Rao-Blackwellized Particle Filters with Out-of-Sequence Measurement Processing

Berntorp, Karl; Robertsson, Anders; Årzén, Karl-Erik

Published in:
IEEE Transactions on Signal Processing

DOI:
10.1109/TSP.2014.2365763

2014

Rao-Blackwellized Particle Filters with Out-of-Sequence Measurement Processing

Karl Berntorp*, Anders Robertsson, and Karl-Erik Årzén

***Abstract*—This paper addresses the out-of-sequence measurement (OOSM) problem for mixed linear/nonlinear state-space models, which is a class of nonlinear models with a tractable, conditionally linear substructure. We develop two novel algorithms that utilize the linear substructure. The first algorithm effectively employs the Rao-Blackwellized particle filtering framework for updating with the OOSMs, and is based on storing only a subset of the particles and their weights over an arbitrary, predefined interval. The second algorithm adapts a backward simulation approach to update with the delayed (out-of-sequence) measurements, resulting in superior tracking performance. Extensive simulation studies show the efficacy of our approaches in terms of computation time and tracking performance. Both algorithms yield estimation improvements when compared with recent particle filter algorithms for OOSM processing; in the considered examples they achieve up to 10% enhancements in estimation accuracy. In some cases the proposed algorithms even deliver accuracy that is similar to the lower performance bounds. Because the considered setup is common in various estimation scenarios, the developed algorithms enable improvements in different types of filtering applications.**

***Index Terms*—Tracking, particle filtering, out-of-sequence measurement (OOSM), Rao-Blackwellization**

I. Introduction

OUT-of-sequence measurements (OOSMs) are measurements that arrive after more recent measurements have already been processed. Since the amount of sensors used in tracking is increasing, and since tracking is frequently performed using distributed sensor platforms, tracking systems increasingly often encounter OOSMs. Delayed measurements occur for several reasons, for example: data preprocessing, communication delays, and acoustic propagation resulting in different sensors observing the current state of the target at different times. An application where OOSMs are present is from the automotive sector, where network links cause transmission delays of radar sensors [1]. Another application is tracking of autonomous vehicles, where cameras have become increasingly important for giving spatial information. When cameras are used, the processing times of the vision algorithms often cause OOSMs, see [2], [3], [4]. Two papers that treat OOSM algorithms in mobile robotics are [5], [6], and [7] discusses OOSMs in passive target tracking in sensor networks. Neglecting the time delays of the measurements seems to be a common choice [8], but this means discarding information and may lead to inferior tracking performance. However, to make efficient use of the OOSMs in nonlinear tracking systems can be challenging.

This paper addresses the OOSM problem for a class of conditionally linear Gaussian state-space (CLGSS) models, known as mixed linear/nonlinear state-space models. In CLGSS models a linear Gaussian substructure is present. Hence, it is possible to use the Rao-Blackwellized particle filter (RBPF) for improved forward-filtering performance compared with the standard particle filter (PF) [9]. Mixed linear/nonlinear state-space models are common in, for example, target tracking, positioning, and navigation [9], [10]. Two occurences are when the system equations are almost linear and/or the measurement equations are highly nonlinear, see [11], [12] for examples.

Previous work has provided the exact Bayesian inference solution and its particle-filter implementation to the OOSM problem, see [13]. The major contributions of this paper are the Bayesian formulation when utilizing substructure in nonlinear state-space models and two RBPF solutions to the OOSM problem. In particular, by exploiting the conditionally linear Gaussian substructure present in the model class, we are able to provide improved tracking performance and favorable computational demands compared with current state-of-the-art OOSM algorithms. We present two alternative approaches. The first algorithm (SERBPF) can be regarded as a generalization of the storage-efficient particle filters reported in [14], with the derivations adapted to the mixed linear/nonlinear setting. Except for the measurements, the approach only stores a subset of the particles over a predefined time interval. These particles are then used in an additional forward filter to associate the current estimates with the OOSMs. Because of its computational simplicity, it is well suited for real-time applications. The second algorithm (RBOOSMBS) instead adapts a backward-simulation approach for the association task. This implies that when the number of particles and smoothing trajectories are sufficiently large, RBOOSMBS achieves close to optimal tracking performance at the OOSM arrival times.

A. Relations to Previous Work and Outline

An earlier version of parts of this work has been presented in [15]. This paper presents several extensions. Specifically, we allow for crosscorrelation between the process noise acting on the linear states and the process noise acting on the nonlinear states, improve RBOOSMBS in terms of computational complexity, and present a more complete and careful derivation

All authors were with the Department of Automatic Control, Lund University, Lund, Sweden at the time of this research (e-mail: firstname.lastname@control.lth.se). Karl Berntorp is now with Mitsubishi Electric Research Laboratories, Cambridge, MA. This research was not sponsored by Mitsubishi Electric or any of its subsidiaries. This work was supported by the Swedish Foundation for Strategic Research through the project ENGROSS, the LCCC Linnaeus Center, and the ELLIIT Excellence Center.

of both algorithms. Moreover, the results contain a third benchmark example in addition to an extensive evaluation and in-depth analysis of all the obtained results.

Sec. II contains an overview of related work, whereas Sec. III provides the problem statement and the notation. Sec. IV briefly describes particle filters and smoothers, both with and without utilizing substructure. We present the two proposed algorithms in Sec. V and assess their performance on three examples in Sec. VI. Sec. VII contains a discussion of the results and the applicability of the algorithms. Finally, we summarize and conclude the paper in Sec. VIII.

II. Related Work

How to incorporate OOSMs for linear-Gaussian systems has been thoroughly investigated during past decades. See [16] for an overview of initial work spanning to the late 1990's. Two suboptimal algorithms are given in [17], [18]. An algorithm that is optimal in the mean-square sense is found in [19]. However, the OOSM in [19] is assumed to be delayed less than one sample (the 1-step lag problem). Several solutions to the general l-step lag OOSM problem have been derived, see [20], [21], [22], [23], [24]. The differences between the papers are mainly the employed approaches and whether the OOSMs arrive one by one or are interleaved with other OOSMs.

Several algorithms for processing OOSMs using particle filters have been proposed. An approach where the measurement equation is allowed to be nonlinear was outlined in [25], [26]. The particle weights are first updated without the OOSM. They are then modified utilizing the OOSM in a Markov chain Monte Carlo smoothing step to overcome the problem of degeneracy in the particle filter. This approach stores the N particles for the last $l_{\max}$ time steps, where $l_{\max}$ is the predetermined maximum delay. Unfortunately, it also needs a linear state-transition model. Moreover, to retrodict (predict backwards) the state vector back to the OOSM time, [27] assumes an invertible state-transition matrix. The storage-efficient particle filter (SEPF), which handles nonlinear state-space models assuming white, Gaussian noise, was presented in [14]. SEPF is computationally fast and memory efficient, because it only stores and processes means and covariances through an extended Kalman smoother to update with the OOSMs. However, SEPF's performance sometimes suffers when the OOSMs introduce a large change in the estimated filtering distribution. In [28], the approach in [14] was extended with an approximate method for separating between informative and uninformative OOSMs, and this was further elaborated in [29] with an optimization-based algorithm. In [30], the algorithms from [14] and [23] were fused to enhance performance and further reduce storage requirements. An exact Bayesian solution and its corresponding particle-filter implementation, denoted A-PF, were derived in [13] for nonlinear models with white noise. One drawback with A-PF is that it is computationally expensive; for OOSMs that have larger delays than one sample, its complexity is $\mathcal{O}((l-1)N^3+N^2)$.

III. Problem Formulation

Throughout, $p(x_k|y_{m:k})$ denotes the conditional probability density function of the variable x at time $t_k \in \mathbb{R}$ conditioned on the variable y from time t_m to time t_k. We assume that the state vector $x_k \in \mathbb{R}^{n_x}$ can be partitioned into a linear part z_k and a nonlinear part η_k as $x_k = \begin{pmatrix} z_k^{\mathrm{T}} & \eta_k^{\mathrm{T}} \end{pmatrix}^{\mathrm{T}}$. Given mean vector μ and covariance matrix Υ, $\mathcal{N}(\mu,\Upsilon)$ and $\mathcal{N}(x|\mu,\Upsilon)$ stand for the Gaussian distribution and probability density function, respectively. The considered models are on the form

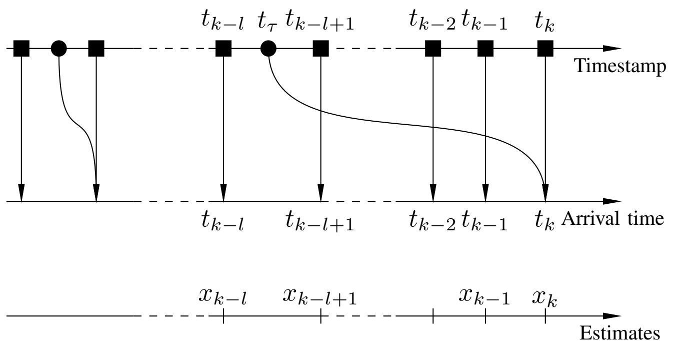

Fig. 1. An illustration of the l-step lag OOSM problem, where the circles denote OOSMs and squares refer to the in-sequence measurements. Measurements $y_{0:k}$ have arrived and been used in the estimation process to calculate x_k. Then, an OOSM y_τ arising from time t_τ arrives at time t_k, and is subsequently utilized to calculate an updated x_k.

$$z_{k+1} = f(\eta_k) + A(\eta_k)z_k + v_k^z(\eta_k), \tag{1a}$$
$$\eta_{k+1} = g(\eta_k) + B(\eta_k)z_k + v_k^\eta(\eta_k), \tag{1b}$$
$$y_k = h(\eta_k) + C(\eta_k)z_k + e_k(\eta_k), \tag{1c}$$

where the process noise $v_k(\eta_k) = \begin{pmatrix}(v_k^z(\eta_k))^{\mathrm{T}} & v_k^\eta(\eta_k))^{\mathrm{T}}\end{pmatrix}^{\mathrm{T}}$ and the measurement noise $e_k(\eta_k)$ are mutually independent, white, and Gaussian distributed according to

$$v_k(\eta_k) \sim \mathcal{N}\left(\begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix} Q_k^z(\eta_k) & Q_k^{z\eta}(\eta_k) \\ Q_k^{z\eta}(\eta_k)^{\mathrm{T}} & Q_k^\eta(\eta_k)\end{pmatrix}\right) \tag{2}$$

and $e_k(\eta_k) \sim \mathcal{N}(0, R_k(\eta_k))$, with $Q_k^\eta(\eta_k)$ and $R_k(\eta_k)$ invertible. Further, $f(\eta_k)$, $g(\eta_k)$, $h(\eta_k)$, the system matrices $A(\eta_k)$, $B(\eta_k)$, $C(\eta_k)$, and the covariance matrices all have a, possibly nonlinear, dependence on η_k. In the following, f_k means $f(\eta_k)$ for any of these. Note that (1a) is linear (affine) given η_k. For the OOSM filtering task, $\mathcal{Y}_k$ denotes the set of in-sequence measurements generated in the interval $[0,k]$. The symbol $\mathcal{Z}_k$ indicates the set of OOSMs generated in the interval $[0,k]$ available at time index k. Moreover, $y_{0:k}$ represents the set $\mathcal{Y}_k \cup \mathcal{Z}_{k-1}$. We discard measurements delayed more than $l_{\max}$ time steps, with $l_{\max}$ predefined, and assume that all sensors give detection of the correct targets each time. Hence, data association and clutter are beyond the scope of this paper [8], [31]. Further, $\hat{z}_{k|k} = \hat{z}_{k|k}(\eta_{0:k})$ is the linear state estimate given the trajectory $\eta_{0:k}$ and measurements $y_{0:k}$, and $P_{k|k} = P_{k|k}(\eta_{0:k})$ is its associated covariance.

Suppose that an estimate of the filtering posterior $p(z_k, \eta_k|y_{0:k})$ exists at time t_k, where z_k is conditioned on $\eta_{0:k}$. Assume that an OOSM $y_\tau \in \mathcal{Z}_k$ with timestamp $t_\tau \in [t_{k-l}, t_{k-l+1})$ arrives, see Fig. 1. The Rao-Blackwellized OOSM filtering task is to update the particle weights and linear estimates at time t_k with y_τ, that is, to obtain $p(x_k|y_{0:k}, y_\tau) = p(z_k, \eta_k|y_{0:k}, y_\tau)$.

IV. Background

This section briefly discusses particle filtering and smoothing, both for pure nonlinear and mixed linear/nonlinear state-space models. First, consider a standard Markov-process with the dynamics and measurement equation as

$$x_{k+1} = f(x_k) + v_k, \tag{3a}$$
$$y_k = h(x_k) + e_k, \tag{3b}$$

where $f(\cdot)$ and $h(\cdot)$ in general are nonlinear functions and v_k, e_k have known densities.

A. Particle Filtering

The Bayesian approach to filtering is to compute the posterior density $p(x_{0:k}|y_{0:k})$. When (3a)–(3b) are linear with Gaussian noise, the solution is analytic and given by the Kalman filter [32]. Most often, however, numerical approximations are required. Particle filters are sequential Monte-Carlo methods that represent the posterior density with a set of weighted particles [33], [34], [35], [36]. Each particle represents a state trajectory $x_{0:k}$, which results in the approximation

$$p(x_{0:k}|y_{0:k}) \approx \sum_{i=1}^{N} w_k^i \delta_{x_{0:k}^i}(x_{0:k}). \tag{4}$$

Here, $\delta(\cdot)$ is the Dirac delta function and w_k^i is the associated weight for the ith particle given the measurements $y_{0:k}$. The particle weights are typically updated as $w_k^i \propto p(y_k|x_k^i)w_{k-1}^i$. An approximation to the marginal (filtering) density is given by discarding $x_{0:k-1}$, yielding $p(x_k|y_{0:k})$. To avoid having a significant dependence on a few particles with large weights (i.e., particle depletion), a crucial resampling step is carried out, which provides an equally-weighted distribution.

To decrease the number of particles and the variance of the estimates, it is advantageous to exploit model structure. This is the idea behind Rao-Blackwellization, where the subset of the state space that allows for analytic expressions is marginalized out. The sampled state space is then smaller and it is therefore possible to use fewer particles [34]. If (3) has the same structure as (1), a Rao-Blackwellized particle filter (RBPF) can be used [9]. The enabler for this is the key factorization

$$p(z_k, \eta_{0:k}|y_{0:k}) = p(z_k|\eta_{0:k}, y_{0:k})p(\eta_{0:k}|y_{0:k}). \tag{5}$$

The second distribution in (5) is approximated by the particle filter. Given the nonlinear state trajectory, the first part in (5) is linear Gaussian. Thus, it can be estimated with constrained Kalman filters, one for each particle. The main difference compared with the standard Kalman filter consists of performing an extra measurement update using η_k. This implies that

$$p(z_k|\eta_{0:k}, y_{0:k}) = \mathcal{N}(z_k|\hat{z}_{k|k}, P_{k|k}). \tag{6}$$

By combining (4) and (6), an approximation to (5) is given by the Gaussian mixture

$$p(z_k, \eta_{0:k}|y_{0:k}) \approx \sum_{i=1}^{N} w_k^i \mathcal{N}(z_k|\hat{z}_{k|k}^i, P_{k|k}^i)\delta_{\eta_{0:k}^i}(\eta_{0:k}), \tag{7}$$

where the weight update resembles that of the standard particle filter. The filtering posterior $p(z_k, \eta_k|y_{0:k})$ can be found by marginalization of (7).

B. Particle Smoothing

Particle smoothers are used to form approximate solutions to the smoothing density $p(x_{0:T}|y_{0:T})$. The marginal and fixed-interval smoothing densities can be found by marginalization. Particle filters (4) can also be used as an approximation to the smoothing density [37]. However, because of the inherent depletion problem in PFs, this estimate is often degenerate for large time lags. Hence, particle smoothing is frequently approached with other algorithms, such as the forward filter/backward simulator (FFBS) [38]. The FFBS utilizes the sequential factorization

$$p(x_{0:T}|y_{0:T}) = p(x_T|y_{0:T}) \prod_{k=0}^{T-1} p(x_k|x_{k+1:T}, y_{0:T}). \tag{8}$$

Starting by sampling a state x'_T from the filtering approximation $p(x_T|y_{0:T})$ at time index T, the Markov property

$$p(x_k|x_{k+1:T}, y_{0:T}) \propto p(x_{k+1}|x_k)p(x_k|y_{0:k}) \tag{9}$$

is utilized to form the approximation

$$p(x_{T-1}|x'_T, y_{0:T}) \approx \sum_{i=1}^{N} w_{T-1|T}^i \delta_{x_{0:T-1}^i}(x_{0:T-1}),$$

where $w_{T-1|T}^i \propto w_{T-1}^i p(x'_T|x_{T-1}^i)$. Iterating backward, x_{T-1} is generated by sampling from the filtering distribution at time index $T-1$ with probability $w_{T-1|T}^i$. This recursion is performed until $k=0$ is reached, whereby an approximation to (8) can be created. For a more diverse approximation, the algorithm is repeated M times to yield

$$p(x_{0:T}|y_{0:T}) \approx \frac{1}{M}\sum_{j=1}^{M} \delta_{x_{0:T}^j}(x_{0:T}). \tag{10}$$

In Rao-Blackwellized particle smoothing (RBPS) [39], the density $p(z_k, \eta_k|y_{0:T})$, $k < T$, is approximated by

$$p(z_k, \eta_k|y_{0:T}) \approx \frac{1}{M}\sum_{j=1}^{M} \mathcal{N}(z_k|\hat{z}_{k|T}^j, P_{k|T}^j)\delta_{\eta_k^j}(\eta_k). \tag{11}$$

Conditionally linear state-space models such as (1) do not have the Markov property (9). Whole trajectories must therefore be sampled from the RBPF, to preserve Gaussianity. To compute (11), the RBPS draws one of the RBPF particles $\{\eta_{0:k}^i\}_{i=1}^N$ with probability $w_{k|T}^i$, and by discarding $\eta_{0:k-1}^i$ the trajectory is appended, yielding $\{\eta_k^i, \eta'_{k+1:T}\}$. This procedure is repeated for each $k = T-1, \dots, 0$, resulting in a backward trajectory that can be used to approximate $p(\eta_{0:T}|y_{0:T})$. Using Bayes rule on (9) gives that the backward kernel is approximated as

$$p(\eta_{0:k}|\eta_{k+1:T}, y_{0:T}) \approx \sum_{i=1}^{N} w_{k|T}^i \delta_{\eta_{0:k}^i}(\eta_{0:k}).$$

The smoothing weights $w_{k|T}^i$ are given by

$$w_{k|T}^i \propto w_k^i p(y_{k+1:T}, \eta'_{k+1:T}|\eta_{0:k}^i, y_{0:k}), \tag{12}$$

where (12) is found by propagating zero, first, and second order moments $\{Z_k, \lambda_k, \Omega_k\}$, dependent on η_k but independent of z_k, backward in time as the nonlinear trajectory is drawn. Given $\{Z_k, \lambda_k, \Omega_k\}$, the predictive density in (12) is

$$p(y_{k+1:T}, \eta'_{k+1:T}|\eta^i_{0:k}, y_{0:k}) \propto Z_k \det(\Lambda_k)^{-1/2} e^{\left(-\frac{1}{2}\zeta_k\right)}, \tag{13}$$

In (13), $\det(\Lambda_k)$ is the determinant of Λ_k and

$$\begin{aligned}\zeta_k &= \|\hat{z}_{k|k}\|^2_{\Omega_k} - 2\lambda_k^{\mathrm{T}}\hat{z}_{k|k} - \|\Gamma_k^{\mathrm{T}}(\lambda_k - \Omega_k\hat{z}_{k|k})\|^2_{\Lambda^{-1}},\\ \Lambda_k &= \Gamma_k^{\mathrm{T}}\Omega_k\Gamma_k + I,\\ \Gamma_k\Gamma_k^{\mathrm{T}} &= P_{k|k}, \text{ with } \|\mu\|^2_\Omega = \mu^{\mathrm{T}}\Omega\mu.\end{aligned}$$

When the full backward trajectory $\eta'_{0:T}$ has been found, the algorithm is typically repeated M times to give a set of backward trajectories $\{\eta^j_{0:T}\}_{j=1}^M$ analogous to (10). Note that (13) is computed for all N particles, yielding the complexity $\mathcal{O}(TMN)$. To find smoothed estimates of the linear states for each nonlinear trajectory, different constrained linear smoothers can be employed. This finally gives the approximated smoothing density as in (11). See [39] for more details.

V. Rao-Blackwellized Particle Filters with Out-of-Sequence Measurement Processing

As described in Sec. III, the aim is to estimate the density

$$p(z_k, \eta_k|y_{0:k}, y_\tau). \tag{14}$$

To utilize the linear structure in (1), factorize (14) as

$$p(z_k, \eta_k|y_{0:k}, y_\tau) = p(z_k|\eta_k, y_{0:k}, y_\tau)p(\eta_k|y_{0:k}, y_\tau). \tag{15}$$

Using Bayes' rule on the second factor of (15) gives

$$p(\eta_k|y_{0:k}, y_\tau) \propto p(y_\tau|\eta_k, y_{0:k})p(\eta_k|y_{0:k}). \tag{16}$$

With the RBPF approximating $p(\eta_k|y_{0:k})$, (16) transforms to

$$p(\eta_k|y_{0:k}, y_\tau) \approx \sum_{i=1}^N w^i_{k|k,\tau}\delta_{\eta^i_k}(\eta_k), \tag{17a}$$

$$w^i_{k|k,\tau} \propto w^i_k p(y_\tau|\eta^i_k, y_{0:k}). \tag{17b}$$

Moreover, it holds that

$$p(y_\tau|\eta^i_k, y_{0:k}) = \int p(y_\tau|z^i_k, \eta^i_k, y_{0:k})p(z^i_k|\eta^i_k, y_{0:k})\,\mathrm{d}z^i_k. \tag{18}$$

Thus, to update the posterior $p(\eta_k|y_{0:k})$ with y_τ to form (17a), the likelihoods $\{p(y_\tau|z^i_k, \eta^i_k, y_{0:k})\}_{i=1}^N$ are needed. To incorporate the OOSM y_τ in the first factor on the right-hand side in (15), recast it as

$$p(z_k|\eta_k, y_{0:k}, y_\tau) = \frac{p(y_\tau|z_k, \eta_k, y_{0:k})p(z_k|\eta_k, y_{0:k})}{\int p(y_\tau|z_k, \eta_k, y_{0:k})p(z_k|\eta_k, y_{0:k})\,\mathrm{d}z_k}. \tag{19}$$

The RBPF (6) can be used to approximate the second factor in the numerator of (19). The first term in the numerator equals the first term on the right-hand side of (18). Hence, what remains is to evaluate the densities $\{p(y_\tau|z^i_k, \eta^i_k, y_{0:k})\}_{i=1}^N$. In the following, we present two different approaches for computing $p(y_\tau|x^i_k, y_{0:k}) = p(y_\tau|z^i_k, \eta^i_k, y_{0:k})$ and thus performing the PF and Kalman filter OOSM updates.

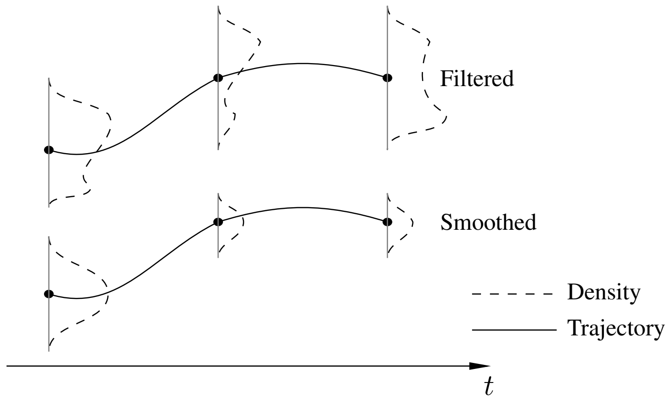

Fig. 2. Filtered and smoothed trajectories (solid) and the corresponding estimated densities (dashed). Since smoothing utilizes more information it typically suppresses the impact of a possibly dominant multimodal behavior in the posterior. Thus, the smoothing density in general needs fewer particles.

A. OOSM Processing with Supporting RBPF

For the first algorithm (SERBPF), the focus is on finding a computationally efficient method that approximately computes $p(y_\tau|x^i_k, y_{0:k})$, by explicitly utilizing the linear substructure (1a). Start with computing $p(y_\tau|x^i_k, y_{0:k})$ as

$$p(y_\tau|x^i_k, y_{0:k}) = \int p(y_\tau|z_\tau, \eta_{0:\tau})p(z_\tau, \eta_{0:\tau}|x^i_k, y_{0:k})\,\mathrm{d}z_\tau \mathrm{d}\eta_{0:\tau}. \tag{20}$$

The density $p(z_\tau, \eta_{0:\tau}|x^i_k, y_{0:k})$ in (20) is a smoothing density, which is rewritten as

$$\begin{aligned}p(z_\tau, \eta_{0:\tau}|x^i_k, y_{0:k}) \propto\ & p(x^i_k, y_{k-l+1:k-1}|z_\tau, \eta_{0:\tau})\\ & \times p(z_\tau, \eta_{0:\tau}|y_{0:k-l}),\end{aligned} \tag{21}$$

where y_k is removed since x^i_k is given. Furthermore, $p(z_\tau, \eta_{0:\tau}|y_{0:k-l})$ in (21) is approximated by the RBPF (5) using a prediction to time t_τ. The first density on the right-hand side in (21) is a measurement update using both $y_{k-l+1:k-1}$ and x^i_k as measurements. We therefore need to propagate the past, $\{z_\tau, \eta_{0:\tau}\}$, to update with $\{x^i_k, y_{k-l+1:k-1}\}$.

As discussed in Sec. IV-B, the PF (or RBPF) also produces an approximate solution to the smoothing problem. Consequently, we apply a supporting (additional) RBPF to find (21). With this approach, we efficiently find both the smoothing weights and the linear smoothing estimates in one forward sweep as follows: Initialize an additional RBPF from time index $k-l$. Because (21) is a smoothing density, it can be represented by fewer particles than for the original RBPF. Fig. 2 illustrates the idea. Therefore, start with sampling $M_{\mathrm{FF}} \ll N$ estimates $\{\hat{z}^j_{k-l|k-l}, \eta^j_{k-l}\}_{j=1}^{M_{\mathrm{FF}}}$ from the filtering density at time index $k-l$. Note that performing this sampling already in the original forward RBPF at each time step reduces the storage requirements. Predict the estimates up to time index τ, and augment the linear state vector and initialize:

$$\begin{aligned}\zeta^j_m &= \left((z^j_m)^{\mathrm{T}} \quad (z^j_\tau)^{\mathrm{T}}\right)^{\mathrm{T}}\\ \zeta^j_\tau &= \begin{pmatrix}\hat{z}^j_{\tau|k-l}\\ \hat{z}^j_{\tau|k-l}\end{pmatrix}, \quad \bar{P}^j_\tau = \begin{pmatrix}P^j_{\tau|k-l} & P^j_{\tau|k-l}\\ P^j_{\tau|k-l} & P^j_{\tau|k-l}\end{pmatrix}.\end{aligned} \tag{22}$$

Then, for each $m = k-l+1, \dots, k-1$, execute the supporting RBPF with resampling as usual, where the lower part of ζ_m^j is the linear smoothing estimate, the lower right block in $\bar{P}_m^j$ gives the smoothing covariance, and the upper right block in $\bar{P}_m$ gives the crosscovariance $P_{m,\tau|m}^j$ between the states at time index m and τ. At time t_k, use $\{\hat{z}_{k|k}^i, \eta_k^i\}_{i=1}^N$ as measurements. Thus, at the end of the recursion an approximation to the smoothing density (21) is given by the Gaussian mixture

$$p(z_\tau, \eta_{0:\tau}|x_k^i, y_{0:k}) \approx \sum_{j=1}^{M_{\text{FF}}} q_{\tau|k,i}^j \mathcal{N}(z_\tau|\hat{z}_{\tau|k,i}^j, P_{\tau|k,i}^j)\delta_{\eta_{0:\tau}^j}(\eta_{0:\tau}), \tag{23}$$

where $q_{\tau|k,i}^j$ are the smoothing weights given measurements up to time t_{k-1} and state estimate i at time t_k. Given (23), an approximation to (20) is

$$p(y_\tau|x_k^i, y_{0:k}) \approx \sum_{j=1}^{M_{\text{FF}}} q_{\tau|k,i}^j p(y_\tau|\hat{z}_{\tau|k,i}^j, \eta_{0:\tau}^j), \tag{24}$$

where the measurement likelihood is given by

$$p(y_\tau|z_{\tau|k,i}^j, \eta_{0:\tau}^j) = \mathcal{N}(y_\tau|\hat{y}_{\tau|k,i}^j, \Sigma_{\tau|k,i}^j), \tag{25}$$

and the mean and covariance are

$$\begin{aligned}\hat{y}_{\tau|k,i}^j &= h_\tau^j + C_\tau^j \hat{z}_{\tau|k,i}^j\\ \Sigma_{\tau|k,i}^j &= C_\tau^j P_{\tau|k,i}^j (C_\tau^j)^{\mathrm{T}} + R_\tau^j.\end{aligned}$$

With (24) inserted into (18), the particle weights (17b) become

$$w_{k|k,\tau}^i \propto w_k^i \sum_{j=1}^{M_{\text{FF}}} q_{\tau|k,i}^j p(y_\tau|\hat{z}_{\tau|k,i}^j, \eta_{0:\tau}^j). \tag{26}$$

To find (19) (i.e., the density for the linear states), we utilize Proposition 1, which is given next.

Proposition 1. *Given* (24)*, for each* $i \in \{1, \dots, N\}$*,* (19) *is given by*

$$p(z_k^i|\eta_k^i, y_{0:k}, y_\tau) = \mathcal{N}(z_k|\hat{z}_{k|k,\tau}^i, P_{k|k,\tau}^i),$$

where

$$\begin{aligned}\hat{z}_{k|k,\tau}^i &= \hat{z}_{k|k}^i + E^i\\ P_{k|k,\tau}^i &= P_{k|k}^i + \sum_{j=1}^{M_{\text{FF}}} q_{\tau|k,i}^j \Big((E^{j,i} - E^i)(E^{j,i} - E^i)^{\mathrm{T}}\\ &\quad - W_{k,\tau}^{j,i} \Sigma_{\tau|k,i}^j (W_{k,\tau}^{j,i})^{\mathrm{T}}\Big)\\ E^i &= \sum_{j=1}^{M_{\text{FF}}} q_{\tau|k,i}^j W_{k,\tau}^{j,i} e_\tau^{j,i}\\ E^{j,i} &= W_{k,\tau}^{j,i} e_\tau^{j,i}\\ e_\tau^{j,i} &= y_\tau - \hat{y}_{\tau|k,i}^j\\ W_{k,\tau}^{j,i} &= P_{k,\tau|k,i}^j (C_\tau^j)^{\mathrm{T}} (\Sigma_{\tau|k,i}^j)^{-1}.\end{aligned} \tag{27}$$

The algorithm is denoted by SERBPF and is summarized in Algorithm 1. The storage requirements of the algorithm are $\{\{\hat{z}_{m|m}^j, P_{m|m}^j, \eta_m^j\}_{m=k-l_{\max}}^{k-1}, \{y_m\}_{m=k-l_{\max+1}}^{k-1}\}_{j=1}^{M_{\text{FF}}}$. Because the RBPF has computational complexity $\mathcal{O}(N)$ [40], the algorithm has complexity $\mathcal{O}(lM_{\text{FF}} + M_{\text{FF}}N)$.

Algorithm 1. SERBPF

```
Input: $\{x_{k-l}^j, P_{k-l|k-l}^j, x_k^i, P_{k|k}^i, w_k^i, y_{k-l+1:k-1}\}_{j=1}^{M_{\text{FF}}}$
 1: Predict up to time $t_\tau$, yielding $p(z_\tau^j, \eta_{0:\tau}^j|y_{0:k-l})$.
 2: Initialize smoothing weights as $q_{\tau|k-l}^j = 1/M_{FF}$.
 3: Augment the state vector and initialize using (22).
 4: for $m = k-l+1$ to $k-1$ do
 5:     Perform RBPF prediction to time $t_m$.
 6:     Perform weight update according to forward-filter measurement likelihood.
 7:     if $(\sum(q_m^j)^2)^{-1} < N_{\text{eff}}$ then
 8:         Resample $M_{\text{FF}}$ new particles with replacement. Keep track of $\{\eta_{0:\tau}^j\}_{j=1}^{M_{\text{FF}}}$ and equalize weights, $q_{\tau|m}^j = 1/M_{\text{FF}}$.
 9:     end if
10:     Perform augmented Kalman-filter measurement update.
11: end for
12: Perform RBPF prediction to time $t_k$.
13: Use $\{\hat{z}_{k|k}^i, \eta_k^i\}_{i=1}^N$ as measurements with (1a) and (1b) as measurement likelihoods.
14: Update weights as in (26).
15: Update linear estimates by applying (27).
Output: $\{\hat{z}_{k|k,\tau}^i, P_{k|k,\tau}^i, w_{k|k,\tau}^i\}_{i=1}^N$
```

B. OOSM Processing with Backward Simulation

Algorithm 1 is computationally efficient, but the generated smoothing density may, especially for large delays, suffer from degeneracy. Accordingly, the estimate of the measurement likelihood used to associate the current estimate with the OOSM may be inaccurate. To avoid this and thus to improve estimation accuracy, we instead rewrite $p(y_\tau|x_k^i, y_{0:k})$ as

$$\begin{aligned}p(y_\tau|x_k^i, y_{0:k}) = &\int p(y_\tau|z_\tau, \eta_{0:\tau})\\ &\cdot p(z_\tau, \eta_\tau, \eta_{0:k-1}|x_k^i, y_{0:k})\mathrm{d}z_\tau \mathrm{d}\eta_\tau \mathrm{d}\eta_{0:k-1}\end{aligned} \tag{28}$$

for updating the weights using (18), where

$$\begin{aligned}p(z_\tau, \eta_\tau, \eta_{0:k-1}|x_k^i, y_{0:k}) &= p(z_\tau|\eta_{0:\tau}, \eta_{k-l:k-1}, x_k^i, y_{0:k})\\ &\cdot p(\eta_{0:\tau}|\eta_{k-l:k-1}, x_k^i, y_{0:k}) p(\eta_{k-l:k-1}|x_k^i, y_{0:k}).\end{aligned}$$

For later use in the update of the linear states (19), we write

$$\begin{aligned}p(y_\tau|x_k^i, y_{0:k}) = &\int p(y_\tau|z_\tau, \eta_\tau)\\ &\cdot p(z_\tau, \eta_\tau, \eta_{k-l:k-1}|x_k^i, y_{0:k})\mathrm{d}z_\tau \mathrm{d}\eta_\tau \mathrm{d}\eta_{k-l:k-1}\end{aligned} \tag{29}$$

where

$$\begin{aligned}p(z_\tau, \eta_\tau, \eta_{k-l:k-1}|x_k^i, y_{0:k}) &= p(z_\tau|\eta_\tau, \eta_{k-l:k-1}, x_k^i, y_{0:k})\\ &\cdot p(\eta_\tau|\eta_{k-l:k-1}, x_k^i, y_{0:k}) p(\eta_{k-l:k-1}|x_k^i, y_{0:k}).\end{aligned}$$

In both (28) and (29), $p(\eta_{k-l:k-1}|x_k^i, y_{0:k})$ factorizes in the same manner as the sequential factorization (8):

$$\begin{aligned}p(\eta_{k-l:k-1}|x_k^i, y_{0:k}) &= p(\eta_{k-1}|x_k^i, y_{0:k})\\ &\cdot \prod_{m=k-l}^{k-2} p(\eta_m|\eta_{m+1:k-1}, x_k^i, y_{0:k}).\end{aligned} \tag{30}$$

The smoothing density (30) can be solved for similar to the RBPS described in Sec. IV-B. However, the smoother derived in [39] assumes that the covariance matrix in (2) is diagonal (i.e., v_k^z and v_k^η are mutually independent). We take this restriction into account by decorrelating the noise in the same manner as in [9], which yields a modifed system. The modified system can then be used in the derivations in [39], which amounts to replacing f_k with $\bar{f}_k = f_k + Q_k^{z\eta}(Q_k^\eta)^{-1}(\eta_{k+1} - g_k)$, A_k with $\bar{A}_k = A_k - Q_k^{z\eta}(Q_k^\eta)^{-1}B_k$, and the covariance matrix for the process noise acting on the linear states, Q_k^z, with $\bar{Q}_k^z = Q_k^z - Q_k^{z\eta}(Q_k^\eta)^{-1}(Q^{z\eta})^{\mathrm{T}}$ in the derivations. Also, we make adaptations for the smoother to be applicable to the OOSM scenario and take a computationally efficient rejection-sampling approach for finding the backward trajectories, instead of computing the smoothing weights for each particle in each time step. How to find (30) is described next.

1) Finding the Nonlinear Backward Trajectories: First, at time t_k, instead of drawing $\eta_k' = \eta_k^j$ with probability w_k^j for $j \in \{1, \ldots, M_{\mathrm{BS}}\}$, giving the starting point for M_{BS} backward trajectories, we must choose $\eta_k' = \eta_k^i$ for each of the N forward particles to associate the filtered estimates with the OOSM. Thus, in total M_{BS} backward trajectories are associated with each forward particle. Second, if (12) was to be computed for all particles in each time step for appending the trajectories, the complexity would be $\mathcal{O}(N^2 M_{\mathrm{BS}})$ in each time step. However, by again using the assumption that smoothing densities are less complex than filtering densities, see Fig. 2, we may sample $D \leq N$ particles instead in each time step to form D smoothing weights. The validity of this assumption is analyzed and discussed in Secs. VI and VII.

There exist computationally efficient particle smoothers that create backward trajectories without explicitly computing the smoothing weights, with sustained algorithm behavior [41]. As will be clear later, we only need the smoothing weights at time step $k - l$. Thus, the approach in [41] can be adapted to our mixed linear/nonlinear model setting for $m = k-1, \ldots, k-l+1$: The aim is to sample from the distribution formed by $\{w_{m|k}^{j,d}\}_{d=1}^D$, without actually computing the smoothing weights. Here, we use the forward weights $\{w_m^d\}_{d=1}^D$ as proposal distribution, which is already known. By noting that the transition density in (13) is bounded from above as $p(y_{m+1:k}, \eta_{m+1:k-1}^j, \eta_k^i | \eta_{0:m}^d, y_{0:m}) \leq \sigma_m^j$ where

$$\sigma_m^j = \max_{d=1,\ldots,D} Z_m^{j,d} \det(\Lambda_m^{j,d})^{-1/2},$$

we can perform rejection sampling to extend each backward trajectory as $\{\eta_m^J, \eta_{m+1:k-1}^j, \eta_k^i\}$. Algorithm 2 outlines the procedure.

Algorithm 2. Rejection Sampling

```
1:  $L = \{1, \ldots, M_{\mathrm{BS}}\}$.
2:  while $L$ is not empty do
3:      Set $n = \mathrm{size}(L)$.
4:      Sample $\{I(j)\}_{j=1}^n$ independently with probabilities proportional to $w_m^{I(j)}$.
5:      Sample $\{U(j)\}_{j=1}^n$ independently and uniformly over $[0, 1]$.
6:      for $j = 1$ to $n$ do
7:          if $U(j) \leq p(y_{m+1:k}, \eta_{m+1:k-1}^{L(j)}, \eta_k^i | \eta_{0:m}^{I(j)}, y_{0:m}) / \sigma_m^{L(j)}$ then
8:              Set $J(L(j)) = I(j)$.
9:              Set $L = L \setminus L(j)$.
10:         end if
11:     end for
12: end while
13: return indices $\{J(j)\}_{j=1}^{M_{\mathrm{BS}}}$.
```

There is no upper bound on the number of executions of the while-loop in Algorithm 2. Hence, a threshold $C_{\max}$ for the maximum number of iterations in the while-loop is set. If L is still empty after $C_{\max}$ iterations, an index is sampled from the smoothing weights (12) by computing the transition densities for the indices that have not already been selected. Although Algorithm 2 occasionally does not succeed in finding an index, on average it provides a noticeable speedup compared with [15]. Note that for $M_{\mathrm{BS}} = D$ and D large, Algorithm 2 has close to linear computational complexity in N.

At (the last) time step $k - l$, for each trajectory, draw D forward particles $\{\eta_{0:k-l}^d\}_{d=1}^D$ with probability w_{k-l}^d and compute the right-hand side of (13) for these D particles. This implies that (30) at time t_{k-l} approximates to

$$p(\eta_{k-l:k-1} | x_k^i, y_{0:k}) \approx \frac{1}{M_{\mathrm{BS}}} \sum_{j=1}^{M_{\mathrm{BS}}} \sum_{d=1}^{D} w_{k-l|k,i}^{j,d} \delta_{\eta_{0:k-l}^d}(\eta_{0:k-l}). \quad (31)$$

By drawing a particle η_{k-l} with probability $w_{k-l|k,i}^{j,d}$ for each j, an approximation to (30) based on the mean of the full backward trajectories is

$$p(\eta_{k-l:k-1} | x_k^i, y_{0:k}) \approx \frac{1}{M_{\mathrm{BS}}} \sum_{j=1}^{M_{\mathrm{BS}}} \delta_{\eta_{k-l:k-1}^j}(\eta_{k-l:k-1}). \quad (32)$$

2) Updating the Weights and Linear Estimates: To update the particle weights with the OOSM, insert (31) into (28):

$$p(y_\tau | x_k^i, y_{0:k}) \approx \frac{1}{M_{\mathrm{BS}}} \sum_{j=1}^{M_{\mathrm{BS}}} \sum_{d=1}^{D} w_{\tau|k,i}^{j,d} p(y_\tau | \hat{z}_{\tau|k-l}^d, \eta_{0:\tau}^d). \quad (33)$$

Here, $w_{\tau|k,i}^{j,d} = w_{k-l|k,i}^{j,d}$ since only a time update differs between t_τ and t_{k-l}. Plugging (33) into (18) and using (17b) gives the weights after processing the l-step lag OOSM as

$$w_{k|k,\tau}^i \propto w_k^i \sum_{j=1}^{M_{\mathrm{BS}}} \sum_{d=1}^{D} w_{\tau|k,i}^{j,d} p(y_\tau | \hat{z}_{\tau|k-l}^d, \eta_\tau^d), \quad (34)$$

where $p(y_\tau | \hat{z}_{\tau|k-l}^d, \eta_\tau^d)$ is computed similarly to (25).

To update the linear density $p(z_k | \eta_k, y_{0:k}, y_\tau)$ (i.e., (19)), the measurement update step (29) needs the smoothing density $p(z_\tau | \eta_\tau, \eta_{k-l:k-1}^j, x_k^i, y_{0:k})$, for which we can resort to different linear smoothers (conditioned on the generated backward trajectories (32) and the measurements). Here we choose an RTS-smoother [42], which for the mixed linear/nonlinear model class is given by Proposition 2:

Proposition 2. *Given the model* (1)*, the filtering density* $p(z_k | \eta_{0:k}, y_{0:k})$*, and the trajectory* $\eta_{m:k}$*, the marginal smoothing density for* z_m *is given by*

$$p(z_m | \eta_{m:k}, y_{0:k}) = \mathcal{N}(z_m | \hat{z}_{m|k}, P_{m|k}),$$

where

$$\begin{aligned}
\hat{z}_{m|k} &= \hat{z}^*_{m|m} + H_m(\hat{z}_{m+1|k} - \hat{z}_{m+1|m}) \\
P_{m|k} &= P^*_{m|m} + H_m(P_{m+1|k} - P_{m+1|m}) \\
P_{m,k|m} &= P_{m+1,k|m+1}H_m^{\mathrm{T}} \\
H_m &= P^*_{m|m}A_m^{\mathrm{T}}P^{-1}_{m+1|m} \\
\hat{z}_{m+1|m} &= \bar{f}_m + \bar{A}_m\hat{z}^*_{m|m} \\
P_{m+1|m} &= \bar{A}_mP^*_{m|m}\bar{A}_m^{\mathrm{T}} + \bar{Q}^z_m \\
\bar{f}_m &= f_m + Q^{z\eta}_m(Q^\eta_m)^{-1}(\eta_{m+1} - g_m) \\
\bar{A}_m &= A_m - Q^{z\eta}_m(Q^\eta_m)^{-1}B_m \\
\bar{Q}^z_m &= Q^z_m - Q^{z\eta}_m(Q^\eta_m)^{-1}(Q^{z\eta})^{\mathrm{T}} \\
\hat{z}^*_{m|m} &= \hat{z}_{m|m} + L_m(\eta_{m+1} - g_m - A_m\hat{z}_{m|m}) \\
P^*_{m|m} &= P_{m|m} - L_mN_mL_m^{\mathrm{T}} \\
L_m &= P_{m|m}B_mN_m^{-1} \\
N_m &= B_mP_{m|m}B_m^{\mathrm{T}} + Q^\eta_m.
\end{aligned} \tag{35}$$

By using Proposition 2 to iterate back from t_k to t_{k-l} and then performing a time update to t_τ,

$$p(z_\tau|\eta_\tau, \eta^j_{k-l:k-1}, x^i_k, y_{0:k}) = \mathcal{N}(z_\tau|\hat{z}^j_{\tau|k,i}, P^j_{\tau|k,i}). \tag{36}$$

With (32) and (36) inserted in (29),

$$p(y_\tau|x^i_k, y_{0:k}) \approx \frac{1}{M_{\mathrm{BS}}}\sum_{j=1}^{M_{\mathrm{BS}}} p(y_\tau|\eta^j_\tau, \hat{z}^j_{\tau|k,i}) \tag{37}$$

where, again, the measurement likelihood is computed similarly to (25). Finally, we update the linear estimates and the associated covariances, and thereby find (19) for each $i = 1, \dots, N$, using the measurement update (37) as

$$\begin{aligned}
\hat{z}^i_{k|k,\tau} &= \hat{z}^i_{k|k} + \frac{1}{M_{\mathrm{BS}}}\sum_{j=1}^{M_{\mathrm{BS}}} W^{j,i}_{k,\tau}e^{j,i}_\tau \\
P^i_{k|k,\tau} &= P^i_{k|k} - \frac{1}{M_{\mathrm{BS}}}\sum_{j=1}^{M_{\mathrm{BS}}} W^{j,i}_{k,\tau}\Sigma^j_{\tau|k,i}(W^{j,i}_{k,\tau})^{\mathrm{T}} \\
W^{j,i}_{k,\tau} &= P^j_{k,\tau|k,i}(C^{j,i}_\tau)^{\mathrm{T}}(\Sigma^j_{\tau|k,i})^{-1} \\
\Sigma^j_{\tau|k,i} &= C^{j,i}_\tau P^j_{\tau|k,i}(C^{j,i}_\tau)^{\mathrm{T}} + R_\tau \\
e^{j,i}_\tau &= y_\tau - h^{j,i}_\tau - C^{j,i}_\tau\hat{z}^j_{\tau|k,i}.
\end{aligned} \tag{38}$$

The update (38) is given in [23] for the purely linear-Gaussian setting, and the extension is straightforward. The algorithm is denoted by RBOOSMBS and is summarized in Algorithm 3. The storage requirements of the algorithm are $\{\hat{z}^j_{m|m}, P^j_{m|m}, \eta^j_m, w^j_m, y_m\}_{j=1}^D$ for $m = k - l_{\max}, \dots, k-1$. Without using rejection sampling, Algorithm 3 has computational complexity $\mathcal{O}(lNM_{\mathrm{BS}}D)$. This complexity is most often reduced when utilizing Algorithm 2, and should be compared with $\mathcal{O}(lM_{\mathrm{FF}} + M_{\mathrm{FF}}N)$ for SERBPF and $\mathcal{O}((l-1)N^3 + N^2)$ for A-PF in [13].

Algorithm 3. RBOOSMBS

```
Input: $\bigcup_{m=k-l}^{k-1}\{x^d_{m|m}, P^d_{m|m}, w^d_m, x^i_k, P^i_{k|k}, w^i_k, y_m\}_{d=1}^D$
 1: for $i = 1$ to $N$ do
 2:   Set $\eta^j_k = \eta^i_k$ for $j = 1, \dots, M_{\mathrm{BS}}$.
 3:   for $m = k-1$ to $k-l$ do
 4:     if $m > k - l$ then
 5:       Execute Algorithm 2.
 6:     else
 7:       for $j = 1$ to $M_{\mathrm{BS}}$ do
 8:         for $d = 1$ to $D$ do
 9:           Draw $\eta^d_m$ with probability $w^d_m$.
10:           Compute $w^{j,d}_{m|k,i}$ using (12) and (13).
11:         end for
12:         Normalize: $w^{j,d}_{m|k,i} = w^{j,d}_{m|k,i}\left(\sum_{d=1}^{D} w^{j,d}_{m|k,i}\right)^{-1}$.
13:         Set $J(j) = d$ with probability $w^{j,d}_{m|k,i}$.
14:       end for
15:     end if
16:     Set $\eta^j_{m:k} = \{\eta^{J(j)}_m, \eta^j_{m+1:k}\}$ for $j = 1, \dots, M_{\mathrm{BS}}$.
17:     Perform a backward RTS step using (35).
18:   end for
19:   Update weight $w^i_k$ using (34).
20:   Update mean and covariance using (38).
21: end for
Output: $\{\hat{z}^i_{k|k,\tau}, P^i_{k|k,\tau}, w^i_{k|k,\tau}\}_{i=1}^N$
```

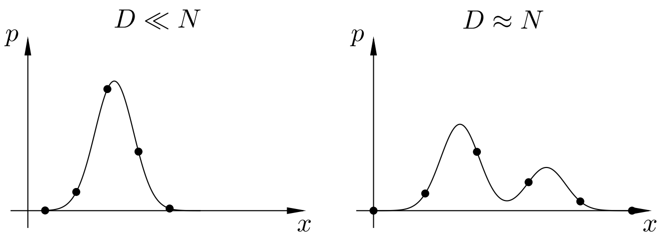

Fig. 3. The density to the left illustrates an example where it may suffice with only using a subset of the available particles for approximating the density well. To the right is an example where a larger amount of particles is needed.

Remark 1. By using $D \leq N$ particles in Algorithm 3, it is possible to trade tracking performance against computation requirements; for example, for sharp densities it may suffice with using $D \ll N$ particles to get adequate performance, thus saving processing time. See Fig. 3 for an illustration. D can also be used as a tradeoff with the number of smoothing iterations M_{BS}, because sometimes it may be advantageous to perform the smoothing iterations many times instead of utilizing all N particles in each smoothing step.

Remark 2. If the measurements are stored it is theoretically possible to reorder and reprocess the measurements when an OOSM arrives. However, reprocessing includes redoing the data association, which in itself can be challenging [13]. Furthermore, SERBPF has lower computational complexity than reordering and reprocessing for $M_{\mathrm{FF}} \ll N$.

VI. Numerical Results

The proposed algorithms are evaluated on three examples by comparing their performance against four other particle filters. For a fair comparison of the algorithms' abilities to process the OOSMs, all filters use identical bootstrap RBPFs in the forward direction [9]. The simulations are conducted in MATLAB. The root-mean square error (RMSE) of the weighted mean at each time step and the time average of it are used as performance measures. The time-averaged RMSE is found by taking the mean of the RMSE. The compared methods are

RBPFDISC: An RBPF that discards all OOSMs and thus only processes measurements with zero delay.
RBPF: An offline, idealized RBPF that collects all measurements from both sensors with zero delay. This filter serves as a performance benchmark.
A-PF: The exact Bayesian inference solution in [13].
SEPFFPS: The SEPF in [30], which uses an augmented-state extended Kalman smoother for processing the OOSMs.
SERBPF: The first method proposed in this paper, described in Sec. V-A and summarized in Algorithm 1.
RBOOSMBS: The backward-simulation based method described in Sec. V-B and summarized in Algorithm 3.

A. Example 1

This example considers the fourth-order system

$$\begin{aligned} z_{k+1} &= \begin{pmatrix} 1 & 0.3 & 0 \\ 0 & 0.92 & -0.3 \\ 0 & 0.3 & 0.92 \end{pmatrix} z_k + v_k^z, \\ \eta_{k+1} &= \arctan(\eta_k) + \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} z_k + v_k^\eta, \\ y_k &= \begin{pmatrix} 0.1\eta_k^2 \mathrm{sign}(\eta_k) \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 1 & -1 & 1 \end{pmatrix} z_k + e_k, \end{aligned} \tag{39}$$

where $\mathrm{sign}(\cdot)$ is the signum function. The noise is mutually independent, white, and Gaussian distributed according to $v_k \sim \mathcal{N}(0, 0.01 I_{4\times4})$ and $e_k \sim \mathcal{N}(0, 0.1 I_{2\times2})$, where $I_{n\times n}$ is the identity matrix of dimension $n \times n$. Model (39) has previously been used in several papers, for example [43]. In this setup, the second sensor (i.e., the second element in y_k) has communication issues:[1] A measurement arrives with probability 0.5 and is delayed according to a discrete-valued uniform distribution in the interval $[1, 4]$ samples, with the sampling period $T_s = 1$ s. This should be interpreted as that 50% of the packets of the second sensor are lost on their way to the communication center, and those that arrive are delayed between one and four seconds. Since only 50% of the measurements arrive on average, the performance of RBPF is impossible to reproduce with any of the other methods. The initial estimate x_0 and covariance matrix P_0 for all filters are

$$x_0 = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}^{\mathrm{T}}, \quad P_0 = \mathrm{diag}(0, 0, 0, 1),$$

where $\mathrm{diag}(\cdot)$ is the diagonal matrix.

The following results are for 20000 Monte-Carlo simulations with $T = 50$ time steps in each simulation. Fig. 4 shows the RMSE for all four states using $N = 100$ particles in the forward filters. SERBPF uses $M_{\mathrm{FF}} = N$ particles in the supporting RBPF. The choice $M_{\mathrm{FF}} = N$ is only made to give an indication of the performance possible to achieve; in practice one should choose $M_{\mathrm{FF}} \ll N$ to save computation time. RBOOSMBS uses $M_{\mathrm{BS}} = 1, D = N$. Clearly, RBOOSMBS performs better than the other methods in terms of RMSE, followed by SERBPF. We expect the performance difference between RBOOSMBS and SERBPF to increase when the delay increases. At first glance, the performance differences might seem minor. To give an indication of how close the proposed algorithms are to the lower performance bounds and to show that the improvements are indeed significant, Fig. 5 presents a close-up of the RMSE for z^2 when also comparing with a particle filter that reorders and reprocesses the measurements when an OOSM arrives (i.e., an optimal approach given the information). This filter is denoted by RERUN in the figure. As seen, RBOOSMBS is very close (approximately 1.5%) to the performance of RERUN when compared with the other filters. Moreover, RBOOSMBS outperforms SERBPF with approximately 6% and A-PF with roughly 10%. Fig. 5 also indicates that the relative performance of RBOOSMBS is far superior to both SERBPF and A-PF. We obtained similar results for the other linear states. For z^1 and η, also SERBPF performed similarly to RERUN. The relatively low performance of A-PF is related to that 100 particles is not enough to reliably associate the estimates with the OOSMs when the model structure in (39) is unexploited.

[1]We also performed simulations where the first sensor had communication issues. This resulted in that all filters had similar performance. The reason is that the second measurement contains more information; consequently, in the following we only present results where the second sensor delivers OOSMs.

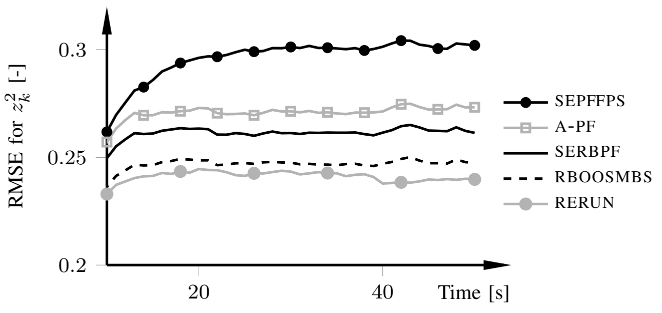

Fig. 5. A close-up of the RMSEs for z_k^2, corresponding to Fig. 4. RBOOSMBS (and to some extent SERBPF) is close to optimal.

Table I displays time-averaged RMSEs for $N = 100$ and $N = 200$ particles. We have added the results when using $M_{\mathrm{FF}} = 0.1N$ in SERBPF, because $M_{\mathrm{FF}} \ll N$ in practical applications. Also added is the results for RBOOSMBS when using $D = 0.25N$. Using $M_{\mathrm{FF}} = 0.1N$ instead of $M_{\mathrm{FF}} = N$ in SERBPF gives almost no decrease in estimation accuracy, but much lower computational complexity. Likewise, using $D = 0.25N$ in RBOOSMBS gives an insignificant change in RMSE but considerably reduced computation time.

The average computation time of 100 Monte-Carlo executions as function of N for A-PF, SERBPF, and RBOOSMBS are shown in Fig. 6 when the delay is fixed to $l = 3$. The implementation was performed in MATLAB, and no measures to code optimization have been taken. The computation time for one set of particles is language and implementation dependent. Nevertheless, Fig. 6 indicates the respective complexity increase when increasing N. For $N = 200$, setting $D = 0.25N$ in RBOOSMBS gives a reduction in computation time with 75% compared with $D = N$, whereas $M_{\mathrm{FF}} = 0.1N$ gives a 87.5% decrease compared with $M_{\mathrm{FF}} = N$.

B. Example 2

This evaluation uses a fifth-order mixed linear/nonlinear system, with the nonlinear part given by

$$\begin{aligned} \eta_{k+1} &= 0.5\eta_k + \theta_k \frac{\eta_k}{1 + \eta_k^2} + 8\cos(1.2k) + v_k^\eta, \\ y_k &= 0.05\eta_k^2 + e_k, \end{aligned} \tag{40}$$

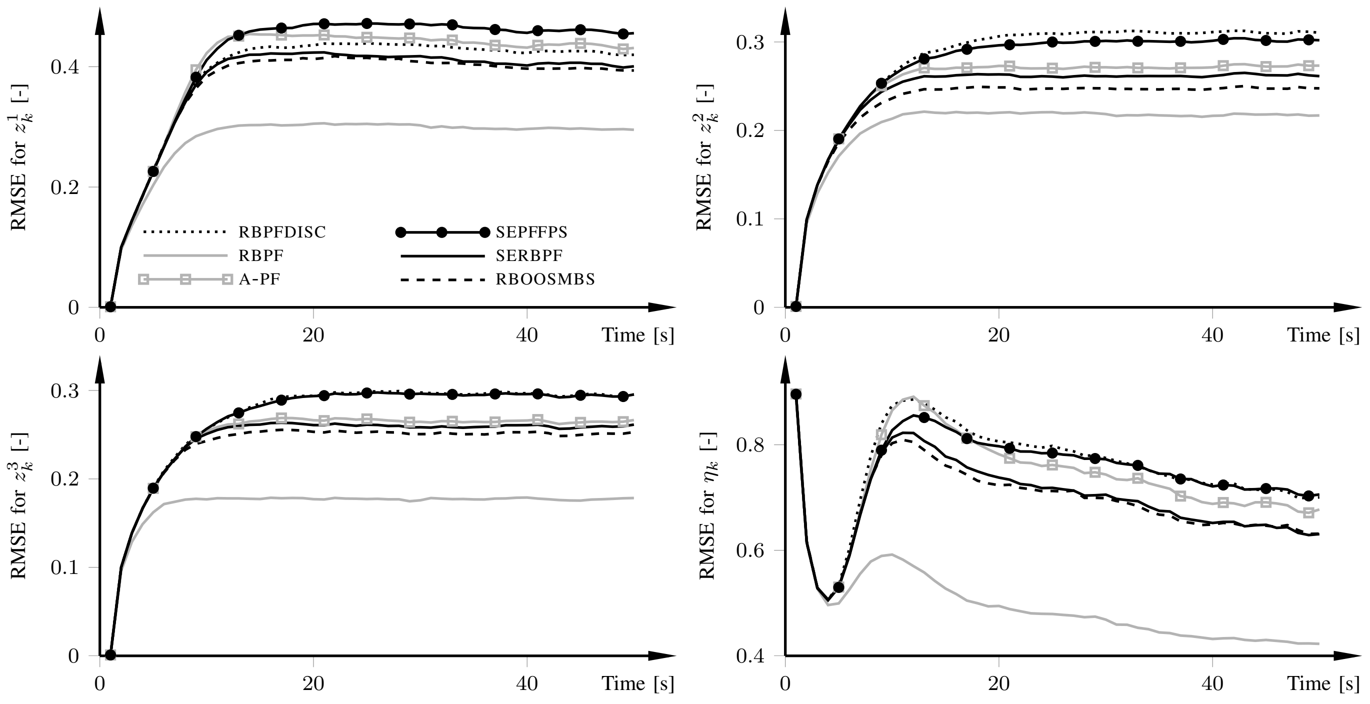

Fig. 4. RMSEs of the four states in Sec. VI-A numbered row-wise from the top left with the nonlinear state in the second row, second column, for 20000 Monte-Carlo simulations with the number of particles set to $N = 100$. Moreover, SERBPF uses $M_{\mathrm{FF}} = N$ particles in the supporting RBPF and RBOOSMBS uses $M_{\mathrm{BS}} = 1$ backward trajectory, something that is enough in most cases for the considered lags.

TABLE I
TIME-AVERAGED RMSE VALUES CORRESPONDING TO FIG. 4. FOR THE ALGORITHMS ACCOUNTING FOR OOSMS, THE SMALLEST ERRORS ARE SHOWN IN BOLDFACE FONT. IN THIS EXAMPLE, IT IS POSSIBLE TO DRASTICALLY REDUCE BOTH M_{FF} AND D WHILE RETAINING ESTIMATION ACCURACY.

	$N=100$				$N=200$			
Algorithm	z_1	z_2	z_3	η	z_1	z_2	z_3	η
RBPFDISC	0.391	0.281	0.270	0.756	0.389	0.280	0.269	0.752
SEPFFPS	0.418	0.273	0.269	0.747	0.414	0.271	0.267	0.740
A-PF	0.403	0.253	0.248	0.734	0.400	0.250	0.245	0.728
SERBPF, $M_{\mathrm{FF}} = 0.1N$	0.378	0.246	0.245	0.707	0.374	0.244	0.242	0.693
SERBPF, $M_{\mathrm{FF}} = N$	0.376	0.244	0.243	0.696	0.373	0.243	0.241	0.689
RBOOSMBS, $D = 0.25N$	0.371	0.232	**0.236**	0.690	0.369	**0.231**	**0.235**	0.686
RBOOSMBS, $D = N$	**0.370**	**0.231**	**0.236**	**0.687**	**0.368**	**0.231**	**0.235**	**0.684**
RBPF	0.279	0.206	0.170	0.494	0.278	0.205	0.170	0.490

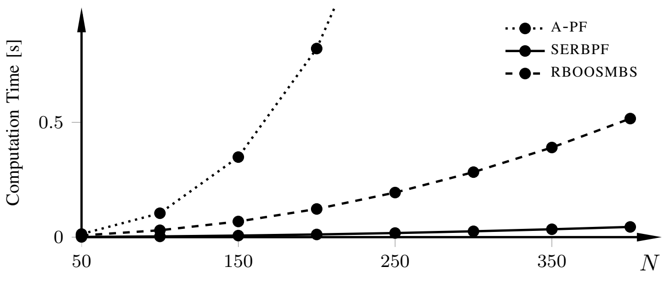

Fig. 6. Average computation times of 100 Monte-Carlo executions for varying number of forward particles for $l = 3$ and $M_{\mathrm{FF}} = N$, $M_{\mathrm{BS}} = 1$, and $D = N$. A decrease to $M_{\mathrm{FF}} = 0.1N$ for SERBPF gives a speedup factor of eight for $N = 200$. Similarly, setting $D = 0.25N$ in RBOOSMBS gives a speedup factor of four. The computation times include the computation time of the RBPF used in the forward filtering.

where $e_k \sim \mathcal{N}(0, 0.1)$ and $v_k^\eta \sim \mathcal{N}(0, 0.005)$. The case with $\theta_k =$ 25 has been used in several papers, among them [13]. Here, θ_k is the output from a linear system with dynamics given by

$$z_{k+1} = \begin{pmatrix} 3 & -1.691 & 0.849 & -0.3201 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \end{pmatrix} z_k + v_k^z, \quad (41)$$

$$\theta_k = 25 + \begin{pmatrix} 0 & 0.04 & 0.044 & 0.008 \end{pmatrix} z_k, \quad (42)$$

where $v_k^z \sim \mathcal{N}(0, 0.01 I_{4\times 4})$. Again, the noise is mutually independent, white, and Gaussian distributed. The system (40)–(42) has previously been used in, for example, [39]. Note that the nonlinear state is squared in the measurement equation, leading to a bimodal posterior. The initial estimate and covariance matrix for all filters are set to zero for all states. The results are based on two data sets, both generated by executing 20000 Monte-Carlo simulations twice, with 100 time steps in each simulation. The sampling period T_s is set to $T_s = 1$ s. In both data sets $N = 400$. In the first

TABLE II
TIME-AVERAGED RMSES AT THE OOSM ARRIVAL TIMES USING 20000 MONTE-CARLO EXECUTIONS FOR EXAMPLE 2. NOTE THAT THE ERRORS BETWEEN $l = 1$ AND $l = 2$ ARE NOT COMPARABLE, BECAUSE THEY ARE MEASURED AT DIFFERENT TIME STEPS.

	$l = 1$		$l = 2$	
Algorithm	η	θ	η	θ
RBPFDISC	0.453	0.954	0.514	0.895
A-PF	0.448	0.940	0.508	0.892
SERBPF, $M_{\mathrm{FF}} = 0.4N$	0.430	0.856	0.493	0.850
SERBPF, $M_{\mathrm{FF}} = N$	**0.428**	**0.853**	0.484	**0.847**
RBOOSMBS, $D = 0.5N$	0.445	0.950	0.496	0.892
RBOOSMBS, $D = N$	0.437	0.881	**0.476**	0.857
RBPF	0.414	0.844	0.443	0.839

data set every second measurement is delayed one time step (i.e., $l = 1$), whereas in the second data set every third measurement is delayed two time steps (i.e., $l = 2$). Note that in this example all measurements arrive. Hence, at the OOSM arrival times the performance of RBPF is attainable, for a sufficiently large number of particles. Table II presents the time-averaged RMSE values at the OOSM arrival times (i.e., $k = 1, 3, \ldots, 99$ and $k = 1, 4, \ldots, 100$, respectively) for η_k and θ_k. SEPFFPS is omitted because of inadequate handling of multimodal distributions. The tracking performance of SERBPF is superior to both RBOOSMBS and A-PF for $l = 1$. This is because for $l = 1$, the smoothing in SERBPF yields a better approximation than only using $M_{\mathrm{BS}} = 1$ in RBOOSMBS. For SERBPF, using $M_{\mathrm{FF}} = 0.1N$ only gives a minor improvement compared with discarding the OOSMs. However, increasing it to $M_{\mathrm{FF}} \approx 0.4N$ renders similar performance as obtained for $M_{\mathrm{FF}} = N$. It is interesting that SERBPF with $M_{\mathrm{FF}} = 0.4N$ performs better than RBOOSMBS with $M_{\mathrm{BS}} = 1$. In this example, setting $D < N$ in RBOOSMBS gives deficient performance.

For $l = 2$, RBOOSMBS and SERBPF have similar tracking performance, both for η_k and θ_k. Fig. 7 gives a visualization of the RMSE values for the filters. The zig-zag type shapes occur because the delay is fixed. There are two reasons for RBOOSMBS not consistently performing better than SERBPF. The first is that the smoothing approach in SERBPF handles this delay well. The second, and perhaps most important, reason is that this is a demanding estimation problem where RBOOSMBS benefits from using more than one backward trajectory per forward particle. This is also indicated by both approaches being more sensitive to the values of D and M_{FF}. Note that for the fifth-order model given by (40)–(42), $N = 400$ particles is not enough to consistently approximate the (bimodal) filtering posterior with high accuracy. Thus, the increased sensitivity in D and M_{FF} can partly be explained by this, both for $l = 1$ and $l = 2$. Still, the estimation accuracy of SERBPF for $M_{\mathrm{FF}} = 0.4N$ is close to the performance for $M_{\mathrm{FF}} = N$, and indicates that SERBPF is robust to increased OOSM delay.

C. Example 3

The third example deals with estimating the states of an aircraft using a planar constant-acceleration process model [40]. The states of interest are the position $p_k \in \mathbb{R}^2$, velocity $v_k \in \mathbb{R}^2$, and acceleration $a_k \in \mathbb{R}^2$, with the state vector

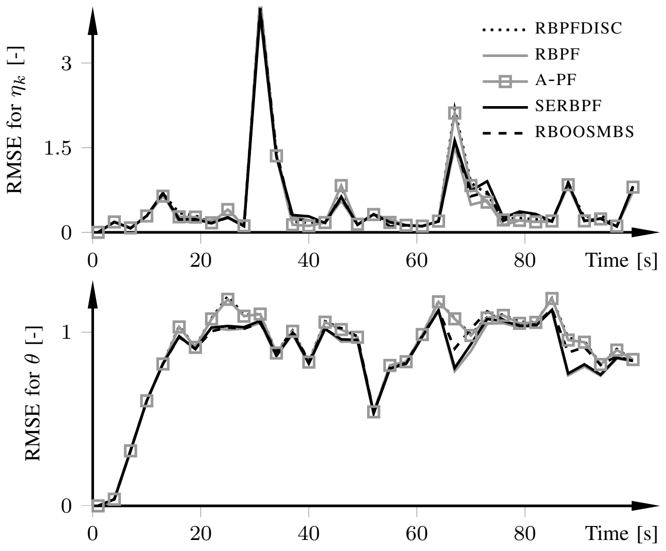

Fig. 7. RMSE values using $l = 2$ in Example 2, corresponding to Table II. All algorithms have distinct peaks in the RMSEs, which occur because of lack of measurements at times when η_k changes sign.

$$x_{k+1} = \begin{pmatrix} p^X_{k+1} & p^Y_{k+1} & v^X_{k+1} & v^Y_{k+1} & a^X_{k+1} & a^Y_{k+1} \end{pmatrix}^{\mathrm{T}}.$$

The measurements are the range r_k and bearing ϕ_k from the radar system to the aircraft. The nonlinearities only enter in the measurement equations. With $T_s = 1$ s, the model is

$$\begin{aligned} x_{k+1} &= Ax_k + \bar{v}_k, \\ y_k &= \begin{pmatrix} r_k \\ \phi_k \end{pmatrix} = \begin{pmatrix} \sqrt{(p^X_k)^2 + (p^Y_k)^2} \\ \arctan\left(\frac{p^Y_k}{p^X_k}\right) \end{pmatrix} + e_k. \end{aligned} \tag{43}$$

The process noise $\bar{v}_k \in \mathbb{R}^6$ is white and Gaussian distributed as $\bar{v}_k \sim \mathcal{N}(0, Q)$, $Q = BQ_{\bar{v}}B^{\mathrm{T}}$, where the system matrix A in (43) and B are given by

$$A = \left(\begin{array}{c|cc} I_{2\times 2} & I_{2\times 2} & 0.5I_{2\times 2} \\ \hline 0_{2\times 2} & I_{2\times 2} & I_{2\times 2} \\ 0_{2\times 2} & 0_{2\times 2} & I_{2\times 2} \end{array}\right), \tag{44a}$$

$$B = \left(\begin{array}{ccc} I_{2\times 2} & 0.5I_{2\times 2} & 0.17I_{2\times 2}, \\ \hline 0_{2\times 2} & I_{2\times 2} & 0.5I_{2\times 2} \\ 0_{2\times 2} & 0_{2\times 2} & I_{2\times 2} \end{array}\right), \tag{44b}$$

where $0_{2\times 2}$ is the 2×2 zero matrix. Furthermore, $Q_{\bar{v}} = \mathrm{diag}(4, 4, 4, 4, 0.01, 0.01)$. This model can be written on the form (1), where the lines in (44) indicate the partition into nonlinear and linear states. Note that the linear and nonlinear states show up in reversed order compared with the standard formulation (1). The structure of B in (44b) gives a cross-correlation between the process noise acting on the linear states and the nonlinear states, respectively—that is, $Q^{z\eta}$ in (2) is nonzero. The measurement noise e_k is distributed as $\mathcal{N}(0, R_k)$, with $R_k = \mathrm{diag}(100, 10^{-6})$. The communication with the bearing measurements (i.e., the second element of y_k) is corrupted. On average only 50% of the packets arrive, and those packages that eventually arrive are delayed between $[0, 3]$ samples according to a discrete uniform distribution.

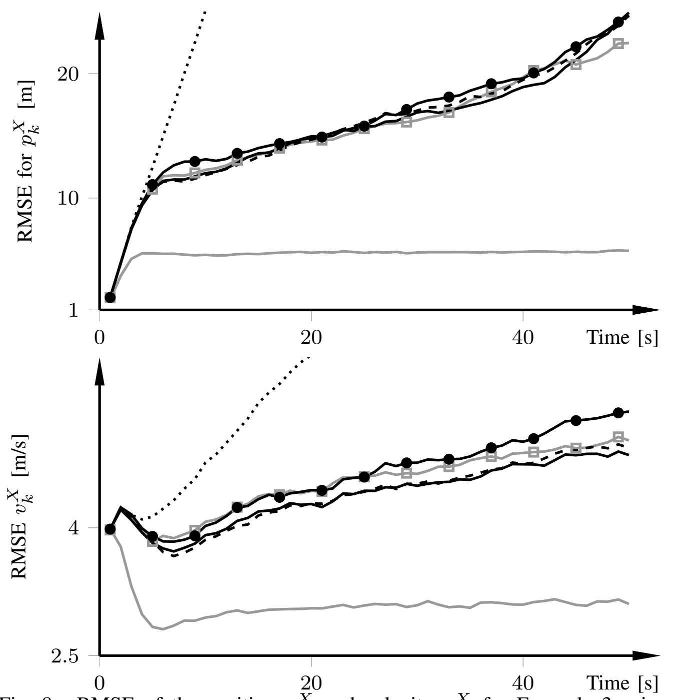

Fig. 8. RMSE of the position p_k^X and velocity v_k^X for Example 3 using 5000 Monte-Carlo simulations, with same notation as in Fig. 4. All filters use $N = 200$ particles and M_{FF} is set to $M_{\text{FF}} = 0.2N$, $D = 0.25N$, and $M_{\text{BS}} = 1$. The results for p_k^Y and v_k^Y are similar because of symmetry.

Fig. 8 shows the RMSE for the position p_k^X and velocity v_k^X for 5000 Monte-Carlo simulations using 200 particles, with initial estimate and covariance matrix for all filters set to

$$x_0 = \begin{pmatrix}2000 & 2000 & 10 & 10 & 0 & 0\end{pmatrix}^{\mathrm{T}},$$
$$P_0 = \operatorname{diag}(4, 4, 16, 16, 0.01, 0.01).$$

Again, the proposed filters perform very well. Compared with the other examples, the filter that associates the OOSMs with the current states using an extended Kalman smoother, SEPFFPS, is now closer in performance to the other filters. This is unsurprising, given that the state dynamics is linear; hence, the Kalman smoother will only be approximate in the measurement-update steps. For this example, especially for the position estimates, SERBPF with $M_{\text{FF}} = 0.2N$ performs better than the rest of the OOSM filters. We have tried various different noise settings in combination with different initial estimates and number of particles, and although the backward-simulation based approach (RBOOSMBS) most often performs best, SERBPF often yields very similar performance. A reason that RBOOSMBS does not consistently outperform SERBPF is the linear process model; it is possible to use very few particles for the smoother in SERBPF and still obtain accurate estimation accuracy when compared with only using one backward trajectory in RBOOSMBS.

The computation time for one time step when the OOSM delay is $l = 3$ s is shown in Table IV. To reduce the effects of memory management and operating-system intervention, Table IV displays the minimum execution time, thus differing to the approach used for the results in Fig. 6 where the dependence on N was of interest. SEPFFPS is fastest, but the computation time for SERBPF is competitive, especially considering that the code is not optimized. In a real-time implementation, the improved tracking performance of SERBPF has to be weighed against the lower execution time of SEPFFPS.

TABLE III
TIME-AVERAGED RMSE VALUES CORRESPONDING TO FIG. 8.

Algorithm	p^X	p^Y	v^X	v^Y
RBPFDISC	64.4	64.6	6.8	6.8
SEPFFPS	16.2	16.2	4.6	4.6
A-PF	15.6	15.8	4.5	4.5
SERBPF, $M_{\text{FF}} = 0.2N$	**15.5**	**15.6**	4.4	4.4
RBOOSMBS, $D = 0.25N$	15.7	15.7	**4.3**	**4.4**
RBPF	5.5	5.5	3.1	3.1

TABLE IV
MINIMUM COMPUTATION TIME FOR ONE TIME STEP, USING SYSTEM (43) WITH OOSM DELAY SET TO $l = 3$ S. IT IS INTERESTING TO COMBINE THE COMPUTATION TIME WITH THE RMSES IN TABLE III AND FIG. 8.

	$N = 200$
Algorithm	Time [s]
A-PF	1.6
RBOOSMBS, $D = N$	0.6
SERBPF, $M_{\text{FF}} = 0.2N$	$3 \cdot 10^{-3}$
SEPFFPS	$6 \cdot 10^{-4}$

Fig. 9 presents the performance difference when comparing RBOOSMBS that is derived in this paper with the version that appeared in our preliminary conference-paper version [15]. Accounting for the cross-correlation between the linear and nonlinear states gives increased tracking performance, and the difference increases with time.

VII. Discussion

For the considered examples and delays, it was often sufficient to only use one backward trajectory per particle in RBOOSMBS and still get excellent estimation accuracy. This is most probably caused by the trajectories in the smoothing iterations being fixed to the forward particle at the end point, thus reducing the number of possible distinct backward trajectories. The performance of RBOOSMBS also turned out to be rather insensitive to the choice of D when compared with having $D = N$. However, for some cases it is essential that $D \approx N$, for example, when the number of forward particles is small and/or the posterior has a multimodal behavior, as is the case for the example in Sec. VI-B.

From our experience the number of iterations in the while-loop in Algorithm 2 depends heavily on the proposal distribution. If the proposal distribution mimics the target distribution well, an index will often be found after a few iterations only. Hence, in those cases it can be worthwhile to allow for a large $C_{\max}$. However, if the proposal distribution differs a lot from the target distribution, $C_{\max}$ is preferably set to a small value since the maximum number of iterations will probably be reached anyway. In our simulations we, quite arbitrarily, set the limit to $C_{\max} = D/4$, but the choice is problem dependent. Although $C_{\max}$ was reached in some cases, the rejection sampling on average provided a noticeable speedup compared with explicitly calculating all smoothing weights.

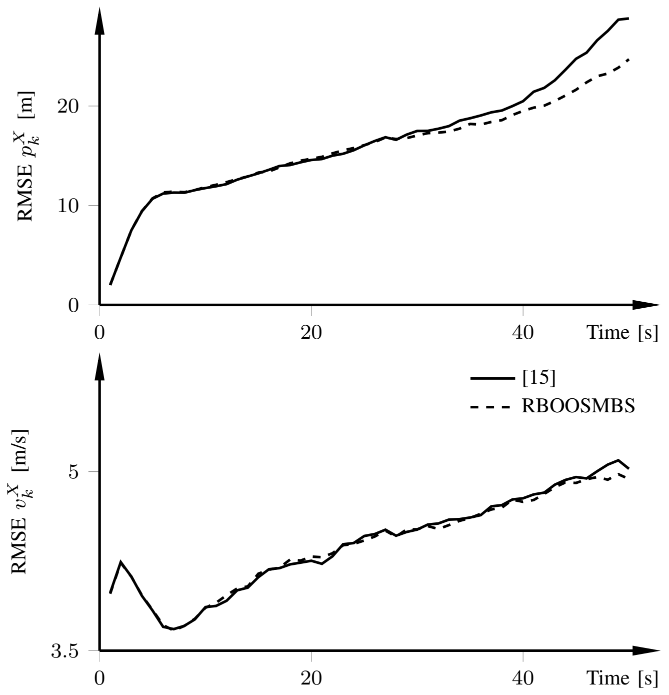

Fig. 9. A comparison between the RMSE of our proposed RBOOSMBS and the version that appeared in our previous paper [15].

A suitable number of particles in the supporting RBPF in SERBPF considering the tradeoff between performance and computation time, turned out to be $0.05N \leq M_{\text{FF}} \leq 0.4N$ depending on noise levels and the number of forward particles used. When N is large the value of M_{FF} can typically be chosen relatively smaller. The reason is that the filter weights are more accurate; hence, when sampling from the filter weights at time index τ, more consistent samples are drawn. Although the choice of M_{FF} was more insensitive than the choice of D in RBOOSMBS, sometimes the tracking performance deteriorated for small M_{FF}—for example, for the problem in Sec. VI-B. Hence, the choice is therefore also dependent on the degree of multimodality in the posterior. In many practical applications, however, we believe that the number of particles used in the smoothing iterations in both SERBPF and RBOOSMBS can be drastically reduced without sacrificing estimation accuracy. This is emphasized by the results in Sec. VI-A and Sec. VI-C, where the model structure is similar to those that are used in many tracking applications. Consequently, both algorithms are feasible alternatives in online implementations, and we expect that SERBPF can estimate full-scale problems online.

VIII. Conclusions

We derived and presented two new algorithms for OOSM processing considering the class of mixed linear/nonlinear state-space models. Both use Rao-Blackwellization to exploit the conditionally linear Gaussian substructure in the model. One of them yields fast execution and the other focuses on estimation accuracy. Simulation examples showed that both approaches yield improvements in terms of RMSE when compared with recent particle-filter algorithms for OOSM processing. In several examples the respective performance of the proposed algorithms was similar to an optimal filter. SERBPF is the most viable option when computation time is a concern. If high-performance estimation is wanted, RBOOSMBS is the preferred choice. Both out-of-sequence measurements and the considered model class are common in target tracking, positioning, and navigation scenarios. The developed algorithms therefore enable performance improvements in relevant filtering applications.

Appendix

Proof of Proposition 1. Follows by applying results in [9] of how to compute expected means and covariances in RBPFs and in [23] of how to update linear states with OOSMs. ☐

Proof of Proposition 2. It holds that

$$p(z_m|\eta_{m:k}, y_{0:k}) = \int p(z_m, z_{m+1}|\eta_{m:k}, y_{0:k})\,\mathrm{d}z_{m+1}.$$

Now, by using Bayes' rule and the Markov property of the linear states, we end up with

$$p(z_m|\eta_{m:k}, y_{0:k}) = p(z_m|\eta_{m:m+1}, y_{0:m}) \times \int \frac{p(z_{m+1}|z_m, \eta_{m:m+1}) p(z_{m+1}|\eta_{m:k}, y_{0:k})}{p(z_{m+1}|\eta_{m:m+1}, y_{0:m})}\,\mathrm{d}z_{m+1}.$$

The density $p(z_{m+1}|\eta_{m:k}, y_{0:k})$ is given by the previous smoothing step. The rest is analogous to the constrained Kalman-filter derivations for the RBPF in [9]. ☐

References

[1] M. M. Muntzinger, M. Aeberhard, S. Zuther, M. Maehlisch, M. R. Schmid, and K. Dietmayer, "Reliable automotive pre-crash system with out-of-sequence measurement processing," in *IEEE Intelligent Vehicles Symp.*, San Diego, CA, June 2010.

[2] K. Berntorp, K.-E. Årzén, and A. Robertsson, "Sensor fusion for motion estimation of mobile robots with compensation for out-of-sequence measurements," in *11th Int. Conf. Control, Automation, and Systems*, KINTEX, Gyeonggi–do, Korea, Oct 2011.

[3] Z. Jia, A. Balasuriya, and S. Challa, "Sensor fusion-based visual target tracking for autonomous vehicles with the out-of-sequence measurements solution," *Robotics and Autonomous Systems*, vol. 56, no. 2, pp. 157–176, 2008.

[4] A. Ranganathan, M. Kaess, and F. Dellaert, "Fast 3D pose estimation with out-of-sequence measurements," in *2007 IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, San Diego, CA, Oct 2007.

[5] A. Murtra, J. Mirats Tur, and A. Sanfeliu, "Integrating asynchronous observations for mobile robot position tracking in cooperative environments," in *2009 IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, St. Louis, MO, Oct 2009.

[6] E. Besada-Portas, J. A. Lopez-Orozco, P. Lanillos, and J. M. de la Cruz, "Localization of non-linearly modeled autonomous mobile robots using out-of-sequence measurements," *Sensors*, vol. 12, no. 3, pp. 2487–2518, 2012.

[7] F. Xue, Z. Liu, and X. Zhang, "Out-of-sequence measurements processing based on unscented particle filter for passive target tracking in sensor networks," in *8th Int. Conf. Signal Process.*, Beijing, China, Nov 2006.

[8] S. R. Maskell, R. G. Everitt, R. Wright, and M. Briers, "Multi-target out-of-sequence data association: Tracking using graphical models," *Information Fusion*, vol. 7, no. 4, pp. 434–447, 2006.

[9] T. B. Schön, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear nonlinear state-space models," *IEEE Trans. Signal Process.*, vol. 53, pp. 2279–2289, 2005.

[10] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 425–437, 2002.

[11] X. Rong Li and V. P. Jilkov, "Survey of maneuvering target tracking . part I: dynamic models," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, 2003.
[12] X. R. Li and V. P. Jilkov, "A survey of maneuvering target tracking - part III: Measurement models," in *2001 SPIE Conf. Signal and Data Process. of Small Targets*, San Diego, CA, July 2001.
[13] S. Zhang and Y. Bar-Shalom, "Out-of-sequence measurement processing for particle filter: Exact Bayesian solution," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 4, pp. 2818–2831, 2012.
[14] U. Orguner and F. Gustafsson, "Storage efficient particle filters for the out of sequence measurement problem," in *11th Int. Conf. Information Fusion*, Cologne, Germany, June 2008.
[15] K. Berntorp, A. Robertsson, and K.-E. Årzén, "Rao-Blackwellized out-of-sequence processing for mixed linear/nonlinear state-space models," in *16th Int. Conf. Information Fusion*, Istanbul, Turkey, July 2013.
[16] S. Blackman and R. Popoli, *Design and analysis of modern tracking systems*, ser. Artech House radar library. Artech House, 1999.
[17] Y. Bar-Shalom, H. Chen, and M. Mallick, "One-step solution for the multistep out-of-sequence-measurement problem in tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, no. 1, pp. 27–37, 2004.
[18] S. Zhang, Y. Bar-Shalom, and G. Watson, "Tracking with multisensor out-of-sequence measurements with residual biases," in *13th Int. Conf. Information Fusion*, Edinburgh, UK, July 2010.
[19] Y. Bar-Shalom, "Update with out-of-sequence measurements in tracking: exact solution," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 38, no. 3, pp. 769–777, 2002.
[20] K. Zhang, X. Li, and Y. Zhu, "Optimal update with out-of-sequence measurements," *IEEE Trans. Signal Process.*, vol. 53, no. 6, pp. 1992–2004, 2005.
[21] W. Koch, "On accumulated state densities with applications to out-of-sequence measurement processing," in *12th Int. Conf. Information Fusion*, Seattle, WA, July 2009.
[22] X. Shen, Y. Zhu, E. Song, and Y. Luo, "Optimal centralized update with multiple local out-of-sequence measurements," *IEEE Trans. Signal Process.*, vol. 57, no. 4, pp. 1551–1562, 2009.
[23] S. Zhang and Y. Bar-Shalom, "Optimal update with multiple out-of-sequence measurements with arbitrary arriving order," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 48, no. 4, pp. 3116–3132, 2012.
[24] F. Govaers and W. Koch, "A generalized solution to smoothing and out-of-sequence processing," in *15th Int. Conf. Information Fusion*, Singapore, July 2012.
[25] M. Orton and A. Marrs, "A Bayesian approach to multi-target tracking and data fusion with out-of-sequence measurements," *IEE Seminar Digests*, vol. 2001, no. 174, 2001.
[26] ——, "Particle filters for tracking with out-of-sequence measurements," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, no. 2, pp. 693–702, 2005.
[27] M. Mallick, T. Kirubarajan, and S. Arulampalam, "Out-of-sequence measurement processing for tracking ground target using particle filters," in *IEEE Aerosp. Conf.*, Big Sky, MT, Mar 2002.
[28] X. Liu, B. Oreshkin, and M. Coates, "Efficient delay-tolerant particle filtering through selective processing of out-of-sequence measurements," in *13th Int. Conf. Information Fusion*, Edinburgh, UK, July 2010.
[29] B. Oreshkin, X. Liu, and M. Coates, "Efficient delay-tolerant particle filtering," *IEEE Trans. Signal Process.*, vol. 59, no. 7, pp. 3369–3381, 2011.
[30] K. Berntorp, K.-E. Årzén, and A. Robertsson, "Storage efficient particle filters with multiple out-of-sequence measurements," in *15th Int. Conf. Information Fusion*, Singapore, July 2012.
[31] S. Särkkä, A. Vehtari, and J. Lampinen, "Rao-Blackwellized Monte Carlo data association for multiple target tracking," in *7th Int. Conf. Information Fusion*, Stockholm, Sweden, June 2004.
[32] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*. Prentice Hall, 1979.
[33] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, 2002.
[34] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.
[35] F. Gustafsson, "Particle filter theory and practice with positioning applications," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 25, no. 7, pp. 53–82, 2010.
[36] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *Radar and Signal Processing, IEE Proceedings F*, vol. 140, no. 2, pp. 107–113, Apr. 1993.
[37] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *J. of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.
[38] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *J. of the American Statistical Association*, vol. 99, pp. 156–168, 2004.
[39] F. Lindsten, P. Bunch, S. J. Godsill, and T. B. Schön, "Rao-Blackwellized particle smoothers for mixed linear/nonlinear state-space models," in *38th Int. Conf. Acoustics, Speech, and Signal Process.*, Vancouver, Canada, May 2013.
[40] R. Karlsson, T. Schön, and F. Gustafsson, "Complexity analysis of the marginalized particle filter," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 53, no. 11, pp. 4408–4411, 2005.
[41] R. Douc, A. Garivier, E. Moulines, and J. Olsson, "Sequential Monte Carlo smoothing for general state space hidden Markov models," *Annals of Applied Probability*, vol. 21, pp. 2109–2145, 2012.
[42] H. E. Rauch, C. T. Striebel, and F. Tung, "Maximum likelihood estimates of linear dynamic systems," *J. of the American Institute of Aeronautics and Astronautics*, vol. 3, no. 8, pp. 1445–1450, Aug. 1965.
[43] F. Lindsten and T. B. Schön, "Identification of mixed linear/nonlinear state-space models," in *49th IEEE Conf. Decision and Control*, Grand Wailea, Maui, Hawaii, Dec 2010.

Karl Berntorp received the M.Sc. degree in Engineering Physics in 2009 and the Ph.D. degree in Automatic Control in 2014, both from Lund University, Lund, Sweden. He is currently with the Mechatronics group at Mitsubishi Electric Research Laboratories in Cambridge, MA. His current research interests are in nonlinear estimation and control, path planning, motion control, and their applications to automotive, robotics, and aerospace systems.

Anders Robertsson received the M.Sc. degree in Electrical Engineering and the Ph.D. degree in Automatic Control from LTH, Lund University, Lund, Sweden, in 1992 and 1999, respectively. He was appointed Docent in 2005 and Excellent Teaching Practitioner in 2007. He is since January 2012 full Professor at the Department of Automatic Control, LTH, Lund University. His current research interests are in nonlinear estimation and control, robotics research on mobile and industrial manipulators, and feedback control of computing systems such as cloud infrastructures.

Karl-Erik Årzén received the M.Sc. degree in Electrical Engineering from Lund University, Lund, Sweden in 1981 and the Ph.D. degree in Automatic Control also from Lund Univerisity in 1987. He has been a full Professor with the Department of Automatic Control, Lund University, since 2000. In the beginning of the 1990s he worked for ABB Corporate Research. His current research interests are embedded real-time control systems, and control applied to computing systems, in particular embedded systems and cloud infrastructures. Prof Årzén is a member of IEEE and of the Royal Swedish Academy of Engineering Sciences. In 2006 he received the Dr. Guido Carlo-Stella award for his contributions to the field of process automation and information integration in the manufacturing industry.