Multitarget Tracking with Interacting Population-based MCMC-PF

Mélanie Bocquel, Hans Driessen
Thales Nederland B.V. - Sens TBU Radar Engineering,
Hengelo, Nederland
Email: {Melanie.Bocquel, Hans.Driessen} @nl.thalesgroup.com

Arun Bagchi
University of Twente - Department of Applied Mathematics
Enschede, Nederland
Email: a.bagchi@ewi.utwente.nl

***Abstract*—In this paper we address the problem of tracking multiple targets based on raw measurements by means of Particle filtering. This strategy leads to a high computational complexity as the number of targets increases, so that an efficient implementation of the tracker is necessary.**

We propose a new multitarget Particle Filter (PF) that solves such challenging problem. We call our filter Interacting Population-based MCMC-PF (IP-MCMC-PF) since our approach is based on parallel usage of multiple population-based Metropolis-Hastings (M-H) samplers. Furthermore, to improve the chains mixing properties, we exploit genetic alike moves performing interaction between the Markov Chain Monte Carlo (MCMC) chains.

Simulation analyses verify a dramatic reduction in terms of computational time for a given track accuracy, and an increased robustness w.r.t. conventional MCMC based PF.

I. Introduction

Multitarget tracking is a well-known problem which consists of sequentially estimating the states of several targets from noisy data. It arises in many applications, e.g. radar based tracking of aircraft. Bayesian methods provide a rigorous general framework for dynamic state estimation problems. The Bayesian recursions update the posterior probability density function (pdf) of the state based on all available information, including the set of received measurements. The application of the Bayesian sequential estimation framework to multitarget tracking problems is plagued by two difficulties. First, the state and observation models are often non-linear and non-Gaussian, so that no closed-form analytic expression can be obtained for the tracking recursions. The second difficulty is due to the fact that in most practical tracking applications the sensors yield unlabeled measurements of the targets. This leads to a challenging data association problem.

The multitarget tracking problem has been traditionally addressed with techniques such as multiple hypothesis tracking (MHT) and joint probabilistic data association (JPDA), which require plot measurements (detection) and a measurement-to-track association procedure [1]. Solutions using a Particle Filter (PF) have been proposed in the past ten years [2]–[4]. These so-called Track-Before-Detect (TBD) approaches define a model for the raw measurements in terms of a multi-target state hypothesis, thus avoiding an explicit data association step. Furthermore, as opposed to the conventional thresholded measurement procedure, these strategies allow the tracker to perform well in the low Signal-to-Noise Ratio (SNR) regime.

Particle filtering is a Sequential Monte Carlo (SMC) simulation-based method of approximately solving the Bayes prediction and update equations recursively using a stochastic samples (particles) cloud [5]. The Sampling Importance Resampling (SIR) Multitarget PF, that recursively estimates the joint multitarget probability density (JMPD), has demonstrated its ability to successfully perform nonlinear and non-Gaussian tracking and to enable more accurate modeling of the target correlations. While the SIR PF is fairly easy to implement and tune, its main drawbacks are the computational complexity which increases rapidly with the state dimensions and the loss of diversity among the particles due to resampling. To mitigate the former problem an adaptive sampling strategy, called the Adaptive Proposal (AP) method [3], has been suggested. This strategy proposes to construct particle proposals at the partition level, incorporating the measurements so as to bias the proposal towards the optimal importance density. The AP method automatically factors the JMPD when targets are behaving independently [6], while attempt to handle the permutation symmetry and correlations that arise when targets are coupled [3]. The latter problem can effectively be addressed by using Particle MCMC (PMCMC) methods [7], [8], e.g. the Particle Marginal Metropolis-Hastings (PMMH) algorithm.

In recent years there has been an increasing interest towards Markov Chain Monte Carlo (MCMC) methods to simulate complex, nonstandard multivariate distributions. There are two basic MCMC techniques: the Gibbs sampler [9], [10], and the Metropolis-Hastings (M-H) [11], [12] algorithm. The former method is based on sampling from the collection of full conditional distributions, while the latter approach is a generalization that can be used when the full conditionals cannot be written in a closed form.

Recently, the research done on Markov chains and Hidden Markov Models has led the definition of the M-H algorithm through the notion of reversibility [13], and allowed for the derivation of convergence results for Sequential MCMC methods, e.g. the PMCMC. Although these methods are very reliable, the computational expense of PMCMC algorithms for complex models remains a limiting factor. Furthermore, the mixing of the resulting MCMC kernels can be quite sensitive, both to the number of particles and the measurement space dimension.

In this paper we focus on issues arising in the implementation of multitarget Bayesian filtering. Thus, we propose an alternative algorithm, which fully exploits the speed and the parallelism of modern computing architectures and resources. We introduce a new M-H based PF in which several MCMC chains will be running in parallel, this way generating *population-based* samples to approximate the target distribution. Furthermore, to improve the *mixing properties* of the resulting MCMC kernel, the diversity of each population is increased by means of an *interaction procedure* [14]. Note that we will assume throughout this paper that the number of targets M is known and constant over time.

The paper is organized as follows: in section II we review the Bayesian Filtering problem; section III briefly reviews the basics of particle filtering and MCMC methods for sampling. Section IV reports the main part of our work. Here we introduce our new algorithm, which we called Interacting Population-based MCMC-PF (IP-MCMC-PF), discuss the improvements attainable by using a fully parallelizable PF, and give theoretical justifications for our approach. In section V the system dynamics and measurement model are introduced for the specific TBD surveillance application. Section VI collects our simulations results. Finally, we report our conclusions and direction for future research in section VII.

II. Bayesian Filtering Problem

In this section, we briefly describe the Bayesian Filtering problem. Let us consider the following dynamical system:

$$\mathbf{x}_{k+1} = f_k(\mathbf{x}_k, \mathbf{v}_k), \tag{1}$$

$$\mathbf{z}_k = h_k(\mathbf{x}_k) + \mathbf{w}_k, \tag{2}$$

where $\mathbf{x}_k \in \mathbb{R}^{n_x}$ denotes the state vector, $\mathbf{z}_k \in \mathbb{R}^{n_z}$ the system measurement, $\mathbf{v}_k \sim p_{\mathbf{v}_k}(\mathbf{v})$ the process noise, and $\mathbf{w}_k \sim p_{\mathbf{w}_k}(\mathbf{w})$ the measurement noise.
Furthermore, let $\mathbf{Z}_k \triangleq \{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$ be the sequence of measurements collected up to time k. The measurement $\mathbf{z}_k$ is assumed to be independent from the past states, i.e.,

$$p(\mathbf{z}_k|\mathbf{z}_{k-1}, \ldots, \mathbf{z}_1, \mathbf{x}_k, \mathbf{x}_{k-1}, \ldots, \mathbf{x}_0) = p(\mathbf{z}_k|\mathbf{x}_k), \tag{3}$$

where $p(\mathbf{z}_k|\mathbf{x}_k)$ denotes the *likelihood* function.

Given a realization of $\mathbf{Z}_k$, Bayesian filtering boils down to finding an approximation of the posterior pdf $p(\mathbf{x}_k|\mathbf{Z}_k)$. In particular, the *marginal filtering distribution* $p(\mathbf{x}_k|\mathbf{Z}_k)$ is obtained at time k by means of a two-step recursion:
(1) the Prediction Step, which is solved using the *Chapman-Kolmogorov equation*, i.e.,

$$p(\mathbf{x}_k|\mathbf{Z}_{k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})\ p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}) d\mathbf{x}_{k-1} \tag{4}$$

where, $p(\mathbf{x}_k|\mathbf{Z}_{k-1})$ is the predictive density at time k;
(2) the Update Step, which is solved using *Bayes theorem*, i.e.,

$$p(\mathbf{x}_k|\mathbf{Z}_k) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)\ p(\mathbf{x}_k|\mathbf{Z}_{k-1})}{p(\mathbf{z}_k|\mathbf{Z}_{k-1})} \tag{5}$$

where $p(\mathbf{z}_k|\mathbf{Z}_{k-1})$ is the Bayes normalization constant (*evidence*). Thus, the filtering pdf is completely specified given some prior $p(\mathbf{x}_0)$, a transition kernel $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ and a likelihood $p(\mathbf{z}_k|\mathbf{x}_k)$.

Given the a posteriori pdf $p(\mathbf{x}_k|\mathbf{Z}_k)$, some averaged statistics of interest can be calculated, e.g. the mean-squared error (MSE), i.e.,

$$\mathbb{E}[\|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2|\mathbf{Z}_k] = \int \|\mathbf{x}_k - \hat{\mathbf{x}}_k\|^2 p(\mathbf{x}_k|\mathbf{Z}_k) d\mathbf{x}_k,$$

which can be used to derive the minimum variance estimator,

$$\hat{\mathbf{x}}_k^{MV} \triangleq \int \mathbf{x}_k\ p(\mathbf{x}_k|\mathbf{Z}_k) d\mathbf{x}_k. \tag{6}$$

III. Fundamentals

In this section we recall the basics of Markov Chain Monte Carlo, Particle Filtering and Particle MCMC methods, necessary for clarifying the strengths of our method. In fact, in section IV we propose an approach for state estimation that combines the differing pros of both MCMC and PF based signal processing. Specifically, subsection III-A is dedicated to MCMC methods; subsection III-B reviews Particle Filtering techniques; while subsection III-C is used to discuss related work on Particle MCMC methods.

A. Basic of Markov Chain Monte Carlo process

Let $\mathcal{X} \subseteq \mathbb{R}^M$ be a compact set and suppose that $\pi(\mathbf{x})$ is a probability distribution on such a high-dimensional space $\mathcal{X}$. Denote by $\mathcal{B}(\mathcal{X})$ the Borel σ-field and by $(\mathcal{X}, \mathcal{B}(\mathcal{X}))$ the associated measure space. Then, thanks to Markov chains theory and Monte Carlo sampling, the following holds:

A Markov Chain Monte Carlo method for the simulation of the distribution π is any method which generates an ergodic Markov chain $\{\mathbf{x}_k^{(i)}\}_{i\geq 1}$ on $\mathcal{X}$, according to a kernel $K(.,.)$ defined on $(\mathcal{X} \times \mathcal{B}(\mathcal{X}))$, with π as stationary distribution.

Let us recall two basic concepts:

1) π is the *stationary distribution* of the Markov chain, i.e., for all $A \in \mathcal{B}(\mathcal{X})$,
$$\pi(A) = \int_{\mathcal{X}} K(x, A)\pi(x)dx. \tag{7}$$
2) The Markov chain is *ergodic*, i.e. for any integrable function $q : \mathcal{X} \to \mathbb{R}$
$$\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} q(\mathbf{x}_k^{(i)}) \to \mathbb{E}_\pi(q) = \int_{\mathcal{X}} q(x)\pi(x)dx \tag{8}$$
with probability 1, $\forall\ \mathbf{x}_k^{(0)} \in \mathcal{X}$, where N is the number of iterations of the MCMC algorithm.

The *Metropolis-Hastings* (M-H) algorithm [15] is a well known procedure based on a Markovian process which fulfills the requirement of ergodicity [16]. The M-H algorithm employs a conditional density q, also known as proposal distribution, to generate a Markov chain with an invariant distribution π. At each MCMC iteration i, a move $\mathbf{x}' \sim q(\mathbf{x}'|\mathbf{x}^{(i-1)})$ is proposed and accepted with a probability $0 < \alpha(\mathbf{x}^{(i-1)}, \mathbf{x}') < 1$.

Notice that the initial *burn-in samples* are strongly influenced by the initial configuration, so discarding them increases the speed of convergence towards the stationary distribution.

Let us now consider a multitarget setting where $\mathbf{x}_k = [\mathbf{x}_{k,1}, ..., \mathbf{x}_{k,M}]$ is the multitarget base state vector, with $\mathbf{x}_{k,j}$ being the state vector of the j^{th} target at time k. Within the Bayesian estimation framework, the target distribution π is chosen as the approximate posterior distribution,

$$\pi(\mathbf{x}_k^{(i-1)}) = p(\mathbf{x}_k^{(i-1)}|\mathbf{Z}_k) \propto p(\mathbf{x}_k^{(i-1)})\ p(\mathbf{Z}_k|\mathbf{x}_k^{(i-1)}). \quad (9)$$

The acceptance ratio is then given by:

$$\alpha = \frac{p(\mathbf{x}')\ p(\mathbf{Z}_k|\mathbf{x}')\ q(\mathbf{x}_k^{(i-1)}|\mathbf{X}')}{p(\mathbf{x}_k^{(i-1)})\ p(\mathbf{Z}_k|\mathbf{x}_k^{(i-1)})\ q(\mathbf{x}'|\mathbf{x}_k^{(i-1)})}. \quad (10)$$

A suitably designed proposal distribution is fundamental to guarantee a well mixing Markov chain. The efficiency of the M-H algorithm is strongly influenced by such choice. In particular, this problem becomes central when dealing with high dimensional and interdependent state vectors.

Such distributions should both be easy to sample from and allow the Markov chain to explore all the high density regions of $\mathcal{X}$ under π freely. However, the design of such efficient proposal distributions might be problematic. Instead, local strategies, focusing on some of the subcomponents of π, $\big($e.g. propose to move one target at each time, $\mathbf{x}'_j \sim q(\mathbf{x}'_j|\mathbf{x}_{k,j}^{(i-1)})\big)$ are often used, to break up the original sampling problem into simpler ones. Nevertheless, these can be prone to poor performance, as local strategies inevitably ignore some of the global features of π.

In summary, Markov Chain Monte Carlo (MCMC) methods are convenient and flexible, but they require to meet the following conditions :

- A sufficiently long burn-in time to allow convergence.
- Enough simulation draws for a suitably accurate inference for estimating the distribution π.

Note however that the convergence of MCMC algorithms strongly depends on whether the model actually fits the data and, in particular, the quality of the proposal distribution. Therefore convergence problems are often related to modeling issues.

B. Basic of Particle Filtering

A Particle Filter is known for its ability to tackle nonlinear and non-Gaussian problems. The detailed PF algorithm, as introduced in [17], is described below. At time step $k-1$, the posterior pdf $p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$ is approximated by a set of particles $\{\mathbf{x}_{k-1}^{(i)}\}_{i=1}^{N_p}$ with associated weights $\{w_{k-1}^{(i)}\}_{i=1}^{N_p}$. Each particle $\mathbf{x}_{k-1}^{(i)}$ is passed through the system dynamics eq.(1) to obtain $\{\hat{\mathbf{x}}_k^{(i)}\}_{i=1}^{N_p}$, the predicted particles at time step k. Once the observation data $\mathbf{z}_k$ is received, the importance weight of each predicted particle can be evaluated according to the weight equation reduces to:

$$\tilde{w}_k^{(i)} = p(\mathbf{z}_k|\hat{\mathbf{x}}_k^{(i)})$$

Then, after normalization of the weights, the a posteriori density $p(\mathbf{x}_k|\mathbf{Z}_k)$ at time step k can be approximated by the empirical distribution as :

$$\hat{p}(\mathbf{x}_k|\mathbf{Z}_k) := \sum_{i=1}^{N_p} w_k^{(i)}\ \delta_{\hat{\mathbf{x}}_k^{(i)}}(\mathbf{x}_k), \quad (11)$$

where $\delta_{\hat{\mathbf{x}}_k^{(i)}}(.)$ denotes the delta-Dirac mass located at $\hat{\mathbf{x}}_k^{(i)}$. See [18] for a proof and [19] for more detail and an overview of convergence results.

Finally, a *resampling step* is used to prevent the variance of the particle weights to increase over time. Although the resampling step reduces the effects of the degeneracy problem, it introduces other practical problems. First, it limits *parallel processing* since all particles have to be combined. Second, by resampling the posterior distribution, the heavily weighted particles are statistically selected several times. This leads to a loss of diversity among the particles, the *sample impoverishment* problem. In the extreme case, all the particles collapse to the same location. The sample impoverishment leads to failure in tracking since less diverse particles are used to represent the uncertain dynamics of the moving object. Both problems can effectively be handled by using a MCMC method implemented by e.g. the M-H algorithm.

C. Particle Markov chain Monte Carlo (PMCMC)

Several algorithms combining MCMC and SMC approaches have already been proposed in the literature [7], [20], [21]. Recently, Andrieu, Doucet and Holenstein [8] introduced a general framework, known as Particle MCMC (PMCMC), which uses a PF to construct proposal kernels for an MCMC sampler. PMCMC algorithms can be thought of as natural approximations to standard and *idealized* MCMC algorithms which cannot be implemented in practice. This framework provides three powerful methods for joint Bayesian state and parameter inference for nonlinear, non-Gaussian state-space models. These methods are referred to as Particle Independent Metropolis-Hastings (PIMH), Particle Marginal Metropolis-Hastings (PMMH) and Particle Gibbs (PG). In particular, the PMMH algorithm aims an exact approximation of a *Marginal Metropolis-Hastings* (MMH) update. The implementation of the PMMH scheme requires an SMC approximation targeting $p(\mathbf{x}_{0:k}|\mathbf{Z}_k, \theta)$ and the filter estimate of the marginal likelihood $p(\mathbf{Z}_k|\theta)$ with $\theta \in \Theta$ some static parameter. Full details of the PMMH scheme including a proof establishing that the method leaves the full joint posterior density $p(\theta, \mathbf{x}_{0:k}|\mathbf{Z}_k)$ invariant can be found in [8].

However, note that to obtain reasonable mixing of the resulting MCMC kernels, it was reported in [8] that a fairly high number of particles was required in the SMC scheme. Since every iteration of the PMMH scheme requires a run of a PF, a lot of computational resources are needed. In fact, if a PF algorithm using N_p particles is applied at each iteration, then the computational complexity of the PMCMC algorithm is $O(N_p\ M\ N)$, where N is the number of MCMC iterations and $M = dim(\mathbf{x}_k)$. In the following section we will propose

an alternative algorithm, which is much more robust to a low number of particles, and well suited to deal with tracking problems.

IV. Interacting Population-based MCMC-PF

In this section, we will present the Interacting Population-based MCMC-PF (IP-MCMC-PF) algorithm. We will give theoretical justifications for our approach and discuss the improvements attainable by using a fully parallelizable PF.

A. Justifications behind the IP-MCMC-PF algorithm

There are two basic ideas behind the proposed algorithm :

(1) Reliable statistical inferences for the target distribution are required. For this purpose we run *multiple MCMC sampling chains* each starting from different seeds, in parallel. A single possibly time varying transition kernel is used for all parallel chains. The only difference is the region of the space explored by each chain. The simulations from the chains are spread across various high probability regions of the target distribution. After a sufficiently long burn-in period, each chain reaches the stationary distribution. This means that, once enough stationary simulations have been drawn, mixing all sets of draws provides a good estimate of the target distribution.

(2) Rapid mixing within each MCMC chain is required. We want to speed up *the MCMC convergence rate* for minimizing the *burn-in* period. For that the history from all the chains is used to adapt the kernel and therefore to guide any particular population member in the exploration of the state space toward regions of higher probability. Thus more global moves (than independent MCMC chains) can be constructed. This ensures the *least correlation* among a single particle's history states resulting in faster mixing within each MCMC chain.

Furthermore the proposed implementation resorts to within-chain analysis to monitor stationarity and between/within chains comparisons to monitor mixing. Combined, stationarity and mixing lead to the convergence of the set of MCMC chains.

B. IP-MCMC-PF algorithm and challenges of implementation

In the proposed IP-MCMC-PF algorithm, the posterior distribution over target states at time $k-1$ is represented by a set of N_p particles $\{\mathbf{x}_{k-1}^{(i)}\}_{i=1}^{N_p}$. Each particle contains the joint state of M partitions corresponding to the different targets. We refer to $\mathbf{x}_{k-1,j}^{(i)}$ as the states of the j^{th} partition of the particle i at time step $k-1$. The particle state vector is given by :

$$\mathbf{x}_{k-1}^{(i)} = \left[\mathbf{x}_{k-1,1}^{(i)}, \ldots, \mathbf{x}_{k-1,M}^{(i)}\right].$$

Let us denote N_{MCMC} the optimal number of MCMC chains that can operate in parallel. The parameter N_{MCMC} depends on firstly, the experiment setup (targets in track and modelling complexity) and secondly, the system specification (parallel processing potential, memory resources). At each time step k, N_{MCMC} particles are randomly selected from $\{\mathbf{x}_{k-1}^{(i)}\}_{i=1}^{N_p}$. These particles are then propagated based on the dynamic model eq.(1) to obtain the predicted random subset of particles $\{\hat{\mathbf{y}}_s^{(0)}\}_{s=1}^{N_{\text{MCMC}}}$. The predicted particles are then chosen as the starting points in the MCMC sampling procedure.

The MCMC sampling procedure consists to run N_{MCMC} M-H samplers in parallel. Each sampler s is initiated with the configuration $\{\hat{\mathbf{y}}_s^{(0)}; \ell_s^{(0)}\}$ and is used to generate a set of N samples from $p(\mathbf{x}_k|\mathbf{Z}_k)$. At each MCMC iteration $n_{\text{M-H}}$, a partition j of a random particle i picked out of $\{\mathbf{x}_{k-1}^{(i)}\}_{i=1}^{N_p}$ is randomly selected. Given this partition $\mathbf{x}_{k-1,j}^{(i)}$, a partition move $\mathbf{y}'$ is proposed. This move seeks to update the single partition $\mathbf{x}_{k-1,j}^{(i)}$ via a Markov kernel that describes the state dynamic eq.(1). This move, also known as Mutation (Genetic Algorithm see [22]), allows for local exploration of the state space, as well as ensuring the required irreducibility of the Markov chain. Then, given $\{\hat{\mathbf{y}}_s^{(n_{\text{M-H}}-1)}; \ell_s^{(n_{\text{M-H}}-1)}\}$ the previous configuration, a new configuration $\{\tilde{\mathbf{y}}_s^{(n_{\text{M-H}})}; \tilde{\ell}_s^{(n_{\text{M-H}})}\}$ is obtained by replacing the partition j by $\mathbf{y}'$ and updating the joint log-likelihood. This new configuration is proposed and accepted as the next realization from the chain s with a probability

$$0 < \min\left(1, \alpha(\hat{\mathbf{y}}_s^{(n_{\text{M-H}}-1)}, \tilde{\mathbf{y}}_s^{(n_{\text{M-H}})})\right) < 1.$$

The acceptance ratio only depends on the likelihoods and is given by :

$$\alpha(\hat{\mathbf{y}}_s^{(n_{\text{M-H}}-1)}, \tilde{\mathbf{y}}_s^{(n_{\text{M-H}})}) = \frac{\tilde{\ell}_s^{(n_{\text{M-H}})}}{\ell_s^{(n_{\text{M-H}}-1)}}. \tag{12}$$

Note that such M-H move can be seen as an Exchange move (Genetic Algorithm see [22]). It can be proven that information exchange between MCMC chains targeting close distributions does not effect convergence properties [16]. Furthermore, in the proposed procedure, the MCMC sampling chains swap information within the M-H samplers without effecting the parallel processing. At the end of the procedure, the initial burn-in simulations, which are strongly influenced by starting values, are discarded, while the remaining samples $\{\hat{\mathbf{y}}_s^{(n_{\text{M-H}})}\}_{n_{\text{M-H}}=B+1}^{B+N}$ are stored as new particles to summarize the target distribution at time step k.

In the proposed algorithm, a validation test is performed to check the convergence on the basis of Gelman and Rubin [23] diagnostic. The potential scale reduction factor (PSRF) $\hat{\mathbf{R}}$ can be used for such test. In fact, $\hat{\mathbf{R}}$, defined as

$$\hat{\mathbf{R}} = \sqrt{\bar{\mathbf{C}}/\bar{\mathbf{W}}}, \tag{13}$$

with $\bar{\mathbf{C}}$ the empirical variance from all chains combined and $\bar{\mathbf{W}}$, the mean of the empirical within-chain variances, provides an ANOVA-like comparison of the within-chain and between-chain variances, i.e.

- $\hat{\mathbf{R}} = 1$, then the MCMC chains have converged within the burn-in period.
- $\hat{\mathbf{R}} > 1$, then all the MCMC chains have not fully mixed and that further simulation might increase the precision of inferences.

If the PSRF is close to 1, we can conclude that each of the N_{MCMC} sets of N simulated observations is close to the target distribution. In the proposed algorithm the parallel chains are considered well-mixed when $\hat{\mathbf{R}}$ is less or equal than 1.1 (the PSRF is estimated with uncertainty because our MCMC chain lengths are finite).

Once the set of chains have reached approximate convergence, the MCMC sampling chains outputs, i.e. the N_{MCMC} sets of simulations, mixed all together give the new set of particles $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N_p}$ which approximates the target distribution $p(\mathbf{x}_k|\mathbf{Z}_k)$.

The proposed IP-MCMC-PF algorithm is summarized in Algorithm 1. Then, a pseudo-code description of the MCMC sampling procedure is given by Algorithm 2.

Algorithm 1: IP-MCMC-PF algorithm

input : $\{\mathbf{x}_{k-1}^{(i)}\}_{i=1}^{N_p}$ and a new measurement, $\mathbf{z}_k$.
output: $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N_p}$.

1 - *Initialisation starting points:*
Select a random subset $\{\mathbf{y}_s^{(0)}\}_{s=1}^{N_{\text{MCMC}}} \subset \{\mathbf{x}_{k-1}^{(i)}\}_{i=1}^{N_p}$

while $s \leftarrow 1$ **to** N_{MCMC} **do**
  Predict a new particle state at time k:
  while $j \leftarrow 1$ **to** M **do**
    $\hat{\mathbf{y}}_{s,j}^{(0)} = f_k(\mathbf{y}_{s,j}^{(0)}) + \mathbf{v}_{k-1,j}^{(i)}$.
  end
  Compute the associated joint log-likelihood:
  $\ell_s^{(0)} = \log p(\mathbf{Z}_k|\hat{\mathbf{y}}_s^{(0)})$.
end
Store the new states in a cache $\{\hat{\mathbf{y}}_s^{(0)}; \ell_s^{(0)}\}_{s=1}^{N_{\text{MCMC}}}$.

2 - *MCMC sampling procedure:*
Run N_{MCMC} Metropolis-Hastings samplers in parallel (Algorithm 2) to obtain $\{\mathbf{y}_s^{(i)}\}$ where $i = 1, \ldots, N$ and $s = 1, \ldots, N_{\text{MCMC}}$.

3 - *Checking convergence:*
Compute $\hat{\mathbf{R}}$: (eq. 13).

if $\hat{\mathbf{R}} \geq 1.1$ **then**
  Run the chains out longer to improve convergence to the stationary distribution.
else
  Mix all the N_{MCMC} set of simulations together to obtain $\{\mathbf{x}_k^{(i)}\}_{i=1}^{N_p}$.
end

Algorithm 2: MCMC sampling procedure

input : $\{\hat{\mathbf{y}}_s^{(0)}; \ell_s^{(0)}\}_{s=1}^{N_{\text{MCMC}}}$ initial configurations, N_p required number of samples, B number of burn-in samples.
output: $\{\mathbf{y}_s^{(i)}\}$ where $s = 1, \ldots, N_{\text{MCMC}}$ and $i = 1, \ldots, N$ with $N := N_p/N_{\text{MCMC}}$.

while $s \leftarrow 1$ **to** N_{MCMC} **do**
  for $n_{M\text{-}H} \leftarrow 1$ **to** $B + N$ **do**
    1 - *Proposal of move:*
    Randomly select a partition $\mathbf{x}_{k-1,j}^{(i)}$ out of $\{\mathbf{x}_{k-1}^{(i)}\}_{i=1}^{N_p}$;
    Given $\mathbf{x}_{k-1,j}^{(i)}$, draw $\mathbf{y}' = f_k(\mathbf{x}_{k-1,j}^{(i)}) + \mathbf{v}_{k-1,j}^{(i)}$;
    Propose $\tilde{\mathbf{y}}_s^{(n_{\text{M-H}})}$ by $\mathbf{y}' \rightarrow \hat{\mathbf{y}}_s^{(n_{\text{M-H}}-1)}$;
    Compute the joint log-likelihood:
    $\tilde{\ell}_s^{(n_{\text{M-H}})} = \log p(\mathbf{Z}_k|\tilde{\mathbf{y}}_s^{(n_{\text{M-H}})})$.
    2 - *M-H acceptance:*
    Sample $u \sim \mathcal{U}_{[0,1]}$, $\mathcal{U}_{[0,1]}$ a uniform distribution in $[0, 1]$;
    Compute the M-H acceptance probability $\alpha(\hat{\mathbf{y}}_s^{(n_{\text{M-H}}-1)}, \tilde{\mathbf{y}}_s^{(n_{\text{M-H}})})$: (eq. 12).
    if $u \leq \min(1, \alpha)$ **then**
      Accept move:
      $\{\hat{\mathbf{y}}_s^{(n_{\text{M-H}})}; \ell_s^{(n_{\text{M-H}})}\} = \{\tilde{\mathbf{y}}_s^{(n_{\text{M-H}})}; \tilde{\ell}_s^{(n_{\text{M-H}})}\}$,
    else
      Reject move:
      $\{\hat{\mathbf{y}}_s^{(n_{\text{M-H}})}; \ell_s^{(n_{\text{M-H}})}\} = \{\hat{\mathbf{y}}_s^{(n_{\text{M-H}}-1)}; \ell_s^{(n_{\text{M-H}}-1)}\}$.
    end
  end
  Discard B initial burn-in samples, store the remaining samples $\hat{\mathbf{y}}_s^{(B+1:B+N)} \rightarrow \{\mathbf{y}_s^{(i)}\}_{i=1}^{N}$.
end

V. System Setup and TBD Problem Formulation

In this section, we will describe the system dynamic model and the measurement model.
Let us denote by $\mathbf{x}_{k,j}$ the vector describing the state of the j^{th} target ($j \in \{1, M\}$) at time step k written as:

$$\mathbf{x}_{k,j} = [\mathbf{s}_{k,j}, \rho_{k,j}]^T \tag{14}$$

where, $\mathbf{s}_{k,j}$ represents the position and velocity of the j^{th} target in Cartesian coordinates and $\rho_{k,j}$ is the unknown modulus of the target complex amplitude. The state vectors $(\mathbf{x}_{k,j})_{j\in\{1,M\}}$ can be concatenated into the multitarget state vector $\mathbf{x}_k$.

A. Dynamic model

To model the dynamics of the targets we adopt a nearly constant velocity model to describe object position and velocity in a Cartesian frame, see e.g. [4], and a random walk model for object amplitude $\rho_{k,j}$. The model uncertainty is handled by the process noise $\mathbf{v}_{k,j}$, which is assumed to be standard white Gaussian noise with covariance G. Under these assumptions, the corresponding state-space model, for $\mathbf{x}_{k,j}$, the state vector associated to the j^{th} target, is given by:

$$\mathbf{x}_{k+1,j} = F\mathbf{x}_{k,j} + \mathbf{v}_{k,j}, \tag{15}$$

where F represents a transition matrix with a constant sampling time T such as:

$$\begin{aligned} F &= \text{diag}(F_1, F_1, 1), \\ \text{where, } F_1 &= \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}. \end{aligned} \tag{16}$$

and G, the covariance process noise, is given by:

$$\begin{aligned} G &= \text{diag}\left(a_x G_1, a_y G_1, a_\rho T\right), \\ \text{where, } G_1 &= \begin{bmatrix} \frac{T^3}{3} & \frac{T^2}{2} \\ \frac{T^2}{2} & T \end{bmatrix}, \end{aligned} \tag{17}$$

where $a_x = a_y$ and a_ρ denote the level of process noise in object motion and amplitude, respectively. This model correctly approximates small accelerations in the object motion and fluctuations in the object amplitude.

B. Measurement model

At discrete instants k, the radar system positioned at the Cartesian origin collects a noisy signal. Each measurement $\mathbf{z}_k$ consists of $N_r \times N_d \times N_b$ reflected power measurements $\mathbf{z}_k^{lmn}$, where N_r, N_d and N_b are the number of range, doppler and bearing cells. The power measurement per range-doppler-bearing cell is defined by :

$$\mathbf{z}_k^{lmn} = |\mathbf{z}_{\rho,k}^{lmn}|^2, k \in \mathbb{N}. \tag{18}$$

where $\mathbf{z}_{\rho,k}^{lmn}$ is the complex envelope data of the target described by the following nonlinear equation, see [4]:

$$\mathbf{z}_{\rho,k}^{lmn} = \sum_{j=1}^{M} \rho_{k,j} e^{i\varphi_k} h^{lmn}(\mathbf{s}_{k,j}) + \mathbf{w}_k^{lmn}, \quad \varphi_k \in (0, 2\pi) \tag{19}$$

where $h^{lmn}(\mathbf{s}_{k,j})$ is the reflection form of the j^{th} target, that for every range-doppler-bearing cell is defined by:

$$\begin{aligned} &h^{lmn}(\mathbf{x}_{k,j}) := e^{-\frac{(r_l - r_{k,j})^2}{2R} - \frac{(d_m - d_{k,j})^2}{2D} - \frac{(b_n - b_{k,j})^2}{2B}}, \\ &l = 1, \ldots, N_r, \ m = 1, \ldots, N_d, \ n = 1, \ldots, N_b \text{ and } k \in \mathbb{N} \end{aligned} \tag{20}$$

where the relationship between the measurement space and the target space can be established as:

$$r = \sqrt{x^2 + y^2}, \ d = \frac{(x\ v_x + y\ v_y)}{\sqrt{x^2 + y^2}} \text{ and } b = \arctan\left(\frac{y}{x}\right). \tag{21}$$

R, D and B are related to the resolutions in range, doppler and bearing. $\mathbf{w}_k^{lmn}$ are independent samples of a complex white Gaussian noise with variance $2\sigma_w^2$, (*i.e.* the real and imaginary components are assumed to be independent, zero-mean white Gaussian with the same variance σ_w^2).

C. TBD Problem Formulation

Consider the system represented by the equations (15), (18) and (19). Assume that the set of measurements collected up to the current time is denoted by $\mathbf{Z}_k = \{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$. The filtering problem can be formulated as finding the a posteriori distribution of the joint state $\mathbf{x}_k$ for all possible numbers of targets conditioned on all past measurements $\mathbf{Z}_k$.

VI. Simulation setup and Results

In this section the multitarget TBD filtering problem has been recursively solved by running the IP-MCMC-PF on the power measurements. We investigate through simulations the improvements provided by interacting population-based simulation over standard SMC and MCMC methods. Simulations are performed with a typical ground radar scenario (one or several targets move in the x-y plane at unknown constant velocity). The radar parameters and the targets settings used in simulation are reported in Table I.

TABLE I
PARAMETERS USED IN SIMULATION

Radar parameters	Radar sampling time	$T = 1\ [sec]$
	Beamwidth in bearing	$B = \frac{\pi}{180}\ [rad]$
	Range-quant size	$R = 10\ [m]$
	Doppler-bin size	$D = 2\ [m.s^{-1}]$
Targets settings	Bearing area	$b_0 \sim U\ [-6, 6]\ [^\circ]$
	Range area	$r_0 \sim U\ [2700, 2900]\ [m]$
	Radial velocity area	$\lvert v_0 \rvert \sim U\ [0, 15]\ [m.s^{-1}]$
	Acceleration in turn	$a_0 \sim U\ [0, 1]\ [m.s^{-2}]$
	Angle of turn	$\varphi_0 \sim U\ [-3, 3]\ [^\circ]$
	SNR	$SNR \sim U\ [9, 20]\ [dB]$
	Maximum Target Speed	$v_{max} = 40\ [m.s^{-1}]$

Throughout the simulation the targets move according to the initial conditions along a constant velocity trajectory. As recursive track filter we have used a filter tracking the two-dimensional position and velocity with a piecewise constant white acceleration model defined in eq.(15) for the target dynamics, where we have set the standard deviation of the random accelerations to 1 $m.s^{-2}$. All the targets remain within the surveillance region until the last time step.

We first consider a simplified version of the Track-Before-Detect (TBD) problem, where a single-target moves within the surveillance area and never leaves the sensor Field-of-View (FoV). In particular, we focus on the scenario depicted in fig.1.

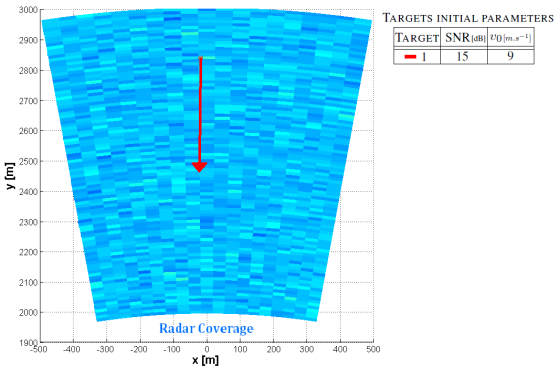


Fig. 1. True track in the x-y plane. Target move with near-constant velocity along the paths shown.

Three algorithms : the conventional (a) SIR PF (alg. see [4]), the recently developed (b) PMMH (alg. see [7]) and the proposed (c) IP-MCMC-PF (algo. 1) are considered.

In fig. 2 we report the empirical posterior PDF and the MV estimate of eq.(6) over a single trial.

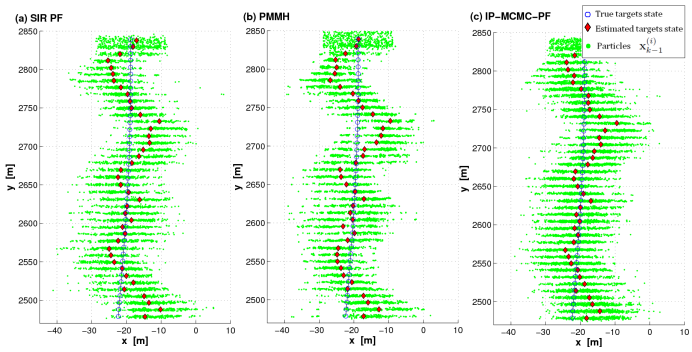


Fig. 2. Empirical posterior distribution and conditional mean over a single trial. Each filter uses 500 particles. It is immediate to verify that the proposed method reduces the intrinsic uncertainty in the posterior PDF when compared to SIR PF.

In fig. 3 we compare the tracking accuracy, measured in terms of the Root Mean Square Error (RMSE), over 100 Monte-Carlo simulations.

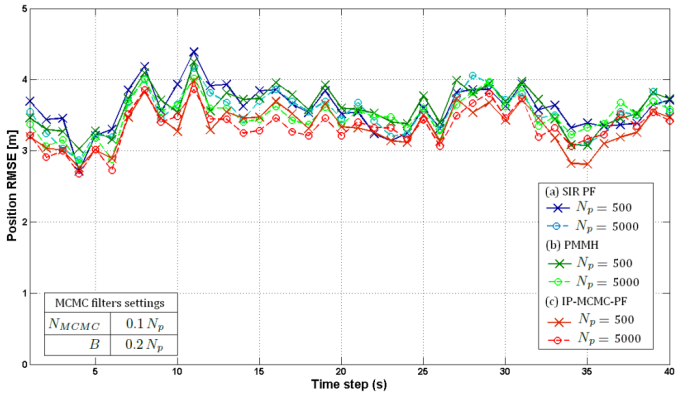
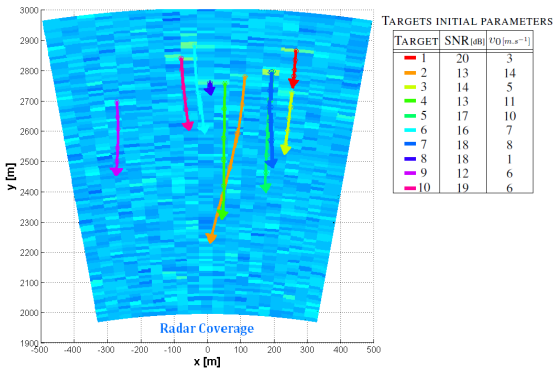


Fig. 3. Position RMSE over time for the (a) SIR PF, (b) PMMH and (c) IP-MCMC-PF. The proposed algorithm matches the tracking performance of the SIR PF and the PMMH.

We then consider a typical scenario with 10 random targets depicted in fig. 4.



TARGETS INITIAL PARAMETERS

TARGET	SNR (dB)	ν_0 [m.s^{-1}]
1	20	3
2	13	14
3	14	5
4	13	11
5	17	10
6	16	7
7	18	8
8	18	1
9	12	6
10	19	6

Fig. 4. True tracks in the x-y plane. Targets move with near-constant velocity along the paths shown.

The tracks (the means of each set of target samples) produced by the IP-MCMC-PF algorithm over 100 Monte Carlo runs are shown in fig. 5.

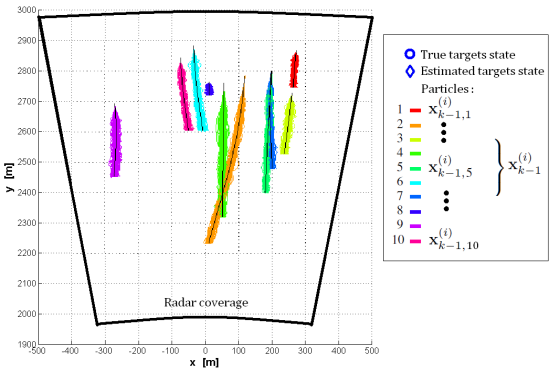


Fig. 5. Tracking trajectory. $N_p = 500$ particles, $N_{\text{MCMC}} = 50$ MCMC chains and $B = 100$ burn-in samples.

We then compared the tracking accuracy, the computational cost and the robustness of the three algorithms over 100 Monte-Carlo simulations. The tracking performance is measured in terms of the Root Mean Square Error (RMSE). The results are reported in fig. 6.

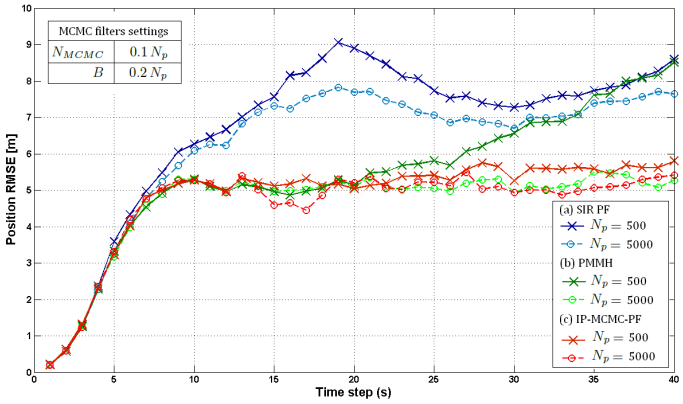


Fig. 6. Position RMSE over time for the (a) SIR PF, (b) PMMH and (c) IP-MCMC-PF.

The performance of the methods is also compared via the average RMSE, the tracking loss rate (TLR) and the average CPU time (avg.CPU), which are listed in Table II.

TABLE II
PERFORMANCE COMPARISON AVERAGED OVER 100 MC RUNS

Algorithm	N_p	**RMSE**	**TLR**	**avg.CPU**
(a) SIR PF	500	6.74 [m]	6%	0.5
	5000	6.23 [m]	3%	3.8
(b) PMMH	500	5.42 [m]	1%	0.6
	5000	4.65 [m]	0%	2.7
(c) IP-MCMC-PF	500	4.84 [m]	0%	0.4
	5000	4.61 [m]	0%	0.69

The tracking loss rate (TLR) is defined as the ratio of the number of simulations, in which the target is lost in track (Maximum Position error $> 20[m]$) , to the total number of simulations carried out. The average CPU time (avg.CPU) is the CPU time needed to execute one time step in MATLAB R2011b (win64) on an Intel Core i7 (Nehalem microarchitecture) operating under Windows 7 Ultimate (Version 6.1). All the important specifications and features are reported in Table III.

TABLE III
SPECIFICATIONS & FEATURES

Processor	Type Model Frequency Cores	Intel(R) Core(TM) i7 920 4.2 GHz 4 (8 threads)
Memory	DRAM Type	6144 MB (3 x 2048 DDR3-SDRAM) PC3-8500 (533 MHz)

VII. CONCLUSION

In this work, we propose an Interacting Population based Monte Carlo Markov Chain based PF (IP-MCMC-PF) that solves the multitarget tracking problem, which is fully parallelizable. In particular, the proposed algorithm matches the tracking performance of the multitarget SIR PF, while allowing for a dramatic reduction of the computational time.

Furthermore, we demonstrate through simulations that the proposed IP-MCMC-PF provides better tracking performance than the conventional MCMC based particle filter. This holds in terms of track accuracy and robustness to a low number of particles. In fact, sampling particles from the target posterior distribution via interacting population-based simulation avoids sample impoverishment and accelerates the MCMC convergence rate.

In a forthcoming work, we will therefore focus on more complex scenarios where targets may enter or leave the observation area. Furthermore, as future research we will investigate the applicability of well known MCMC techniques to additionally reduce the computational burden.

ACKNOWLEDGMENT

The research leading to these results has received funding from the EU's Seventh Framework Programme under grant agreement n^{o}238710.

The research has been carried out in the MC IMPULSE project: https://mcimpulse.isy.liu.se.

REFERENCES

[1] J. Vermaak, S. Godsill, and P. Perez, "Monte carlo filtering for multi-target tracking and data association," *IEEE Tr. Aerospace and Electronic Systems*, vol. 41, no. 1, pp. 309–332, January 2005.

[2] D. J. Salmond and H. Birch, "A particle filter for track-before-detect," *Proc. of the 2001 American Control Conference,*, pp. 3755–3760, 2001.

[3] C. Kreucher, K. Kastella, and A. Hero, "Tracking multiple targets using a particle filter representation of the joint multitarget probability density," *Proceedings of the SPIE*, vol. 5204, pp. 258–269, 2003.

[4] Y. Boers and J. N. Driessen, "Multitarget particle filter track-before-detect application," *IEE Proc. on Radar, Sonar and Navigation*, vol. 151, pp. 351–357, 2004.

[5] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: fifteen years later," *In Handbook of Nonlinear Filtering, University Press*, 2009.

[6] M. Orton and W. Fitzgerald, "Bayesian approach to tracking multiple targets using sensor arrays and particle filters," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 216–223, February.

[7] C. Andrieu, A. Doucet, and R. Holenstein, "Particle markov chain monte carlo for efficient numerical simulation," *Monte Carlo and Quasi-Monte Carlo Methods*, pp. 45–60, Spinger-Verlag Berlin Heidelberg 2008.

[8] ——, "Particle markov chain monte carlo methods," *Journal of the Royal Statistical Society: Series B*, vol. 72, no. 3, pp. 269–342, 2010.

[9] A. E. Gelfand and A. F. M. Smith, "Sampling-based approaches to calculaing marginal densities," *Journal of the American Statistical Association*, vol. 85, pp. 398–409, 1990.

[10] G. Casella and E. I. George, "Explaining the gibbs sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, August 1992.

[11] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equations of state calculations by fast computing machines," *J. Chemical Physics*, vol. 21, pp. 1087–1091, 1953.

[12] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, pp. 97–109, 1970.

[13] S. Chib and E. Greenberg, "Understanding the metropolis-hastings algorithm," *The American Statistician*, vol. 49, pp. 327–335, 1995.

[14] S. Park, J. P. Hwang, E. Kim, and H. J. Kang, "A new evolutionary particle filter for the prevention of sample impoverishment," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 801–809, August 2009.

[15] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.

[16] C. Andrieu and E. Moulines, "On the ergodicity properties of some adaptive mcmc algorithms," *Annals of Applied Probability*, vol. 16, no. 3, pp. 1462–1505, 2006.

[17] N. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation problems," *IEE Proc. on Radar and Signal Processing*, vol. 140(2), pp. 107–113, 1993.

[18] A. F. M. Smith and A. E. Gelfand, "Bayesian statistics without tears: A sampling-resampling perspective," *The American Statistician*, vol. 46, pp. 84–88, 1992.

[19] X.-L. Hu, T. Schon, and L. Ljung, "A basic convergence result for particle filtering," *IEEE Transactions on Signal Processing*, vol. 56, no. 4, pp. 1337–1348, April 2008.

[20] Z. Khan, T. Balch, and F. Dellaert, "Mcmc-based particle filtering for tracking a variable number of interacting targets," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 27, no. 11, pp. 1805–1819, 2005.

[21] C. Andrieu and G. O. Roberts, "The pseudo-marginal approach for efficient monte carlo computations," *Annals of Statistics*, vol. 37, no. 2, pp. 697–725, 2009.

[22] F. Liang and W. H. Wong, "Real parameter evolutionary monte carlo with applications to bayesian mixture models," *Journal of the American Statistical Association*, vol. 96, pp. 653–666, 2001.

[23] A. Gelman and D. B. Rubin, "Inference from iterative simulation using multiple sequences (with discussion)," *Statistical Science*, vol. 7, pp. 457–511, 1992.