ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9413364

PARALLEL ITERATED EXTENDED AND SIGMA-POINT KALMAN SMOOTHERS

Fatemeh Yaghoobi, Adrien Corenflos, Sakira Hassan, Simo Särkkä

Department of Electrical Engineering and Automation, Aalto University, Finland

ABSTRACT

The problem of Bayesian filtering and smoothing in nonlinear models with additive noise is an active area of research. Classical Taylor series as well as more recent sigma-point based methods are two well-known strategies to deal with this problem. However, these methods are inherently sequential and do not in their standard formulation allow for parallelization in the time domain. In this paper, we present a set of parallel formulas that replace the existing sequential ones in order to achieve lower time (span) complexity. Our experimental results done with a graphics processing unit (GPU) illustrate the efficiency of the proposed methods over their sequential counterparts.

Index Terms— parallel computing, nonlinear estimation, iterated extended Kalman smoother, sigma-point smoother

1. INTRODUCTION

In recent years, the rapid advancements in hardware technologies such as graphics processing units (GPUs) and tensor processing units (TPUs) allow compute-intensive workloads to be offloaded from the central processing units (CPUs) by introducing parallelism [1–3]. There is a wide variety of areas that can benefit from parallelization [4], one of which is state estimation.

State estimation is a common task that arises in various areas of science and engineering [5–7]. It aims at combining the noisy measurements and the model to estimate the hard-to-measure states. A frequent and classical method for solving this problem is based on Bayesian filtering and smoothing [5] which inherently provides a sequential solution with linear complexity in the number of time steps.

In order to tackle the computational burden of Kalman type of filters and smoothers, [8,9] provide sub-linear computational methods by taking advantage of the sparse structures of the matrices appearing in the batch forms of the problems. In other works, using an ensemble formulation of Kalman filter has been used to speed up the matrix computations through parallelization [10,11]. The primary focus of these works was the efficient computation of the covariance matrices either by introducing sparse or sample covariance matrices rather than considering the temporal state-space structure per se. While in the aforementioned works, parallelization of the subproblems in the area of Bayesian filtering and smoothing were considered, [12] presented a general parallelizable formulations specifically designed for parallelizing state-estimation problems in the temporal direction. Moreover, for the special case of linear Gaussian model, parallel equations for computing Kalman filter and Raugh–Tung–Striebel smoother solutions were derived.

Overcoming the computational burden in the case of nonlinear dynamical systems with additive Gaussian noise is also of paramount importance. In these types of models, various linearization approaches can be used. Taylor series expansion based iterated extended Kalman smoother (IEKS) methods [13–15] and sigma-point based methods [5] are well-established techniques in literature. Iterated sigma-point methods have been proposed, for example, in [16, 17]. Despite the capabilities of the aforementioned methods in state estimation in nonlinear Gaussian models, they lack a framework which enables the computations in a more efficient way when using parallelization.

The contribution of this paper is to present a set of parallelizable formulas for filtering and smoothing in nonlinear Gaussian systems, in particular, IEKS and sigma-point based methods using a scan algorithm [12, 18]. The proposed methods reduce the linear span complexity of the state estimation methods to logarithmic with respect to the number of measurements.

This paper is organized as follows: Section 2 briefly reviews the generic parallel framework for Bayesian filters and smoothers. Sections 3 and 4 are concerned with presenting the formulation of the problem and proposing our method. Section 5 analyzes the efficiency and the computational complexity of the proposed method through one numerical example, and Section 6 concludes the paper.

2. GENERAL PARALLEL FRAMEWORK FOR BAYESIAN FILTERS AND SMOOTHERS

It is shown in [12] that the computation of sequential Bayesian filtering and smoothing can be converted to general parallel formulas in terms of associative operations. This allows for the use of the parallel scan method [18] which is a common algorithm used to speed-up sequential computations, for example, on GPU-based computing systems. In the rest of this

The authors would like to thank Academy of Finland for funding.

section, we review the general parallel algorithms provided in [12] which we then extend to nonlinear Gaussian models.

Given a state space model of the following form:

$$x_k \sim p(x_k \mid x_{k-1}), \quad y_k \sim p(y_k \mid x_k), \tag{1}$$

the goal of the filtering problem is to find the posterior distributions $p(x_k \mid y_{1:k})$ for $k = 1, \ldots, n$. This distribution is a probabilistic representation of the available statistical information on the state $x_k \in \mathbb{R}^{n_x}$ given the measurements $y_{1:k} = \{y_1, \ldots, y_k\}$ with $y_k \in \mathbb{R}^{n_y}$. Having acquired the filtering results for $k = 1, \ldots, n$, and using all the n measurements, the Bayesian smoother can be used to compute the posterior distributions $p(x_k \mid y_{1:n})$. The following strategies are used in [12] so as to particularize a_k and the binary associative operator $\otimes$ which provide a parallel framework for solving the aforementioned sequential filtering and smoothing problem.

Filtering. Given two positive functions $g'_i(y), g'_j(y)$ and two conditional densities $f'_i(x \mid y), f'_j(x \mid y)$, the authors of [12] proved that the binary operation $(f'_i, g'_i) \otimes (f'_j, g'_j) = (f'_{ij}, g'_{ij})$ defined by

$$\begin{aligned} f'_{ij}(x|z) &= \frac{\int g'_i(y) f'_j(x \mid y) f'_i(y \mid z) dy}{\int g'_j(y) f'_i(y \mid z) dy}, \\ g'_{ij}(z) &= g'_i(z) \int g'_j(y) f'_i(y \mid z) dy, \end{aligned} \tag{2}$$

is associative and by selecting $a_k = (f'_k, g'_k)$ as follows:

$$\begin{aligned} f'_k(x_k \mid x_{k-1}) &= p(x_k \mid y_k, x_{k-1}), \\ g'_k(x_{k-1}) &= p(y_k \mid x_{k-1}), \end{aligned} \tag{3}$$

where $p(x_1 \mid y_1, x_0) = p(x_1 \mid y_1)$ and $p(y_1 \mid x_0) = p(y_1)$, the Bayesian map $\binom{p(x_k|y_{1:k})}{p(y_{1:k})}$ can be rewritten as the k-th prefix sum, $a_1 \otimes \cdots \otimes a_k$.

Smoothing. Similarly [12], for any conditional densities $f'_i(x \mid y)$ and $f'_j(x \mid y)$ the binary operation $f'_i \otimes f'_j := \int f'_i(x|y) f'_j(y|z) dy$ is associative and by selecting $a_k = p(x_k \mid y_{1:k}, x_{k+1})$ with $a_n = p(x_n \mid y_{1:n})$, the Bayesian smoothing solution can then be calculated as $p(x_k \mid y_{1:n}) = a_k \otimes a_{k+1} \otimes \cdots \otimes a_n$.

Having considered the aforementioned general formulations, in this paper, we aim to extend the element a_k and the binary associative operator $\otimes$ to linear approximations of non-linear Gaussian systems, specifically, to the extended Kalman filter and smoother, and sigma-points methods.

3. PROBLEM FORMULATION

We consider the following model:

$$\begin{aligned} x_k &= f_{k-1}(x_{k-1}) + q_{k-1}, \\ y_k &= h_k(x_k) + r_k, \end{aligned} \tag{4}$$

where $f_{k-1}(.)$ and $h_k(.)$ are nonlinear functions. The q_k and r_k are the process and measurement noises, which are assumed to be zero-mean, independent Gaussian noises with known covariance matrices, Q_k and R_k, respectively. Furthermore, the initial state is Gaussian $x_0 \sim N(m_0, P_0)$ with known mean m_0 and covariance P_0. This paper is concerned with the computing approximate posterior distributions of the states $x_{0:n} = \{x_0, x_1, \ldots, x_n\}$ given all the measurements $y_{1:n} = \{y_1, x_1, \ldots, y_n\}$ in parallel form, or more precisely, the corresponding filtering and smoothing distributions.

Since the filtering and smoothing problems are not solvable in closed-form in the general non-linear case, one needs to resort to approximations. Here we follow the Gaussian filtering and smoothing frameworks [5] and form linear approximations of the system (4) in the following form:

$$\begin{aligned} f_{k-1}(x_{k-1}) &\approx F_{k-1} x_{k-1} + c_{k-1} + e_{k-1}, \\ h_k(x_k) &\approx H_k x_k + d_k + v_k, \end{aligned} \tag{5}$$

where $F_k \in \mathbb{R}^{n_x \times n_x}$, $c_k \in \mathbb{R}^{n_x}$, $H_k \in \mathbb{R}^{n_y \times n_x}$, $d_k \in \mathbb{R}^{n_y}$, $e_k \in \mathbb{R}^{n_x}$ and $v_k \in \mathbb{R}^{n_y}$ are zero mean Gaussian noises with covariance matrices Λ_k and Ω_k, respectively.

There are different strategies to effectively select the parameters of (5). In this paper, we will consider two such strategies widely-used in the Gaussian filtering literature, namely iterated sigma-point and extended Kalman smoothers [14–16]. In these approaches, the linearized-filter-smoother method is repeated M times, with the linearization parameters leveraging the results of the previous smoothing pass instead of the previous step. We can therefore see our successive linear approximations as being parametrized by the following vectors and matrices:

$$F^{(i)}_{0:n-1}, c^{(i)}_{0:n-1}, \Lambda^{(i)}_{0:n-1}, H^{(i)}_{0:n-1}, d^{(i)}_{1:n}, \Omega^{(i)}_{1:n}. \tag{6}$$

In the rest of this section, we will discuss how to acquire the linearized parameters of (6) using these methods. Also, for the sake of notational simplicity, we drop the index i from these parameters.

Iterated sigma-point method. In this approach, we select the parameters $(F_{k-1}, c_{k-1}, \Lambda_{k-1})$ and (H_k, d_k, Ω_k) using sigma-point-based statistical linear regression (SLR) method [16] as follows. First, we select m sigma points $\mathcal{X}^{(i)}_{1,k}, \ldots, \mathcal{X}^{(i)}_{m,k}$ and their associated weights $w^{(i)}_{1,k}, \ldots, w^{(i)}_{m,k}$ according to the posterior moments $\bar{x}^{(i-1)}_k$ and $\bar{P}^{(i-1)}_k$ of the previous iteration, which are the best available estimates for the means and covariances of the smoothing distribution. Then, in order to find the parameters $(F_{k-1}, c_{k-1}, \Lambda_{k-1})$, transformed sigma-points are obtained as $\mathcal{Z}_j = f_{k-1}(\mathcal{X}^{(i)}_{j,k-1})$ for $j = 1, \ldots, m$, and the linearization parameters are then

given by:

$$\begin{aligned} F_{k-1} &= \Psi^\top \bar{P}_{k-1}^{-1}, \\ c_{k-1} &= \bar{z} - F_{k-1}\bar{x}_{k-1}, \\ \Lambda_{k-1} &= \Phi - F_{k-1}\bar{P}_{k-1}F_{k-1}^\top. \end{aligned} \tag{7}$$

If we now write $\bar{x} = \bar{x}_{k-1}$ and $w_j = w_{j,k-1}^{(i)}$, the required moment approximations for Equation (7) are [19]:

$$\begin{aligned} \bar{z} &\approx \sum_{j=1}^{m} w_j \mathcal{Z}_j, \\ \Psi &\approx \sum_{j=1}^{m} w_j (\mathcal{X}_j - \bar{x})(\mathcal{Z}_j - \bar{z})^\top, \\ \Phi &\approx \sum_{j=1}^{m} w_j (\mathcal{Z}_j - \bar{z})(\mathcal{Z}_j - \bar{z})^\top. \end{aligned} \tag{8}$$

Similarly, reusing Equations (8) with $\bar{x} = \bar{x}_k$, $w_j = w_{j,k}^{(i)}$, and $\mathcal{Z}_j = h_k(\mathcal{X}_{j,k}^{(i)})$ the parameters (H_k, d_k, Ω_k) can be calculated as follows:

$$\begin{aligned} H_k &= \Psi^\top \bar{P}_k^{-1}, \\ d_k &= \bar{z} - H_k\bar{x}_k, \\ \Omega_k &= \Phi - H_k\bar{P}_k H_k^\top. \end{aligned} \tag{9}$$

The iterated posterior linearization smoother (IPLS) [16] now consists in iterating Equations (7) and (9) with updated approximate means and covariances of the posterior distribution at each iteration.

Iterated extended Kalman smoother. In this case, Ω and Λ are selected as zeros, and (F_{k-1}, c_{k-1}) and (H_k, d_k) are obtained by analytical linearization at the previous posterior (smoother) mean estimate of $x_{0:N}$. This approach is recognized as Gauss–Newton method when computing the MAP estimates [14] and it can also be extended to correspond to Levenberg–Marquardt method [15]. Here, we aim to obtain the linearized parameters according to this method which will be used in the next section to get parallel formulas.

By expanding $f_{k-1}(x_{k-1})$ and $h_k(x_k)$ in the first-order Taylor series utilizing the previous posterior means $\bar{x}_k$, the parameters of (6) are:

$$\begin{aligned} F_{k-1} &= \nabla f(\bar{x}_{k-1}), \\ c_k &= f(\bar{x}_{k-1}) - F_{k-1}\bar{x}_{k-1}, \\ H_k &= \nabla h(\bar{x}_k), \\ d_k &= h(\bar{x}_k) - H_k\bar{x}_k, \end{aligned} \tag{10}$$

where ∇f and ∇h are the Jacobians of f and h, respectively. Please note that in this paper computation of parameters in (7) and (9), and (10) is performed offline, which means that we have all measurements as well as the results of previous trajectory, that is, $\bar{x}_{1:n}$ and $\bar{P}_{1:n}$ for all n data points.

Having obtained the linearized parameters, the remaining task is to find the parallel formulas which will be discussed in the next section.

4. THE PROPOSED METHOD

Probability densities for the model of form (4) with linearization parameters of form (6) can be formulated as follows:

$$\begin{aligned} p(x_k \mid x_{k-1}) &\approx N(x_k; F_{k-1}x_{k-1} + c_{k-1}, Q'_{k-1}), \\ p(y_k \mid x_k) &\approx N(y_k; H_k x_k + d_k, R'_k), \end{aligned} \tag{11}$$

where $Q'_{k-1} = Q_{k-1} + \Lambda_{k-1}$ and $R'_k = R_k + \Omega_k$. The goal here is to obtain the parallel nonlinear Gaussian filter and smoother for the model (11). To meet this goal, similar to the method used in [12], we define a_k and binary operator $\otimes$ for our new linearized model.

Nonlinear Gaussian filtering. Aiming to specify the element a_k for obtaining parallel filtering equations according to (3), we apply Kalman filter update step to the density $p(x_k \mid x_{k-1})$ with measurement y_k. The results of the matching terms are as follows:

$$\begin{aligned} f'_k(x_k \mid x_{k-1}) &= p(x_k \mid y_k, x_{k-1}) \\ &= N(x_k; A_k x_{k-1} + b_k, C_k), \end{aligned} \tag{12}$$

where:

$$\begin{aligned} A_k &= (I_{n_x} - K_k H_k)F_{k-1}, \\ b_k &= c_{k-1} + K_k(y_k - H_k c_{k-1} - d_{k-1}), \\ C_k &= (I_{n_x} - K_k H_k)Q'_{k-1}, \\ K_k &= Q'_{k-1}{H_k}^\top S_k^{-1}, \\ S_k &= H_k Q'_{k-1} H_k^\top + R'_k. \end{aligned} \tag{13}$$

It is worth noticing that in order to find parameters of (13) at $k = 1$ and given m_0 and P_0, conventional formulations of the Kalman filter method with the linearized parameters are applied directly for prediction and update steps.

Also, using the information form of Kalman filter [20], the distribution $g'_k(x_{k-1}) = p(y_k \mid x_{k-1}) \propto N_I(x_{k-1}; \eta_k, J_k)$ can be obtained as follows:

$$\begin{aligned} J_k &= (H_k F_{k-1})^\top S_k^{-1} H_k F_{k-1} \\ \eta_k &= (H_k F_{k-1})^\top S_k^{-1} H_k (y_k - H_k c_{k-1} - d_k). \end{aligned} \tag{14}$$

Equations (13) and (14) provide the parameters of element $a_k = (A_k, b_k, C_k, \eta_k, J_k)$ in the filtering step, and they can be computed in parallel. Also, given a_i and a_j with the mentioned parameters, the binary associative operator $a_i \otimes a_j =$

a_{ij} can then be calculated with the following parameterization [12, lemma 8]:

$$\begin{aligned} A_{ij} &= A_j(I_{n_x} + C_i J_j)^{-1} A_i, \\ b_{ij} &= A_j(I_{n_x} + C_i J_j)^{-1}(b_i + C_i \eta_j) + b_j, \\ C_{ij} &= A_j(I_{n_x} + C_i J_j)^{-1} C_i A_j^\top + C_j, \\ \eta_{ij} &= A_i^\top(I_{n_x} + J_j C_i)^{-1}(\eta_j - J_j b_i) + \eta_i, \\ J_{ij} &= A_i^\top(I_{n_x} + J_j C_i)^{-1} J_j A_i + J_i. \end{aligned} \tag{15}$$

The proof for Equations (15) can be found in [12].

Nonlinear Gaussian smoothing. Assume that the filtering means x_k^* and covariance matrices P_k^* for the model (11) have been acquired as described above. We now get the following parameters for the smoothing step:

$$p(x_k \mid y_{1:k}, x_{k+1}) = N(x_k; E_k x_{k+1} + g_k, L_k) \tag{16}$$

for $k < n$:

$$\begin{aligned} E_k &= P_k F_k^\top (F_k P_k^* F_k^\top + Q'_{k-1})^{-1}, \\ g_k &= x_k^* - E_k(F_k x_k^* + c_k), \\ L_k &= P_k^* - E_k F_k P_k^*, \end{aligned} \tag{17}$$

and for $k = n$:

$$\begin{aligned} E_n &= 0, \\ g_n &= x_n^*, \\ L_n &= P_n^*. \end{aligned} \tag{18}$$

In the smoothing step, the parameters $a_k = (E_k, g_k, L_k)$ can be calculated in parallel. Now, given two elements a_i and a_j, the binary associative operator defined by $a_i \otimes a_j = a_{ij}$ can be parametrized as follows [12, lemma 10]:

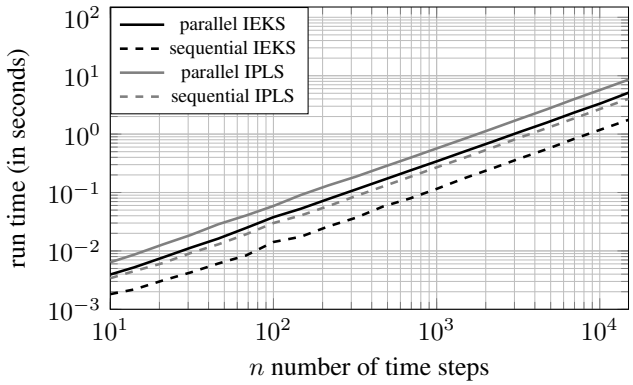
$$\begin{aligned} E_{ij} &= E_i E_j, \\ g_{ij} &= E_i g_j + g_i, \\ L_{ij} &= E_i L_j E_i^\top + L_i. \end{aligned} \tag{19}$$

5. EXPERIMENTAL RESULTS

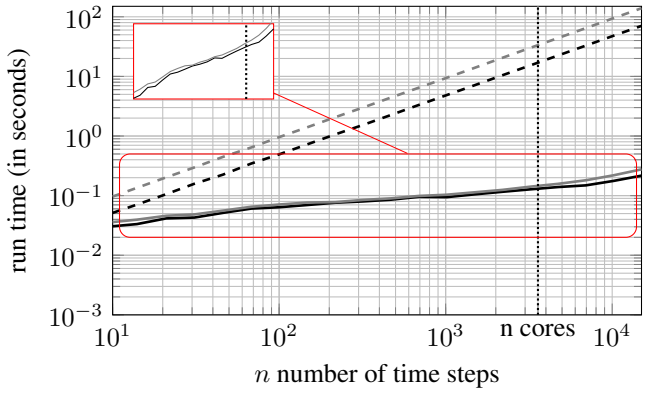
In this section, we evaluate the performance of the proposed methods on a simulated coordinated turn model with a bearings only measurement model [21] which was also used in [15]. To this end, we compare the effective average run time of the parallel versions of the extended (IEKS) and cubature integration [5] based sigma-point iterated smoothers (IPLS) with $M = 10$ iterations, as described in Section 4, with their sequential counterparts both on a CPU (Intel® Xeon® running at 2.30GHz) and on a GPU (Nvidia® Tesla® P100 PCIe 16 GB with 3584 cores). For our experiments we leverage the JAX framework [22] which implements the Blelloch parallel-scan algorithm [18] natively[1].

[1]The code can be found here: `https://github.com/EEA-sensors/parallel-non-linear-gaussian-smoothers`

In Figures 1a and 1b we observe that while the total computational cost of the parallel implementation of the iterated smoothers is higher than that of their sequential counterparts (Figure 1a), the parallelization properties of our proposed algorithms prove beneficial on a distributed environment such as a GPU (Figure 1b). Moreover, as outlined by the medallion in Figure 1b, our experiments indeed exhibit the theoretical logarithmic span complexity - derived in [12] for a linear Gaussian state space model - up to the parallelization capabilities of our GPU (3584 cores).



(a) CPU run time



(b) GPU run time

Fig. 1: Run time comparison of the parallel and sequential versions of the IEKS and IPLS on CPU (a) and GPU (b)

6. CONCLUSION

In this paper, parallel formulations for two kinds of nonlinear smoothers, namely, iterated sigma-point-based smoothers and iterated extended Kalman smoothers, have been presented. The proposed algorithms have the capability of diminishing the span-complexity from linear to logarithmic. Furthermore, the experimental results, which were conducted on a GPU, showed the benefits of the proposed methods over classical sequential methods.

7. REFERENCES

[1] T. Rauber and G. Rünger, *Parallel Programming: For multicore and cluster systems*, Springer, 2013.

[2] J. D. Owens, M. Houston, D. Luebke, S. Green, J. E. Stone, and J. C. Phillips, "GPU computing," *Proceedings of the IEEE*, vol. 96, no. 5, pp. 879–899, 2008.

[3] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, et al., "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, 2017, pp. 1–12.

[4] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and S. Clifford, *Introduction to Algorithms*, MIT Press, 2009.

[5] S. Särkkä, *Bayesian Filtering and Smoothing*, Cambridge University Press, 2013.

[6] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*, Wiley, 2001.

[7] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*, Academic Press, 1970.

[8] T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch continuous-time trajectory estimation as exactly sparse Gaussian process regression," in *Robotics: Science and Systems*, 2014, vol. 10.

[9] A. Grigorievskiy, N. Lawrence, and S. Särkkä, "Parallelizable sparse inverse formulation Gaussian processes (SpInGP)," in *2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2017, pp. 1–6.

[10] H. Ghorbanidehno, A. Kokkinaki, J. Lee, and E. Darve, "Recent developments in fast and scalable inverse modeling and data assimilation methods in hydrology," *Journal of Hydrology*, p. 125266, 2020.

[11] G. Evensen, "The ensemble Kalman filter: Theoretical formulation and practical implementation," *Ocean Dynamics*, vol. 53, no. 4, pp. 343–367, 2003.

[12] S. Särkkä and Á. F. García-Fernández, "Temporal parallelization of Bayesian smoothers," *IEEE Transactions on Automatic Control*, vol. 66, no. 1, pp. 299–306, 2021.

[13] B. M. Bell and F. W. Cathey, "The iterated Kalman filter update as a Gauss-Newton method," *IEEE Transactions on Automatic Control*, vol. 38, no. 2, pp. 294–297, 1993.

[14] B. M. Bell, "The iterated Kalman smoother as a Gauss–Newton method," *SIAM Journal on Optimization*, vol. 4, no. 3, pp. 626–636, 1994.

[15] S. Särkkä and L. Svensson, "Levenberg-Marquardt and line-search extended Kalman smoothers," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 5875–5879.

[16] Á. F. García-Fernández, L. Svensson, and S. Särkkä, "Iterated posterior linearization smoother," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 2056–2063, 2016.

[17] Á. F. García-Fernández, L. Svensson, M. R. Morelande, and S. Särkkä, "Posterior linearization filter: Principles and implementation using sigma points," *IEEE transactions on signal processing*, vol. 63, no. 20, pp. 5561–5573, 2015.

[18] G. E. Blelloch, "Scans as primitive parallel operations," *IEEE Transactions on Computers*, vol. 38, no. 11, pp. 1526–1538, 1989.

[19] I. Arasaratnam, S. Haykin, and R. J. Elliott, "Discrete-time nonlinear filtering algorithms using Gauss–Hermite quadrature," *Proceedings of the IEEE*, vol. 95, no. 5, pp. 953–977, 2007.

[20] B. D. O. Anderson and J. B. Moore, *Optimal Filtering*, Prentice-Hall, 1979.

[21] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*, vol. 19, YBS, 1995.

[22] J. Bradbury, R. Frostig, P Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, and S. Wanderman-Milne, "JAX: composable transformations of Python+NumPy programs," `http://github.com/google/jax`, 2018.