error in the estimation of the target position, measured in meters (m). *Left:* Tracking is carried out using independent APF and DA-MKF algorithms at each DFC and performing fusion by coherent combination of point estimates. *Right:* Tracking is carried out using a distributed implementation of the APF and DA-MKF algorithms over the N_c=4 DFCs, by means of the RNA technique.

Figure 3 (left) depicts the mean absolute error in the estimation of the sensor positions attained by the schemes built around the APF and MKF algorithms. Again, these results are the average of 500 independent simulation trials. In this case, more accurate results are obtained with the MKF algorithm. This means that the density-assisted approach to the estimation of fixed parameters in the MKF scheme is more efficient than the adaptive kernel approximation employed by the APF procedure. The difference, however, is small (≈0.06 m) and the superior sampling efficiency of the proposal function in the APF technique obviously has a dominant role in the overall performance of the tracker.

Figures 2 (right) and 3 (right) show the results, for the absolute errors in the estimation of the target and sensors positions, obtained when we apply the RNA algorithm described in Section 6.2. The curves have been obtained by averaging the results of 500 independent simulation trials. In this scenario, the different DFCs cooperate to share the observtions $\mathbf{z}_{n,t}$, n=1,...,N_c, hence the matrix $\mathbf{Z}_t$ is available for all DFCs at time t. Again, the APF tracker turns out more efficient than the MKF in turns of target positioning, whereas the MKF yields better estimates of the sensor locations. Both plots show a clear `step' shape. The reason is the sudden improvement in the estimates that occurs when a local exchange of particles is carried out as a part of the RNA procedure. This suggests the need to increase the number of exchanged particles (L=8 for this set of simulations) or even the number of particles assigned to each DFC, which seems too small to exploit the potential of the RNA scheme. Indeed, the accuracy of positioning when using coherent point estimation is superior (for the APF, there is an advantage of ≈0.8 meters in absolute error). This fact should be attributed to the averaging of the angle error that is carried out when selecting the correction angles $\phi_{2:N_c}$ in Eq. (46). Nevertheless, our simulation results (not shown) indicate that RNA-based schemes

always outperform point-estimation methods when the system runs for ≈300 time steps or more.

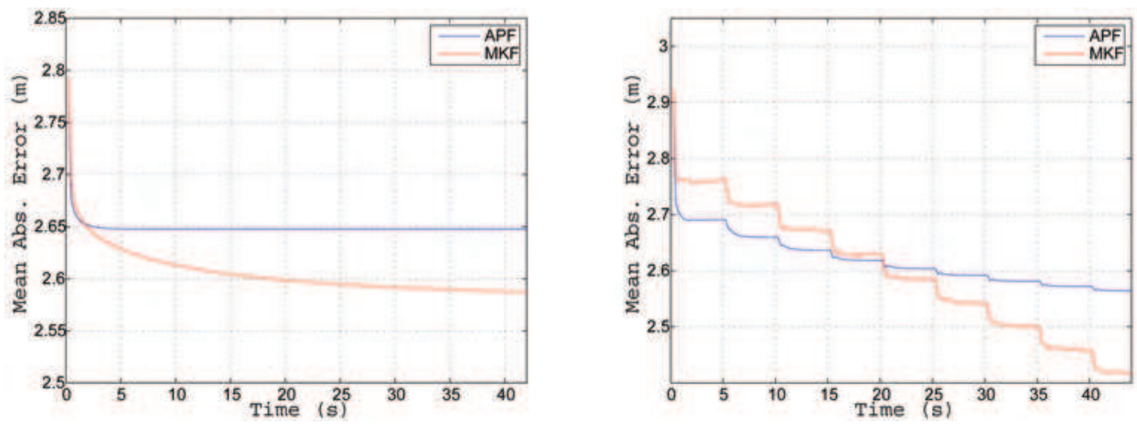


Fig. 3. Averaged absolute error in the estimation of the sensor positions, measured in meters/second (m). *Left:* Estimation is carried out using independent APF and DA-MKF algorithms at each DFC and performing fusion by coherent combination of point estimates. *Right:* Estimation is carried out using a distributed implementation of the APF and DA-MKF algorithms over the N_c=4 DFCs, by means of the RNA technique.

8. Summary

We have proposed a novel scheme for joint node localization and target tracking in wireless sensor networks using Monte Carlo methods. The proposed approach does not require the aid of beacons in order to locate the network nodes. Instead, it resorts to a novel combination of Monte Carlo optimization and iterated sampling procedures in order to generate an initial population of node locations with sufficient quality. Starting from this population, we have described novel particle filtering algorithms that recursively track the target position *and* sequentially generate new samples of node positions, as new data become available, in order to improve node positioning. For networks equipped with more than one data fusion center, we have also proposed two schemes that enable the combination of the estimates obtained by different fusion centers, possibly using different data. Computer simulation results have also been presented to illustrate the performance of the proposed techniques.

9. Acknowledgements

This work has been partially supported by the Ministry of Science and Innovation of Spain (project MONIN, ref. TEC-2006-13514-C02-01/TCM), the Ministry of Industry, Tourism and Commerce of Spain (project TIMI, ref. CENIT-2002-2007) and the Autonomous Community of Madrid (project PROMULTIDIS-CM, ref. S-0505/TIC/0233).

10. References

M. J. Appel, R. Labarre, and D. Radulovic. On accelerated random search. *SIAM Journal of Optimization*, 14(3):708–730, 2003.

M. Bolić, P. M. Djurić, and S. Hong. Resampling algorithms and architectures for distributed particle filters. *IEEE Transactions Signal Processing*, 53(7):2442–2450, July 2005.

O. Cappé, A. Gullin, J. M. Marin, and C. P. Robert. Population monte carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929, 2004.

R. Chen and J. S. Liu. Mixture Kalman filters. *Journal of the Royal Statistics Society B*, 62:493–508, 2000.

D. Crisan and A. Doucet. A survey of convergence results on particle filtering. *IEEE Transactions Signal Processing*, 50(3):736–746, March 2002.

M. H. DeGroot and M. J. Schervish. *Probability and Statistics, 3rd ed.* Addison- Wesley, New York, 2002.

P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Míguez. Particle filtering. *IEEE Signal Processing Magazine*, 20(5):19–38, September 2003.

P. M. Djurić, M. F. Bugallo, and J. Míguez. Density assisted particle filters for state and parameter estimation. In *Proceedings of the 29th IEEE ICASSP*, May 2004.

A. Doucet, S. Godsill, and C. Andrieu. On sequential Monte Carlo Sampling methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 2000.

A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods in Practice*. Springer, New York (USA), 2001.

L. Fang, W. Du, and P. Ning. A beacon-less location discovery scheme for wireless sensor networks. In *Proceedings of INFOCOM*, volume 1, pages 161–171, March 2005.

F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund. Particle filters for positioning, navigation and tracking. *IEEE Transactions Signal Processing*, 50(2):425–437, February 2002.

S. Haykin. *Adaptive Filter Theory, 4th Edition*. Prentice Hall, Information and System Sciences Series, 2001.

A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky. Nonparametric belief propagation for self-localization of sensor networks. *IEEE Transactions on Selected Areas in Communications*, 23(4):809–819, April 2005.

R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82:35–45, 1960.

J. Liu and M. West. Combined parameter and state estimation in simulation-based filtering. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*, chapter 10, pages 197–223. Springer, 2001.

J. Míguez. Analysis of selection methods for cost-reference particle filtering with applications to maneuvering target tracking and dynamic optimization. *Digital Signal Processing*, 17:787–807, 2007.

N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero III, R. L. Moses, and N. S. Correal. Locating the nodes. *IEEE Signal Processing Magazine*, 22(4):54–69, July 2005.

M. K. Pitt and N. Shephard. Auxiliary variable based particle filters. In A. Doucet, N. de Freitas, and N. Gordon, editors, *Sequential Monte Carlo Methods in Practice*, chapter 13, pages 273–293. Springer, 2001.

B. Ristic, S. Arulampalam, and N. Gordon. *Beyond the Kalman Filter*. Artech House, Boston, 2004.

G. Sun, J. Chen, W. Guo, and K. J. R. Liu. Signal processing techniques in networkaided positioning. *IEEE Signal Processing Magazine*, 22(4):12–23, July 2005.

M. Vemula, M.F. Bugallo, and P.M. Djurić. Fusion of information for sensor selflocalization by a Monte Carlo method. In *Proceedings of ICIF*, July 2006.



Sensor and Data Fusion

Edited by Nada Milisavljevic

ISBN 978-3-902613-52-3
Hard cover, 436 pages
Publisher I-Tech Education and Publishing
Published online 01, February, 2009
Published in print edition February, 2009

Data fusion is a research area that is growing rapidly due to the fact that it provides means for combining pieces of information coming from different sources/sensors, resulting in ameliorated overall system performance (improved decision making, increased detection capabilities, diminished number of false alarms, improved reliability in various situations at hand) with respect to separate sensors/sources. Different data fusion methods have been developed in order to optimize the overall system output in a variety of applications for which data fusion might be useful: security (humanitarian, military), medical diagnosis, environmental monitoring, remote sensing, robotics, etc.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

Joaquín Míguez, Luis Arnaiz and Antonio Artés-Rodríguez (2009). Monte Carlo Methods for Node Self-Localization and Nonlinear Target Tracking in Wireless Sensor Networks, Sensor and Data Fusion, Nada Milisavljevic (Ed.), ISBN: 978-3-902613-52-3, InTech, Available from: http://www.intechopen.com/books/sensor_and_data_fusion/monte_carlo_methods_for_node_self-localization_and_nonlinear_target_tracking_in_wireless_sensor_netw

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2009 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen