Hostettler, Roland; Särkkä, Simo

**Rao-Blackwellized Gaussian Smoothing**

*Published in:* IEEE Transactions on Automatic Control

*DOI:* 10.1109/TAC.2018.2828087

Published: 01/01/2019

*Document Version*
Peer reviewed version

*Please cite the original version:*
Rao–Blackwellized Gaussian Smoothing

Roland Hostettler and Simo Särkkä

This is a pre-print of a paper accepted for publication in IEEE Transactions on Automatic Control. When citing this work, you must always cite the original article:


DOI:
10.1109/TAC.2018.2828087

Copyright:
Copyright 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.
Rao–Blackwellized Gaussian Smoothing

Roland Hostettler, Member, IEEE, and Simo Särkkä, Senior Member, IEEE

Abstract—In this paper, we consider Rao–Blackwellization of linear substructures in sigma-point-based Gaussian assumed density smoothers. We derive marginalized prediction, smoothing, and update steps for the mixed linear/nonlinear Gaussian state-space model as well as for a hierarchical model for both conventional and iterated posterior linearization Gaussian smoothers. The proposed method is evaluated in a numerical example and it is shown that the computational complexity is reduced considerably compared to non-Rao–Blackwellized Gaussian smoothers for systems with high-dimensional linear subspaces.

Index Terms—Gaussian assumed density smoothing, Rao–Blackwellization, nonlinear smoothing, nonlinear state estimation

I. INTRODUCTION

Gaussian assumed density filtering and smoothing are popular approaches to approximate Bayesian filtering and smoothing in nonlinear state-space systems. In these methods, the filtering density
\[ p(\mathbf{x}_t \mid y_{1:t}) \]
(where \( \mathbf{x}_t \) is the state vector and \( y_{1:t} = \{y_1, y_2, \ldots, y_t\} \) are the measurements) and the smoothing density \( p(\mathbf{x}_t \mid y_{1:T}) \) (for \( 1 \leq t \leq T \)) are approximated as Gaussian densities (see, e.g., [11]–[3]). These approaches have the advantage that only estimating the first two moments of the posterior densities is required, similarly to the Kalman filter and Rauch–Tung–Striebel smoother [3]. The resulting moment matching integrals can be solved numerically using different types of sigma-point methods such as the third and higher order unscented transforms [4]–[9], Gauss–Hermite quadratures [1], [2], or spherical cubatures [4], [10], [11]. In many applications, these approaches are superior to simple Taylor series-based linearizations [12], [13].

A drawback of sigma-point methods is that even when the state-space model exhibits a linear substructure, the numerical integration is performed over the whole state-space. This wastes computational resources by numerically integrating the analytically tractable subspaces. The aim of this article is to use Rao–Blackwellization to reduce this computational burden of sigma-point-based Gaussian smoothers. The use of Rao–Blackwellization is common in the context of sequential Monte Carlo (SMC) methods (particle filters and smoothers) that approximate the posterior densities using a set of weighted samples rather than an assumed density [14]–[17]. In SMC methods Rao–Blackwellization is used to reduce the number of samples required for approximating the posterior density.

Rao–Blackwellization has also been considered in Gaussian filtering. For example, Rao–Blackwellized unscented Kalman filters for non-mixing and completely mixing conditional linear Gaussian state-space models were introduced in [18] and [19]. Furthermore, [20] discussed models with directly and indirectly observed subsets of state variables as well as linear and nonlinear observations. Similarly, [21] considered models where only part of the state is observed nonlinearly and proposes a truncated unscented Kalman filter. A more generalized approach was presented in [22], where it is assumed that the problem exhibits a generic conditionally analytically tractable (with respect to the assumed Gaussian density) substructure. A similar approach was introduced in [23] in the context of Gauss–Hermite filtering. A marginalized UKF for correlated process and measurement noises was developed in [24]. Finally, a unified view on marginalized Gaussian filtering from a subspace projection perspective was introduced in [25]. The contribution of this paper is to derive novel Rao–Blackwellized Gaussian smoothing algorithms for two general classes of models recently considered in the context of Rao–Blackwellized sequential Monte Carlo smoothers [17]. As the model classes are more general than what has previously been considered in the context of Rao–Blackwellized Gaussian filtering algorithms, the results also extend the existing results for the Rao–Blackwellized Gaussian filters. Specifically, the technical contributions of this paper are as follows. First, we derive statistical linear regression for conditionally affine transformations of Gaussian random variables. Second, we develop Rao–Blackwellized Gaussian smoothers and Rao–Blackwellized posterior linearization smoothers [26] for two commonly encountered models, the fully mixing linear/nonlinear state-space model and a hierarchical model [17]. Third, we analytically analyze the reduction of the computational complexity in the proposed method. Fourth, the proposed methods are evaluated and compared to their non-Rao–Blackwellized counterparts as well as sequential Monte Carlo methods.

II. PROBLEM FORMULATION

Consider the general nonlinear state-space system
\[
\mathbf{x}_t = f(\mathbf{x}_{t-1}) + q_t,
\]
\[
y_t = g(\mathbf{x}_t) + r_t,
\]
where \( \mathbf{x}_t \in \mathbb{R}^{n_x} \) is the latent state vector with initial density \( \mathcal{N}(\mathbf{x}_0; \mathbf{0}, \mathbf{P}_0) \), \( q_t \sim \mathcal{N}(0, \mathbf{Q}_t) \) the Gaussian process noise, \( f(\cdot) \) the dynamic model function, \( y_t \in \mathbb{R}^{n_y} \) the measurement, \( r_t \sim \mathcal{N}(0, \mathbf{R}_t) \) the measurement noise, \( g(\cdot) \) the measurement function, and \( t \) denotes the discrete time index. Then, the objective of Gaussian smoothing is to find a Gaussian approximation to the smoothing density
\[
p(\mathbf{x}_t \mid y_{1:T}) \approx \mathcal{N}(\mathbf{x}_t; \hat{\mathbf{x}}_t | y_{1:T}, \mathbf{P}_t | y_{1:T}).
\]

In this work, two particular subclasses of the general model (1) are considered. In both of these classes, the state space can be split into nonlinear and linear subspaces \( \mathbf{x}_t^\nu \in \mathbb{R}^{n_{x^\nu}} \) and \( \mathbf{x}_t^l \in \mathbb{R}^{n_{x^l}} \) such that \( \mathbf{x}_t = [(\mathbf{x}_t^\nu)^T, (\mathbf{x}_t^l)^T]^T \), which yields an analytically tractable substructure. The first model, Model 1, is a commonly encountered (see, e.g., [27]–[29]) mixed linear/nonlinear Gaussian state-space model defined as follows.

Model 1 (Mixed linear/nonlinear Gaussian State-Space Model). The mixed linear/nonlinear Gaussian state-space model is defined as
\[
x_t^\nu = f_t^\nu(x_{t-1}^\nu) + A_t^\nu(x_{t-1}^\nu)x_{t-1}^l + q_t^\nu,
\]
\[
x_t^l = f_t^l(x_{t-1}^l) + A_t^l(x_{t-1}^l)x_{t-1}^l + q_t^l,
\]
\[
y_t = g_t(x_t^\nu) + B_t^l(x_t^\nu)x_t^l + r_t.
\]

The noise terms \( q_t^\nu, q_t^l \), and \( r_t \) are independent zero-mean, Gaussian random variables with \( \text{Cov}(q_t^\nu) = Q_t^\nu(x_{t-1}^\nu), \text{Cov}(q_t^l) = Q_t^l(x_{t-1}^l) \), \( \text{Cov}(q_t^\nu, q_t^l) = Q_t^{\nu l}(x_{t-1}^\nu), \text{and} \ \text{Cov}(r_t) = R_t(x_t^\nu) \), respectively, and the initial state is Gaussian according to \( \mathbf{x}_0 = \mathcal{N}(\mathbf{x}_0; \hat{x}_0^\nu, P_0^\nu) \).

In this model, the dynamics are mixing, that is, both the nonlinear and linear states may affect each other (see (3a)–(3b)). Furthermore, the nonlinear states are observed through some nonlinear function \( g_t(\cdot) \)
while the linear states are observed through the measurement matrix \( B_t(\cdot) \). This type of model is frequently encountered in applications such as target tracking, where, for example, the nonlinear states are the target’s position that are observed nonlinearly (e.g., range and bearing measurements) and the linear states are the target’s velocity [28].

The second model is the hierarchical model given below [30–32].

**Model 2 (Hierarchical Model).** The hierarchical model is defined as

\[
x_t^n = f_t^n(x_{t-1}^n) + q_t^n, \\
x_t^l = f_t^l(x_t^l) + A_t^l(x_t^l)x_{t-1}^l + q_t^l, \\
y_t = g_t(x_t^l) + B_t(x_t^l)x_t^l + r_t,
\]

where \( q_t^n \sim p(q_t^n) \) with \( E(q_t^n) = 0 \) and \( \text{Cov}(q_t^n) = Q^n(x_{t-1}^n) \), \( q_t^l \sim N(0, Q^l(x_t^l)) \).\( \text{Cov}(q_t^l, q_t^l) = 0 \), \( r_t \sim N(0, R(x_t^n)) \), and

\[
p(x_0) = N(x_0; \hat{x}_{0|0}, P_{0|0}).
\]

Model 2 can be thought of as a generalized jump Markov model. Here, the linear and nonlinear states do not mix fully: the nonlinear state affects the linear state but not vice versa. Note that Model 2 is not a special case of Model 1: the difference lies in the linear dynamics (4b) that depend on \( x_t^l \) rather than \( x_{t-1}^n \), while \( y_t \) depends on the current state \( x_t \) in both models. Also note that for Model 2, the state dynamics for \( x_t^l \) are sometimes given in terms of the transition density \( x_t^n \sim p(x_t^n | x_{t-1}^n) \) (see [32]). Here, the functional form is chosen since it will simplify the derivations later on.

Given these two models, the aim is then to find Gaussian approximations of the form (2), taking the analytically tractable substructure into account. Note that the explicit dependence of \( f, A, Q, g, B, \) and \( R \) on \( t \) and \( x_t^n \) is omitted for the remainder of this paper.

### III. Gaussian Smoothing

In this section, Gaussian filtering, smoothing, and posterior linearization smoothing are briefly reviewed. First, note that the Rao–Tung–Striebel (RTS) smoothing recursion [3], [33] is given by a recursion backward in time and found from the joint approximation

\[
p(x_t, x_{t+1} \mid y_{1:T}) = p(x_t \mid x_{t+1}, y_{1:T})p(x_{t+1} \mid y_{1:T}) \\
\approx N \left( \begin{bmatrix} x_t^l \\ x_{t+1}^l \end{bmatrix}, \begin{bmatrix} P_t^l & E_t \\ E_t^T & P_{t+1|T} \end{bmatrix} \right),
\]

by marginalizing with respect to \( x_{t+1}^l \). The density \( p(x_t \mid x_{t+1}, y_{1:T}) \) is obtained from the joint-approximation \( p(x_t, x_{t+1} \mid y_{1:T}) \) during the prediction step of a Gaussian filter. This is given by [3]

\[
p(x_t, x_{t+1} \mid y_{1:T}) \\
\approx N \left( \begin{bmatrix} x_t^l \\ x_{t+1}^l \end{bmatrix}, \begin{bmatrix} P_t^l & C_t \\ C_t^T & P_{t+1|T} \end{bmatrix} \right).
\]

Furthermore, since the recursion is backwards in time, a Gaussian approximation of the form (2) is given for \( p(x_{t+1} \mid y_{1:T}) \). Then, the well-known RTS smoothing equations [3], [33] are found to be

\[
G_t = C_t P_{t+1|T}^{-1}, \\
\hat{x}_{t|T} = \hat{x}_{t|t} + G_t (\hat{x}_{t+1|T} - \hat{x}_{t+1|t}), \\
P_{t|T} = P_t + G_t (P_{t+1|T} - P_{t+1|t}) G_t^T, \\
E_t = \hat{G}_t P_{t+1|T}.
\]

In order to find the approximation (5), a Gaussian filter has to be run in forward direction which provides a Gaussian approximation of the one step ahead prediction density \( p(x_t \mid y_{1:t-1}) \). The approximation of the filtering density \( p(x_t \mid y_{1:t-1}) \) is found from the joint approximation

\[
p(x_t, y_t \mid y_{1:t-1}) \approx N \left( \begin{bmatrix} x_t^l \\ y_t \end{bmatrix}, \begin{bmatrix} P_{t|t-1}^l & D_t \\ D_t^T & S_t \end{bmatrix} \right),
\]

and subsequent conditioning on \( y_t \) (see, e.g., [3] for details). This yields the well-known Kalman filter update given by

\[
K_t = D_t S_t^{-1}, \\
\hat{x}_{t|t} = \hat{x}_{t|t-1} + K_t (y_t - \hat{y}_{t|t-1}), \\
P_{t|t} = P_{t|t-1} - K_t S_t K_t^T.
\]

Note that the smoothing recursion (6) not only depends on the means \( \hat{x}_{t|t}, \hat{x}_{t+1|T}, \hat{x}_{t|T} \) and their respective covariances, but also the cross-covariance \( C_t \). The latter is not required for filtering alone but is calculated during the prediction step (see (5)).

The Gaussian approximations in (5) and (7) require us to calculate the unknown moments \( \hat{x}_{t+1|T}, P_{t+1|T}, \) and \( C_t \) as well as \( \hat{y}_{t|t-1}, D_t, \) and \( S_t \), respectively. These can be found by approximating the nonlinear state transition and measurement functions \( f(\cdot) \) and \( g(\cdot) \) by using statistical linear regression (SLR) [34]. In regular one-pass Gaussian smoothing (e.g., [2]), SLR is performed with respect to the prior densities \( p(x_{t-1} \mid y_{1:t-1}) \) for the prediction and \( p(x_t \mid y_{1:t-1}) \) for the measurement update. In contrast to regular Gaussian smoothing, the recently proposed posterior linearization smoothing approach [26] tries to linearize with respect to the posterior \( p(x_t \mid y_{1:t}) \) directly. This can lead to significant performance gain in terms of the error in cases where the prior and posterior overlap poorly. Since the posterior is unknown to start with, the following iterative scheme can be used to gradually obtain improved approximations of the posterior [26]. First, regular smoothing is used to obtain an initial approximation of the posterior \( p(x_t \mid y_{1:T}) \approx N(\hat{x}_{t|T}, \hat{P}_{t|T}) \). Then, the nonlinear functions \( f(\cdot) \) and \( g(\cdot) \) are linearized using SLR with respect to \( p(x_t \mid y_{1:T}) \) and smoothing is done anew in order to obtain \( p(x_t \mid y_{1:T}) \). The process is repeated for a predefined number of iterations or until convergence is achieved.

### IV. Rao–Blackwellized Gaussian Smoothing

In this section, the Rao–Blackwellized Gaussian smoothing algorithms for the two models discussed in Section II will be developed. First, Rao–Blackwellized statistical linear regression of conditionally affine transformations of Gaussian random variables is derived. This is then applied to Gaussian filtering and smoothing to obtain the Rao–Blackwellized prediction and measurement update steps. Note that similar to the state vector, the covariance matrices can be partitioned too, such that \( p(x_{1:T}|y_{1:T}) \)

\[
\begin{bmatrix} p_{x_{1:T}|y_{1:T}} \\ \mu_{x_{1:T}} \\ \Sigma_{x_{1:T}} \end{bmatrix} = \begin{bmatrix} P_{x_{1:T}|y_{1:T}} \\ P_{x_{1:T}} \end{bmatrix} \begin{bmatrix} \mu_{x_{1:T}} \\ \Sigma_{x_{1:T}} \end{bmatrix} \begin{bmatrix} P_{y_{1:T}|x_{1:T}} \\ P_{y_{1:T}} \end{bmatrix}.
\]

**A. Rao–Blackwellized Statistical Linear Regression**

As discussed in the previous section, statistical linear regression can be used to approximate the nonlinear state transition and measurement functions [34]. In case these functions exhibit a conditionally affine substructure, the resulting integrals to be solved can be reduced in dimensionality since the conditionally affine subspace is analytically tractable. This is reviewed in Lemma 1, followed by the special case when SLR is applied with respect to the prior density in Corollary 1. The latter is the case for traditional Gaussian filtering and smoothing [3], [13].

**Lemma 1** (Rao–Blackwellized Statistical Linear Regression). Let \( z_2 = h(z_1^n) + H(z_1^n)z_1^l + v \).

\[
p(z_1^n, z_2) = N \left( \begin{bmatrix} z_1^n \\ z_2 \\ \mu_1^n \\ \mu_1^l \end{bmatrix}, \begin{bmatrix} \Sigma_1^n \\ \Sigma_1^l \\ (S_1^T)^{-1} \Sigma_1^l \end{bmatrix} \right),
\]
and \( p(v \mid z^n_1) = \mathcal{N}(v; 0, \Sigma_v(z^n_1)) \). Furthermore, let us fix the linearization density to
\[
\pi(z^n_1, z^n_2) = \mathcal{N}(z^n_2; \mu_{1,n}^0, \Sigma_{1,n}^0) \quad \text{and} \quad \rho(v \mid z^n_1) = \mathcal{N}(v; \mu_{1,n}^0, \Sigma_{1,n}^0).
\]

Given these, let
\[
\mu_{1,n}^1 = \mu_{1,n}^0 + \left( \Sigma_{1,n}^{-1} \Sigma_{1,2}^0 \right) \left( z^n_1 - \mu_{1,n}^0 \right), \\
\Sigma_{1,n}^1 = \Sigma_{1,n}^{-1} - \left( \Sigma_{1,2}^0 \right)^T \Sigma_{1,n}^{-1} \Sigma_{1,2}^0,
\]
and
\[
\mu_2 = \int \left[ (h + H\mu_1^T)(z^n_1) \right] \pi(z^n_1)dz^n_1, \\
\Sigma_{12} = \int \left[ (z^n_1 - \mu_{1,n}^0)(h + H\mu_1^T) \right] \pi(z^n_1)dz^n_1.
\]

Then, the statistical linear regression of \( z_2 \) with respect to \( \pi(z^n_1, z^n_2) \) is given by \( z_2 \approx \Phi z + \Gamma + \nu \), where
\[
\Phi = \Sigma_{12}, \\
\Gamma = \mu_{1,n} - \Phi \mu_{1,n}^0, \\
\nu = \Sigma_{2,n} - \Phi \Sigma_{1,2,n} \Phi^T.
\]

Proof. The matrix \( \Phi \) and vector \( \Gamma \) are chosen such that they minimize the error \( \epsilon = E(h(z^n_1) + H(z^n_1)z^n_2 + \nu - \Phi z - \Gamma) \), leading to the mean squared sense with respect to (10). This yields (13a)-(13b) (see [34] for details) where
\[
\mu_{2,n} = E_x \{ h(z^n_1) + H(z^n_1)z^n_2 + \nu \}, \\
\Sigma_{2,n} = Cov_x \{ h(z^n_1) + H(z^n_1)z^n_2 + \nu \}, \\
\Sigma_{12,n} = Cov \{ z_1, h(z^n_1) + H(z^n_1)z^n_2 + \nu \}.
\]

Furthermore, \( \nu \sim N(0, \Sigma_{2,n}) \) with
\[
\Sigma_{2,n} = Cov_x \{ h(z^n_1) + H(z^n_1)z^n_2 + \nu \} = \Phi Cov_x \{ z_1 \} \Phi^T.
\]

In order to calculate the resulting expectations in (14a)-(15), the linear substructure can now be exploited which yields the integrals
\[
\mu_{2,n} = E_x \{ h(z^n_1) + H(z^n_1)z^n_2 + \nu \} = \int (h(z^n_1) + H(z^n_1)\mu_1^0) \pi(z^n_1)dz^n_1,
\]
and similar for \( \Sigma_{2,n} \), \( \Sigma_{12,n} \), and \( \Sigma_{2,n} \), which leads to (12).

Having calculated the linear approximation (12)-(13), the Gaussian approximation of the joint density \( p(z_1, z_2) \) follows directly to be
\[
p(z_1, z_2) \approx \mathcal{N} \left( \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}; \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12} & \Sigma_{22} \end{bmatrix} \right)
\]

with
\[
\mu_2 = \Phi \mu_1 + \Gamma, \\
\Sigma_2 = \Phi \Sigma_1 \Phi^T + \Sigma_{2,n}, \\
\Sigma_{12} = \Sigma_1 \Phi^T.
\]

Corollary 1. If SLR is done with respect to the prior \( p(z_1) \) (i.e. \( \pi(z_1) = p(z_1) \)), then \( \Sigma_{1,2} = \Sigma_1 \) and \( \mu_{1,n} = \mu_{1} \). Thus (17) simplifies to \( \mu_2 = \mu_{2,n} \), \( \Sigma_2 = \Sigma_{2,n} \), and \( \Sigma_{12} = \Sigma_{1,2,n} \).

The expectations with respect to \( \pi(z^n_1) \) in (12) are in general not analytically tractable. Instead, numerical integration schemes such as sigma-point methods can be used. Here, a set \\{\( z_{1,m}^n \), \( w_{1,m}^n \)\}_{m=1}^M of M sigma-points \( z_{1,m}^n \) and their weights \( w_{1,m}^n \) are calculated analytically based on the moments of \( z_1^n \) [3], [13]. Then, an arbitrary expectation with respect to the density \( \pi(z^n_1) \) can be approximated as
\[
E_x \{ h(z^n_1) \} = \int h(z^n_1)\pi(z^n_1)dz^n_1 \approx \sum_{m=1}^M w_{1,m}^n h(z_{1,m}^n).
\]

Choosing the sigma-points can be done according to different sigma-point rules such as the unscented transform, Gauss–Hermite quadratures, or spherical cubatures [3], [9].

B. Model 1 Prediction and Smoothing

For Model 1, given the density \( \pi(x_{t-1}) = \mathcal{N}(x_{t-1}; \bar{x}_t, \Sigma_t) \), Lemma 1 is applied directly by choosing \( z^n_{1} = x^n_{t-1}, z^n_{2} = x^n_{t} \) and the covariance matrices accordingly. This yields
\[
\bar{x}_{t|t-1} = E_x \{ f + Ax_{t-1} \}, \\
\bar{P}_{t|t-1} = E_x \{ (f + Ax_{t-1} - \bar{x}_{t|t-1})(f + Ax_{t-1} - \bar{x}_{t|t-1})^T \} + E_x \{ AP_t A^T + Q \},
\]

Choosing the sigma-points can be done according to different sigma-point rules such as the unscented transform, Gauss–Hermite quadratures, or spherical cubatures [3], [9].

1) Regular Smoothing: For regular smoothing SLR is performed with respect to \( \pi(x^n_{t-1}) = p(x^n_{t-1} \mid y_{1:t}) \). Hence, \( \bar{x}_t = \bar{x}_{t-1 \mid t-1} \) and \( P_t = P_{t-1 \mid t-1} \) in (19) and Corollary 1 applies. Thus
\[
\bar{x}_{t|t-1} = \bar{x}_{t-1}, \quad \bar{P}_{t|t-1} = \bar{P}_{t-1}, \quad \text{and} \quad C_{t-1} = C_{t-1}.
\]

2) Posterior Linearization Smoothing: For posterior linearization smoothing, \( \pi(x^n_{t-1}) = p(x^n_{t-1} \mid y_{1:t}) \). Hence, \( \bar{x}_t = \bar{x}_{t-1 \mid t} \) and \( P_t = P_{t-1 \mid t} \), while the predictive moments become
\[
\bar{x}_{t|t-1} = \bar{x}_{t-1 \mid t} + \bar{x}_t, \quad \bar{P}_{t|t-1} = \bar{P}_{t-1 \mid t} + \Sigma_{12}, \quad \text{and} \quad C_{t-1} = C_{t-1}. \]

This leads to the prediction step summarized in Algorithm 1 where \( SP(\mu, \Sigma) \) denotes the generation of sigma-points with respect to the mean \( \mu \) and covariance \( \Sigma \).

C. Model 2 Prediction and Smoothing

For the second model, first note that the prediction for \( x^n_{t} \) is purely nonlinear and independent of \( x^n_{t-1} \). Thus, \( \bar{x}_{t|t-1} \), \( \bar{P}_{t|t-1} \), and \( C_{t-1} \) are found through SLR with respect to \( \pi(x^n_{t-1}) \) and are given by
\[
\bar{x}_{t|t-1} = E_x \{ f^n \}, \\
\bar{P}_{t|t-1} = E_x \{ (f^n - \bar{x}_{t|t-1})(f^n - \bar{x}_{t|t-1})^T + Q^n \},
\]

This yields the following approximation
\[
\Phi_{t-1} = \rho_{t-1,0} \bar{P}_{t-1}^{-1}, \\
\Gamma_{t-1} = \bar{x}_{t|t-1} - \Phi_{t-1} \bar{x}_{t|t-1}.
\]
Algorithm 1 Prediction Step for Model 1

1: Let \( \{x_{t-1,m}^n, z_{t-1,m}^n\}_{m=1}^M \leftarrow \text{SP}(\hat{x}_x, P_n) \)
2: Calculate \( \bar{x}_{t-1}^n \) and \( P_{t-1}^n \) according to (11) using \( \hat{x}_x \) and \( P_n \)
3: Calculate \( x_{t-1}^n = \left[ f^n(x_{t-1,m}^n) + A^n(z_{t-1,m}^n) \right] z_{t-1,m}^n \)
4: Calculate \( \bar{x}_{t-1} \), \( \bar{P}_{t-1} \), and \( C_{t-1} \):
   - Regular prediction: (21)
   - Posterior linearization prediction: (20) and (22)

\[ \Sigma_{\nu_x} = P_{t-1}^n - \Phi^*_{t-1} P_n (\Phi^*_{t-1})^T, \]

where \( \bar{x}_x \) and \( P_n \) denote the mean and covariance used in (23).

The prediction for \( x_{t-1}^n \) is found from Lemma 1 with \( z_{t-1}^n = x_{t-1}^n \), \( z_2 = x_2 \). This gives

\[ \bar{x}_{t-1} = E \{ f^l + A^l \bar{x}_x \} \]

\[ \bar{P}_{t-1} = E \{ (f^l + A^l \bar{x}_x - \bar{x}_{t-1}) (f^l + A^l \bar{x}_x - \bar{x}_{t-1})^T \}
+ E \{ A^l \bar{P}_{t-1}^l (A^l)^T + Q^l \} \]

The coefficients of the linear approximation are then

\[ \Phi^l_{t-1} = \left[ P_{t-1}^l (C_{t-1}^l)^{-1} \right]^T \]

\[ \Gamma_{t-1} = \bar{x}_{t-1} - \Phi^l_{t-1} \bar{x}_x \]

\[ \Sigma_{\nu_x} = P_{t-1}^n - \Phi^l_{t-1} P_n (\Phi^l_{t-1})^T \]

and \( \bar{x}_x \) and \( P_n \) are the moments used for SLR with respect to the dynamics of the linear model.

The two moments \( C_{t-1}^n \) and \( C_{t-1}^m \) do not arise directly from (23)–(26). However, given the approximation \( x_{t-1}^n \approx \Phi^l_{t-1} x_{t-1} + \Gamma^l_{t-1} + \nu_{t-1} \), \( C_{t-1}^n \) is found as follows

\[ C_{t-1}^n = E \{ (x_{t-1} - \bar{x}_{t-1}) (x_{t-1} - \bar{x}_{t-1})^T \} (\Phi^l_{t-1})^{-1} \]

Similarly we have the approximation \( x_{t-1} \approx \Phi^l_{t-1} x_{t-1} + \Gamma^l_{t-1} + \nu_{t-1} \)
with \( x_{t-1} = \left[ (\bar{x}_x)^T \right]^T \). Thus, \( C_{t-1}^m \) becomes

\[ C_{t-1}^m = E \{ (x_{t-1} - \bar{x}_{t-1}) (x_{t-1} - \bar{x}_{t-1})^T \} (\Phi^l_{t-1})^{-1} \]

The resulting integrals are with respect to either \( \pi(x_{t-1}^n) \) or \( \pi(x_{t-1}) \)
(c.f. (23)–(25)) and thus, they need to be calculated in two steps. In each step, a new set of sigma-points with respect to each of these densities has to be calculated. Fortunately, the sigma-points with respect to the density \( \pi(x_{t-1}^n) \) will be re-used in the measurement update and thus, in practice, no additional sigma-points are needed.

1) Regular Smoothing: For predicting the nonlinear states \( x_{t-1}^n \), the expectations in (23) are evaluated with respect to \( \pi(x_{t-1}^n) = p(x_{t-1}^n | y_{t-1}) \). Thus, \( \bar{x}_{t-1} = \bar{x}_{t-1}^n - \bar{x}_{t-1} \) and \( \bar{P}_{t-1} = P_{t-1}^n \)
Hence, Corollary 1 applies, and

\[ \bar{x}_{t-1} = \bar{x}_{t-1}^n - \bar{x}_{t-1} \]

\[ \bar{P}_{t-1} = P_{t-1}^n \]

In the second step, the integrals are calculated with respect to \( \pi(x_{t-1}^n, x_{t-1}^m) = p(x_{t-1}^n | y_{t-1}) \). Thus, \( \bar{x}_{t} = \bar{x}_{t-1}^n - \bar{x}_{t-1} \) and \( P_n = P_{t-1}^n \) from (30). Again, Corollary 1 applies and thus,

\[ \bar{x}_{t} = \left[ \bar{x}_{t-1}^n \right] \]

\[ P_n = \left[ C_{t-1}^n \right] \]

and the de-correlation is as in (11) with \( \mu_1 = \bar{x}_{t} \) and \( \Sigma_{\nu_x} = P_n \) from (30).

2) Posterior Linearization Smoothing: In posterior linearization smoothing, predicting the nonlinear state is with respect to \( \pi(x_{t-1}^n) = p(x_{t-1}^n | y_{t-1}) \). Thus, \( \bar{x}_{t-1} = \bar{x}_{t-1}^n - \bar{x}_{t-1} \) and \( P_{t-1} = P_{t-1}^n \)

\[ \bar{x}_{t-1}^n = \Phi^l_{t-1} \bar{x}_{t-1}^n + \Gamma^l_{t-1} + \nu_{t-1} \]

\[ P_{t-1} = \left[ C_{t-1}^n \right] \]

\[ \left[ \begin{array}{c} p_{t-1} \ 
\end{array} \right] \]

The predicted moments thus become

\[ \bar{x}_{t} = \Phi^l_{t-1} \bar{x}_{t-1}^n + \Gamma^l_{t-1} + \nu_{t-1} \]

\[ P_{t-1} = \left[ \begin{array}{c} C_{t-1}^n \ 
\end{array} \right] \]

This yields the prediction step for Model 2 given in Algorithm 2.

D. Measurement Update

Since the structure of the measurement model is the same for both models (c.f. Eq. (3c) vs. (4c)), \( \tilde{y}_{t-1} = D_t \), and \( \tilde{S}_t \) follow directly by applying Lemma 1 with \( z_1 = \tilde{y}_t \), \( z_2 = y_t \). This gives

\[ \tilde{y}_{t-1} = E_x (g + B \tilde{z}_t) \]

\[ D_t = E_x (\tilde{y}_t - \tilde{y}_t)(g + B \tilde{z}_t - \tilde{y}_t)^T \]

\[ \tilde{S}_t = E_x (g + B \tilde{z}_t - \tilde{y}_t)(g + B \tilde{z}_t - \tilde{y}_t)^T + E_x (B \tilde{P}_{t-1} B^T + R) \]

Algorithm 2 Prediction Step for Model 2

1: Let \( \{x_{t-1}^{n,m}, w_{t-1}^{m}\}_{m=1}^{M} \leftarrow \text{SP}(\tilde{x}_\pi, \tilde{P}_\pi) \)
2: Calculate \( \tilde{x}_{t|t-1}^{n,m} = f_t(x_{t-1}^{n,m}) \)
3: \( \tilde{x}_{t|t-1} = \sum_{m=1}^{M} w_{t-1}^{m} \tilde{x}_{t|t-1}^{n,m} \)
4: \( \tilde{P}_{t|t-1}^{n} = \sum_{m=1}^{M} w_{t-1}^{m} \left[ (x_{t|t-1}^{n,m} - \tilde{x}_{t|t-1}^{n,m}) (x_{t|t-1}^{n,m} - \tilde{x}_{t|t-1}^{n,m})^T \right] \)
5: \( \tilde{P}_{t|t-1} = \sum_{m=1}^{M} w_{t-1}^{m} \left[ (x_{t|t-1}^{n,m} - \tilde{x}_{t|t-1}^{n,m}) (x_{t|t-1}^{n,m} - \tilde{x}_{t|t-1}^{n,m})^T \right] + Q_t(x_{t-1}^{n,m}) \)
6: \( \tilde{C}_{t|t-1}^{n} = \sum_{m=1}^{M} w_{t-1}^{m} \left( (x_{t|t-1}^{n,m} - \tilde{x}_{t|t-1}^{n,m}) (x_{t|t-1}^{n,m} - \tilde{x}_{t|t-1}^{n,m})^T \right) \)

Algorithm 3 Measurement Update

1: Let \( \{x_{t}^{n,m}, w_{t}^{m}\}_{m=1}^{M} \leftarrow \text{SP}(\tilde{x}_\pi, \tilde{P}_\pi) \)
2: Calculate \( \tilde{x}_{t}^{1,m} \) and \( \tilde{P}_{t}^{1} \) according to (11) using \( \tilde{x}_\pi \) and \( P_\pi \)
3: Calculate \( \tilde{y}_{t|t-1} = g(x_{t}^{n,m}) + B(x_{t}^{n,m}) \tilde{x}_{t}^{1,m} \)
4: \( \tilde{y}_{t|t-1} = \sum_{m=1}^{M} w_{t}^{m} \tilde{y}_{t|t-1}^{m} \)
5: \( \tilde{D}_t = (\tilde{P}_{t}^{1})^T (\tilde{P}_{\pi}^{1})^{-1} \tilde{D}_t^{1} + \tilde{P}_{t}^1 \sum_{m=1}^{M} w_{t}^{m} B(x_{t}^{n,m})^T \)
6: \( \tilde{S}_t = \sum_{m=1}^{M} w_{t}^{m} \left( (\tilde{y}_{t|t-1}^{m} - \tilde{y}_{t|t-1}) (\tilde{y}_{t|t-1}^{m} - \tilde{y}_{t|t-1})^T \right) + B(x_{t}^{n,m}) \tilde{P}_{t}^1 B(x_{t}^{n,m})^T + R(x_{t}^{n,m}) \)

Algorithm 4 Rao–Blackwellized Gaussian Smoother

1: for \( t = 1, \ldots, T \) do
2: Calculate and store \( \tilde{x}_{t|t-1}, \tilde{P}_{t|t-1}, \) and \( \tilde{C}_{t|t-1} \)
3: Calculate and store \( \tilde{x}_{t}, P_{t|t}, \) and \( D_{t} \) (Algorithm 3)
4: end for
5: for \( t = T - 1, \ldots, 1 \) do
6: Calculate \( \tilde{x}_{t|T}, \tilde{P}_{t|T}, \) and \( E_{t} \) according to (6)
7: end for

E. Filtering, Smoothing, and Posterior Linearization Smoothing

Having developed the prediction and update steps for both models, the complete algorithm can now be formalized. First, filtering is achieved by simply alternating between prediction and measurement update at each time step over the whole dataset. After filtering, smoothing is performed by a backward sweep over the filtered data, implementing the Rauch–Tung–Striebel smoothing equations. This yields the Rao–Blackwellized Gaussian smoother in Algorithm 4.

Finally, posterior linearization smoothing is achieved by first running Algorithm 4 to obtain \( p^0(x_t | y_{1:T}) \), followed by re-linearization and iteratively improving the posterior approximation (Algorithm 5).

F. Computational Complexity

One of the main reasons for considering Rao–Blackwellization in the context of Gaussian filtering and smoothing is the reduction of the computational complexity when the dimension of the conditionally linear subspace is large. Since the proposed methods are based on Lemma 1, we start by analyzing the computational complexity of regular SLR and Rao–Blackwellized SLR with respect to the linear subspace dimension \( N_l^{1} \), which is provided in Lemma 2 below.

Lemma 2 (Computational Complexity of Rao–Blackwellized Statistical Linear Regression). Let \( z_2 = h(z_1^T) + H(z_1^T)z_1^T + v \) be a conditionally affine transformation of the Gaussian random variable \( z_1 = (z_1^T, z_1^T)^T \), where \( z_1^T \in \mathbb{R}^{N_l^1}, z_1^T \in \mathbb{R}^{N_l^1}, z_2 \in \mathbb{R}^N, \) and \( N_l = N_l^1 + N_l^2 \). Then the asymptotic (in \( N_l^1 \) ) computational
Algorithm 5 Rao–Blackwellized Posterior Linearization Smoother

1: Run Algorithm 4 to obtain $p^0(x_t | y_{1:t})$ and set $i \leftarrow 0$
2: do
3:   for $t = 1, \ldots, T$ do
4:      Calculate and store $z_{i|t-1}^{i+1}$, $P_{i|t-1}^{i+1}$, and $C_{i|t-1}^{i+1}$.
5:      a) Model 1: Algorithm 1
6:         b) Model 2: Algorithm 2
7:   end for
8:   for $t = T-1, \ldots, 1$ do
9:      Calculate and store $z_{i|t-1}^{i+1}$, $P_{i|t-1}^{i+1}$, and $D_{i|t-1}^{1}$ (Algorithm 3)
10: end for
11: while Not converged

Table I: Comparison of the asymptotic computational complexity for SLR and Rao–Blackwellized SLR (RB-SLR).

<table>
<thead>
<tr>
<th>Calculation</th>
<th>SLR $O(\cdot)$</th>
<th>RB-SLR $O(\cdot)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sigma-points</td>
<td>$N_1^3$</td>
<td>$(N_1^3)^3$</td>
</tr>
<tr>
<td>Orthogonalization</td>
<td>$n/a$</td>
<td>$M^2 N_1^2 N_1^3 + (N_1^2)^2 N_1^3$</td>
</tr>
<tr>
<td>$z_2 M, \Sigma_0$</td>
<td>$M(C_0 + C_H + C_0)$</td>
<td>$M(C_0 + C_H + C_0)$</td>
</tr>
<tr>
<td>$P_{2,\Sigma}$</td>
<td>$MN_2 N_1^3$</td>
<td>$+M N_2 N_1^3$</td>
</tr>
<tr>
<td>$\Sigma_{12,\Sigma}$</td>
<td>$MN_2^2 N_2$</td>
<td>$+M(N_1^2)^2 N_2 + M N_1^2 N_2^2$</td>
</tr>
<tr>
<td>$\Phi$</td>
<td>$N_2^3$</td>
<td>$N_2^3$</td>
</tr>
<tr>
<td>$\Gamma$</td>
<td>$N_2^3 N_2 + N_1 N_2^2$</td>
<td>$+N_1 N_2^2$</td>
</tr>
<tr>
<td>$\Sigma_2$</td>
<td>$N_2^3 N_2 + N_1 N_2^2$</td>
<td>$N_2^3 N_2 + N_1 N_2^2$</td>
</tr>
<tr>
<td>$\Sigma_{2,\Sigma}$</td>
<td>$N_2^3 N_2 + N_1 N_2^2$</td>
<td>$N_2^3 N_2 + N_1 N_2^2$</td>
</tr>
</tbody>
</table>

Implementing (13) and (17) requires the same computations, irrespective of whether Rao–Blackwellization is used or not. Here, the most complex operations are calculating the inverse (or Cholesky factorization) of an $N_1 \times N_1$ matrix when calculating $\Phi (N_1^2$ operations) and calculating $\Sigma_2 \Phi^T$ when calculating $\Sigma_2 (N_1^2 N_2 + N_1 N_2^2)$. This is in addition to the cost for calculating the sums. Thus, the overall computational complexity becomes $O(M N_1^2 N_2 + (N_1^2)^3)$ for regular SLR and $O((N_1^2)^2 N_2 + (N_1^2)^3)$ for Rao–Blackwellized SLR.

Given the asymptotic computational complexity for regular and Rao–Blackwellized SLR in Lemma 2, Corollary 2 is readily found.

Corollary 2. The asymptotic computational complexity for sigma-point-based SLR where the number of sigma-points is an affine function $M = \alpha N_1^2 + \beta$ is $O((N_1^2)^2 N_2 + (N_1^2)^3)$ for both regular and Rao–Blackwellized SLR. Then, the ratio between nonlinear and linear states as well as the cost for evaluating the functions $h(\cdot)$, $H(\cdot)$, and $\Sigma_{\omega}(\cdot)$ determine which method is faster. Conversely, if the number of sigma-points increases faster than linear, the asymptotic complexity of Rao–Blackwellized SLR is always lower.

It follows that for the third order unscented transform ($M = 2N_1 + 1$), the constants determine which method is faster, whereas for higher order methods, such as the fifth order unscented transform ($M = 2N_1^2 + 1$) or the Gauss–Hermite quadrature ($M = p^3$), Rao–Blackwellization is beneficial for systems with high-dimensional linear subspaces.

Finally, the overall complexity of the proposed methods is given in Lemma 3.

Lemma 3 (Computational Complexity of Rao–Blackwellized Gaussian Smoothing). For Algorithm 4 and Algorithm 5, the asymptotic computational complexity with respect to $N_2$ is $O((N_1^2)^3)$ and the computational complexity for the corresponding non-Rao–Blackwellized smoothers is $O(M(N_1^2)^3 + (N_1^2)^3)$.

Proof. First, note that the prediction for Model 1 is a direct application of Lemma 1 with $N_1 = N_2 = N_1$ with no additional operations. Hence, the total computational complexity of Algorithm 1 readily follows to be $O((N_1^2)^3)$. For Algorithm 2 (prediction for Model 2), the nonlinear state is predicted in a regular prediction step with $N_1 = N_2 = N_1^0$ which is independent of $N_1$ and thus has constant cost. For predicting the linear state, Lemma 1 is applied with $N_1 = N_2$ and $N_2 = N_1^0$ and thus, the complexity is again $O((N_1^2)^3)$. The measurement update (Algorithm 3) is again a direct application of Lemma 1 with $N_1 = N_2 = N_2$ with no further operations. Hence, SLR scales according to $O((N_1^2)^2)$ for regular filtering while it scales according to $O((N_2^2)^3)$ for posterior linearization filtering/smoothing (due to (13) and (17) which scale according to $O((N_1^2)^3)$). Furthermore, the measurement update (8) is done, which has the complexity $O(N_2^6)$ due to the covariance update. The backward pass for smoothing just adds the Rauch–Tung–Striebel smoothing calculations in (6). Here, the most expensive operations are calculating the gain $G_t$ and the covariance matrices $P_{t|T}$ and $E_t$ which all scale according to $O(N_2^6)$. Thus, it follows that the overall asymptotic computational complexity in $N_2$ for Rao–Blackwellized Gaussian smoothing for Model 1 as well as Model 2 is $O((N_2^2)^5)$.

Similarly, for non-Rao–Blackwellized smoothing, it follows from Lemma 2 that the prediction step for both Model 1 and Model 2 scale according to $O(M(N_1^2)^2 + (N_1^2)^3)$. Furthermore, the regular measurement update scales as $O(M(N_1^2)(N_1^2)^3)$, while the smoothing recursion is the same as for Rao–Blackwellized smoothing (i.e. $O(N_2^6)$). Thus, the overall computational complexity for non-Rao–Blackwellized smoothing is $O(M(N_1^2)^2 + (N_1^2)^3)$.
V. NUMERICAL ILLUSTRATIONS

A. Setup

In order to illustrate the proposed method, a system of harmonic oscillators with unknown, time-varying frequency is considered. This system allows for simple extension to higher order linear state-spaces by just considering higher order harmonics. The model is given by

\[\begin{align*}
x^n_t &= x^n_{t-1} + q^n_t, \\
x^I_t &= A^I(x^n_{t-1})x^I_{t-1} + q^I_t, \\
y_t &= Bx^I_{t-1} + r_t.
\end{align*}\]  

(39a)  

(39b)  

(39c)

The matrices \(A^I\) and \(B\) in (39) are

\[F(\omega) = \begin{bmatrix}
\cos(\omega T_s) & -\sin(\omega T_s) \\
\sin(\omega T_s) & \cos(\omega T_s)
\end{bmatrix},\]

\[A^I(x^n_t) = \text{blkdiag}(F(x^n_t), F(2x^n_t), \ldots, F(N^I_t/2x^n_t)),\]

\[B = \begin{bmatrix} 1 & 0 & 1 & 0 & \ldots & 1 & 0 \end{bmatrix},\]

and \(T_s = 0.05\,\text{s}\) is the sampling time.

The process and measurement noise covariances are chosen as \(Q = 0.01I_{N^I+1}\) and \(R = 0.1\), respectively. The number of harmonics varies from 1 to 5, which yields \(N^I = 2, 4, \ldots, 10\). For each case, \(T = 100\) samples are generated and 100 Monte Carlo runs are performed. As performance measures, the time-averaged root mean squared error (RMSE) is calculated over all Monte Carlo simulations. Additionally, the computational time per sample is measured. The algorithms are implemented in m-code in Matlab and the simulations are run on a 3.4 GHz 3rd generation Intel Xeon E3 CPU with 16 GB RAM. We compare the following five filtering methods:

- An unscented Kalman filter (UKF) [35],
- a Rao–Blackwellized unscented Kalman filter (RB-UKF),
- a Gauss–Hermite filter (GHF) of order \(p = 3\),
- a Rao–Blackwellized Gauss–Hermite filter (RB-GHF),
- a Rao–Blackwellized particle filter [16] with \(M_f = 250\) particles (RB-PF),

as well as the corresponding smoothers, that is,

- an unscented Rauch–Tung–Striebel smoother (URTSS) [6],
- a Rao–Blackwellized unscented Rauch–Tung–Striebel smoother (RB-URTSS),
- a Gauss–Hermite smoother (GHS),
- a Rao–Blackwellized Gauss–Hermite smoother (RB-GHS),
- a Rao–Blackwellized forward-filtering backward-simulation particle smoother [17] with \(M_f = 250\) filter and \(M_s = 100\) smoother particles (RB-FFBSi).

B. Results and Discussion

Fig. 1 (top) illustrates the RMSE of the compared methods as a function of \(N^I_t\). As it can be seen, all filters perform roughly equally well, with a slightly lower RMSE for the UKF and RB-PF for higher dimensions compared to the remaining filters. However, there is no difference between the GHF and the RB-GHF. Furthermore, Fig. 1 (bottom) depicts the time required for a complete filter update (time and measurement update). Here, the differences are obvious and striking. The computational load increases exponentially for the GHF and cubically for the RB-GHF, while it increases cubically for both the UKF as well as the RB-UKF but at a slower rate for the latter.

The results for smoothing are shown in Fig. 2. Similar to filtering, there is no significant difference in the RMSE for the different smoothers (Fig. 2, top) with the Gaussian smoothers performing slightly better than the particle smoother. The significant difference is again in the computational time (Fig. 2, bottom). For the Gaussian smoothers, the results from filtering carry over since only little overhead is added by the backward recursion (see Algorithm 4).

As shown in Section IV-F, the significant increase in computational time observed for the GHF and UKF are directly related to the number of sigma-points. For the Gauss–Hermite quadrature used in the GHF, the number of sigma-points scales as \(O(p^{2N^I})\) and for the third order unscented transform used here, it scales as \(O(2N^I+1)\). By using Rao–Blackwellization, the number of sigma-points only depends on the size of the nonlinear subspace \(N^I_t\), which is constant in this example. This yields the reduction of computational complexity from exponential to cubic for the RB-GHF and RB-GHS. For the unscented filter and smoother, the scaling remains cubic, but the Rao–Blackwellized version is still faster. Closer inspection showed that evaluating \(A^I\) is one of the most expensive operations in this case (evaluating the sine and cosine functions). Thus, since the Rao–Blackwellized smoother requires less function evaluations, it is faster in this case.

The results also indicate that the RMSE is not affected significantly by the Rao–Blackwellization for the compared methods. Furthermore,
note that while the RB-PF is slightly superior in terms of RMSE, the RB-FFBSi is not. This is due to a particular set of circumstances that make the RB-PF more effective in this case. The Gaussian smoothing is a key component of this choice, and the results suggest that this choice is effective in the context of the specific problem at hand.

The dimension of the nonlinear state might be different in the dynamic and observation models, as in the example considered in this section. In such cases, the algorithms may be further improved by extending the dimension of the respective nonlinear subspaces and exploring the resulting performance gains. In this way, the framework can be extended to a broader range of applications.

In this paper, Gaussian smoothers with analytically tractable conditional linear substructures were developed and their computational complexity was analyzed. The framework is general in the sense that it can be used together with any sigma-point based Gaussian filtering/smoothing algorithm such as the unscented Rauch–Tung–Striebel smoother or Gaussian–Hermite quadrature-based smoothers. The simulations showed that a significant performance gain in terms of computational efficiency can be achieved, even for low-dimensional linear subspaces. Hence, it is useful to use Rao–Blackwellization not only to reduce the computational burden but also to enable the usage of otherwise computationally prohibitive methods such as Gaussian–Hermite quadratures for higher dimensional systems.

VI. CONCLUSIONS

In this paper, Gaussian smoothers with analytically tractable conditional linear substructures were developed and their computational complexity was analyzed. The framework is general in the sense that it can be used together with any sigma-point based Gaussian filtering/smoothing algorithm such as the unscented Rauch–Tung–Striebel smoother or Gaussian–Hermite quadrature-based smoothers. The simulations showed that a significant performance gain in terms of computational efficiency can be achieved, even for low-dimensional linear subspaces. Hence, it is useful to use Rao–Blackwellization not only to reduce the computational burden but also to enable the usage of otherwise computationally prohibitive methods such as Gaussian–Hermite quadratures for higher dimensional systems.

REFERENCES


