Parallel Kalman Smoothing Augmented Lagrangian Method for Constrained State Estimation

Qingwang Han, Rui Gao
dept. Naval Architecture and Ocean Engineering
Shanghai Jiao Tong University
Shanghai, China
{hanqingwang, rgao}@sjtu.edu.cn

Simo Särkkä
dept. Electrical Engineering and Automation
Aalto University
Espoo, Finland
simo.sarkka@aalto.fi

***Abstract*—This paper presents a novel parallel computing framework for solving constrained state estimation problems. The proposed methodology integrates an augmented Lagrangian formulation with a parallel Kalman filter and smoother implementation, specifically designed to optimize the primal update step through efficient parallel computation. Experimental results demonstrate that our approach provides a significant computational time improvement while maintaining the estimation accuracy.**

***Index Terms*—State estimation, parallel implementation, Kalman smoothing, augmented Lagrangian method.**

I. Introduction

State estimation plays a critical role in a wide range of applications, including sensor fusion, trajectory recovery, and signal processing [1]–[3]. It serves as a fundamental tool for reconstructing the states of dynamic systems from noisy incomplete sensor data [4]. However, a significant challenge in state estimation lies in managing the computational cost, particularly in large-scale systems where complexity and resource demands become important [5], [6]. To address this challenge, we propose a novel method that integrates the augmented Lagrangian method with parallel Kalman smoother.

Mathematically, the state estimation task is formulated as an inference problem within a statistical framework, where the evolution of the dynamic system's state is governed by constraints derived from inherent physical properties [7], [8]. Traditional Bayesian filtering and smoothing methods address this problem by integrating prior knowledge with measurement data to derive the posterior probability distribution of the system's state [9], [10]. For instance, in linear Gaussian models, the Kalman filter and Rauch–Tung–Striebel (RTS) smoother (also called Kalman smoother) provide efficient solutions to this estimation problem [11]–[13].

With the advancement of parallel computing, several parallelized filtering and smoothing methods have been developed to enhance computational efficiency [14]–[16]. For example, in [14], the Bayesian filtering and smoothing equations were reformulated as prefix-sum operations, enabling efficient computation through parallel scan algorithms [17]. The authors of [15] proposed a method that combines variational filtering with a parallel-iterative structure. However, these approaches often overlook the constraints inherent in the dynamic systems, limiting their applicability in many real-world scenarios.

Numerous optimization methods have been developed to address constrained optimization problems, including the augmented Lagrangian method [18], Peaceman–Rachford splitting [19], and the alternating direction method of multipliers (ADMM) [20], [21]. These approaches are particularly effective as they decompose the original problem into smaller subproblems by explicitly incorporating constraints into their frameworks [20], [22]. For instance, ADMM has been used for constrained state estimation by exploiting the Markovian structure of the model [23]. Similarly, the augmented Lagrangian method has been successfully applied to solve joint state estimation and parameter learning problems [24]. However, these methods often fail to leverage the computational power of GPUs, limiting their potential for large-scale applications.

The contribution of this paper is to present a novel parallel method for constrained state estimation, which integrates the augmented Lagrangian method with a parallel Kalman smoother. Specifically, for affine Gaussian systems, we develop a parallel Kalman smoother implementation of the primal update step that efficiently handles the update of the primal variable within the augmented Lagrangian method. This approach significantly reduces computational costs while maintaining the estimation accuracy. Experimental results demonstrate the effectiveness of the proposed method, highlighting its potential for practical applications in large-scale and computationally demanding scenarios.

II. Problem formulation

Let $\mathbf{x}_t \in \mathbb{R}^{N_x}$ be an unknown state vector of the dynamic system and $\mathbf{y}_t \in \mathbb{R}^{N_y}$ be a noisy measurement vector at time step t. Then, we have the following state-space model [1], [25]

$$\begin{aligned} \mathbf{x}_t &= \mathbf{A}_t\,\mathbf{x}_{t-1} + \mathbf{b}_t + \mathbf{q}_t, \\ \mathbf{y}_t &= \mathbf{H}_t\,\mathbf{x}_t + \mathbf{e}_t + \mathbf{r}_t,\ t = 1, \ldots, T, \end{aligned} \tag{1}$$

where $\mathbf{A}_t$ and $\mathbf{H}_t$ are the transition and measurement matrices, and $\mathbf{b}_t$ and $\mathbf{e}_t$ are the bias terms. The noises $\mathbf{q}_t$ and $\mathbf{r}_t$ are independent zero-mean time-white Gaussian noises with known covariances $\mathbf{Q}_t$ and $\mathbf{R}_t$. When $t = 0$, the initial state

$\mathbf{x}_0$ is assumed to be Gaussian with mean $\mathbf{m}_0$ and covariance $\mathbf{P}_0$. Also, we have the inequality constraint function

$$\mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t \leq \mathbf{0}, \tag{2}$$

where $\mathbf{C}_t$ is a matrix, and $\mathbf{d}_t$ is a vector. The objective here is to estimate the state sequence $\mathbf{x}_{1:T}$ from the measurements $\mathbf{y}_{1:T}$ under the constraints.

Given the affine Gaussian model with constraints, we can incorporate the constraint (2) into the posterior distribution using an indicator function $\mathbb{I}$:

$$\mathbb{I}(\mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t \leq \mathbf{0}) = \begin{cases} 1, & \text{if } \mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t \leq \mathbf{0}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Then, the posterior probability density can be written as

$$\begin{aligned} p(\mathbf{x}_{1:T} \mid \mathbf{y}_{1:T}) \propto \prod_{t=1}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) \prod_{t=1}^{T} p(\mathbf{y}_t \mid \mathbf{x}_t) \\ \prod_{t=1}^{T} \mathbb{I}(\mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t \leq \mathbf{0}), \end{aligned} \tag{4}$$

where $\propto$ denotes proportionality and

$$\begin{aligned} p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) &= \mathcal{N}(\mathbf{x}_t \mid \mathbf{A}_t\mathbf{x}_{t-1} + \mathbf{b}_t, \mathbf{Q}_t), \\ p(\mathbf{y}_t \mid \mathbf{x}_t) &= \mathcal{N}(\mathbf{y}_t \mid \mathbf{H}_t\mathbf{x}_t + \mathbf{e}_t, \mathbf{R}_t). \end{aligned} \tag{5}$$

Here, $\mathcal{N}(\mathbf{x} \mid \mathbf{m}, \mathbf{P})$ denotes a Gaussian probability density function with mean $\mathbf{m}$ and covariance $\mathbf{P}$ evaluated at $\mathbf{x}$. To make the problem tractable, we aim to compute the *maximum a posteriori* estimate of the state sequence $\mathbf{x}_{1:T}$. We aim to solve the following optimization problem:

$$\begin{aligned} \min_{\mathbf{x}_{1:T}} \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{y}_t - \mathbf{H}_t\mathbf{x}_t - \mathbf{e}_t\|^2_{\mathbf{R}_t^{-1}} \\ + \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{A}_t\mathbf{x}_{t-1} - \mathbf{b}_t\|^2_{\mathbf{Q}_t^{-1}}, \\ \text{s.t. } \mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t \leq \mathbf{0},\ t = 1, \ldots, T. \end{aligned} \tag{6}$$

In particular case where $\mathbf{C}_t = \mathbf{0}$ and $\mathbf{d}_t = \mathbf{0}$, the optimization problem (6) could be efficiently solved by using Kalman smoother [4], [11], [12]. However, when the Kalman smoother is no longer applicable (due to constraints) and the time step T is large, the cost function in (6) becomes computationally demanding. To address this issue, we propose a parallel implementation for constrained state estimation in the following section.

III. Proposed method

The proposed method integrates an augmented Lagrangian framework with a parallel Kalman smoother. The main idea involves decomposing the minimization of the complex objective function into an iterative sequence of computationally simpler subproblems via ADMM, similarly to [2], [23], [26]. The subproblem involving the optimization over the primal variable is then implemented using the parallel Kalman smoother [14].

A. Augmented Lagrangian Method

For solving (6), we start with introducing additional variable sequence $\mathbf{v}_{1:T}$ and defining the augmented Lagrangian function

$$\begin{aligned} \mathcal{L}(\mathbf{x}_{1:T}, \mathbf{v}_{1:T}; \boldsymbol{\zeta}_{1:T}) = \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{A}_t\mathbf{x}_{t-1} - \mathbf{b}_t\|^2_{\mathbf{Q}_t^{-1}} \\ + \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{y}_t - \mathbf{H}_t\mathbf{x}_t - \mathbf{e}_t\|^2_{\mathbf{R}_t^{-1}} + \sum_{t=1}^{T} \boldsymbol{\zeta}_t^{\top}(\mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t + \mathbf{v}_t) \\ + \frac{\rho}{2}\sum_{t=1}^{T} \|\mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t + \mathbf{v}_t\|^2. \end{aligned} \tag{7}$$

Here, $\boldsymbol{\zeta}_t$ is a Lagrangian multiplier, ρ is a parameter, and $\|\mathbf{x}\|_{\mathbf{R}} = \sqrt{\mathbf{x}^{\top}\mathbf{R}\mathbf{x}}$ denotes the $\mathbf{R}$-weighted Euclidean norm of a vector $\mathbf{x}$. The augmented Lagrangian method minimizes the function $\mathcal{L}(\mathbf{x}_{1:T}, \mathbf{v}_{1:T}; \boldsymbol{\zeta}_{1:T})$ by alternating the update steps of the variables $\mathbf{x}_t$, $\mathbf{v}_t$, and $\boldsymbol{\zeta}_t$.

Given the initial value $(\mathbf{x}_{1:T}^{(1)}, \mathbf{v}_{1:T}^{(1)}, \boldsymbol{\eta}_{1:T}^{(1)})$, the iteration associating with (7) has the following steps:

$$\begin{aligned} \mathbf{x}_{1:T}^{(l+1)} = \arg\min_{\mathbf{x}_{1:T}} \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{x}_t - \mathbf{A}_t\mathbf{x}_{t-1} - \mathbf{b}_t\|^2_{\mathbf{Q}_t^{-1}} \\ + \frac{1}{2}\sum_{t=1}^{T} \|\mathbf{y}_t - \mathbf{H}_t\mathbf{x}_t - \mathbf{e}_t\|^2_{\mathbf{R}_t^{-1}} \\ + \frac{\rho}{2}\sum_{t=1}^{T} \left\|\mathbf{C}_t\mathbf{x}_t + \mathbf{d}_t + \boldsymbol{\zeta}_t^{(l)}/\rho + \mathbf{v}_t^{(l)}\right\|^2, \end{aligned} \tag{8a}$$

$$\mathbf{v}_t^{(l+1)} = \max\big(\mathbf{0},\ -\mathbf{C}_t\mathbf{x}_t - \mathbf{d}_t - \boldsymbol{\zeta}_t^{(l)}/\rho\big), \tag{8b}$$

$$\boldsymbol{\zeta}_t^{(l+1)} = \boldsymbol{\zeta}_t^{(l)} + \rho(\mathbf{C}\mathbf{x}_t^{(l+1)} + \mathbf{d}_t + \mathbf{v}_t^{(l+1)}), \tag{8c}$$

where we solve the $\mathbf{v}_t$ and $\boldsymbol{\eta}_t$ subproblems for each t in parallel. Updating $\mathbf{x}_{1:T}$ via the subproblem in (8a) involves minimization of a quadratic optimization problem, which can be computed by the closed form solution given by the Kalman smoother, see [23], [26], for details. Since the main computational demand is in updating the primal variables $\mathbf{x}_{1:T}$, we use the parallel Kalman smoother to speed up this update step.

B. Parallel Implementation of Kalman Smoother

We first combine matrices $\mathbf{H}_t$ and $\mathbf{C}_t$ to an artificial measurement matrix $\overline{\mathbf{H}}_t$, combine $\mathbf{e}_t$ and $\mathbf{d}_t$ to a bias term $\overline{\mathbf{e}}_t$, combine $\mathbf{y}_t$ and $(-\boldsymbol{\zeta}_t/\rho - \mathbf{v}_t)$ to an artificial measurement $\overline{\mathbf{y}}_t$, and combine $\mathbf{R}_t$ and $\mathbf{I}/\rho$ to a covariance $\overline{\mathbf{R}}_t$. Then we can write

$$\begin{aligned} \overline{\mathbf{H}}_t = \begin{bmatrix} \mathbf{H}_t \\ \mathbf{C}_t \end{bmatrix}, \overline{\mathbf{y}}_t = \begin{bmatrix} \mathbf{y}_t \\ -\boldsymbol{\zeta}_t/\rho - \mathbf{v}_t \end{bmatrix}, \\ \overline{\mathbf{e}}_t = \begin{bmatrix} \mathbf{e}_t \\ \mathbf{d}_t \end{bmatrix}, \overline{\mathbf{R}}_t = \begin{bmatrix} \mathbf{R}_t & \mathbf{0} \\ \mathbf{0} & \mathbf{I}/\rho \end{bmatrix}. \end{aligned} \tag{9}$$

Hence, the solution in (8a) can be then computed by running the augmented Kalman filter and the RTS smoother on the following model:

$$p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t \mid \mathbf{A}_t\mathbf{x}_{t-1} + \mathbf{b}_t, \mathbf{Q}_t), \tag{10a}$$
$$p(\overline{\mathbf{y}}_t \mid \mathbf{x}_t) = \mathcal{N}(\overline{\mathbf{y}}_t \mid \overline{\mathbf{H}}_t\mathbf{x}_t + \overline{\mathbf{e}}_t, \overline{\mathbf{R}}_t). \tag{10b}$$

Following [14], we can reformulate the Kalman filter in an associative operator form by parametrizing the density functions $p(\mathbf{x}_t \mid \overline{\mathbf{y}}_t, \mathbf{x}_{t-1})$ and $p(\overline{\mathbf{y}}_t \mid \mathbf{x}_{t-1})$ as

$$\begin{aligned} p(\mathbf{x}_t \mid \overline{\mathbf{y}}_t, \mathbf{x}_{t-1}) &= \mathcal{N}(\mathbf{x}_t; \boldsymbol{\Phi}_t\mathbf{x}_{t-1} + \boldsymbol{\omega}_t, \mathbf{O}_t), \\ p(\overline{\mathbf{y}}_t \mid \mathbf{x}_{t-1}) &\propto \mathcal{N}_I(\mathbf{x}_{t-1}; \boldsymbol{\eta}_t, \mathbf{J}_t), \end{aligned} \tag{11}$$

where $\mathcal{N}_I(\mathbf{x}_{t-1}; \boldsymbol{\eta}_t, \mathbf{J}_t)$ denotes a information-form Gaussian density with information vector $\boldsymbol{\eta}_t$ and information (i.e., precision) matrix $\mathbf{J}_t$. For $t > 1$, the parameters are

$$\begin{aligned} \boldsymbol{\Phi}_t &= \left(\mathbf{I} - \mathbf{K}_t\overline{\mathbf{H}}_t\right)\mathbf{A}_t, \\ \boldsymbol{\omega}_t &= \mathbf{b}_t + \mathbf{K}_t\left(\overline{\mathbf{y}}_t - \overline{\mathbf{H}}_t\mathbf{b}_t - \overline{\mathbf{e}}_t\right), \\ \mathbf{O}_t &= \left(\mathbf{I} - \mathbf{K}_t\overline{\mathbf{H}}_t\right)\mathbf{Q}_t, \\ \mathbf{K}_t &= \mathbf{Q}_t\overline{\mathbf{H}}_t^\top\mathbf{S}_t^{-1}, \\ \mathbf{S}_t &= \overline{\mathbf{H}}_t\mathbf{Q}_t\overline{\mathbf{H}}_t^\top + \overline{\mathbf{R}}_t, \end{aligned} \tag{12}$$

and

$$\begin{aligned} \boldsymbol{\eta}_t &= \mathbf{A}_t^\top\overline{\mathbf{H}}_t^\top\mathbf{S}_t^{-1}\left(\overline{\mathbf{y}}_t - \overline{\mathbf{H}}_t\mathbf{b}_t - \overline{\mathbf{e}}_t\right), \\ \mathbf{J}_t &= \mathbf{A}_t^\top\overline{\mathbf{H}}_t^\top\mathbf{S}_t^{-1}\overline{\mathbf{H}}_t\mathbf{A}_t. \end{aligned} \tag{13}$$

When $t = 1$, the values of $(\boldsymbol{\Phi}_1, \boldsymbol{\omega}_1, \mathbf{O}_1, \boldsymbol{\eta}_1, \mathbf{J}_1)$ can be derived by a Kalman filter.

We can now define an associative operator $\otimes$ [14] which combines the aforementioned parameters as

$$\begin{aligned} &(\boldsymbol{\Phi}_i, \boldsymbol{\omega}_i, \mathbf{O}_i, \boldsymbol{\eta}_i, \mathbf{J}_i) \otimes (\boldsymbol{\Phi}_j, \boldsymbol{\omega}_j, \mathbf{O}_j, \boldsymbol{\eta}_j, \mathbf{J}_j) \\ &\quad = (\boldsymbol{\Phi}_{ij}, \boldsymbol{\omega}_{ij}, \mathbf{O}_{ij}, \boldsymbol{\eta}_{ij}, \mathbf{J}_{ij}), \end{aligned} \tag{14}$$

where

$$\begin{aligned} \boldsymbol{\Phi}_{ij} &= \boldsymbol{\Phi}_j\left(\mathbf{I} + \mathbf{O}_i\mathbf{J}_j\right)^{-1}\boldsymbol{\Phi}_i, \\ \boldsymbol{\omega}_{ij} &= \boldsymbol{\Phi}_j\left(\mathbf{I} + \mathbf{O}_i\mathbf{J}_j\right)^{-1}\left(\boldsymbol{\omega}_i + \mathbf{O}_i\boldsymbol{\eta}_j\right) + \boldsymbol{\omega}_j, \\ \mathbf{O}_{ij} &= \boldsymbol{\Phi}_j\left(\mathbf{I} + \mathbf{O}_i\mathbf{J}_j\right)^{-1}\mathbf{O}_i\boldsymbol{\Phi}_j^\top + \mathbf{O}_j, \\ \boldsymbol{\eta}_{ij} &= \boldsymbol{\Phi}_i^\top\left(\mathbf{I} + \mathbf{J}_j\mathbf{O}_i\right)^{-1}\left(\boldsymbol{\eta}_j + \mathbf{J}_j\boldsymbol{\omega}_i\right) + \boldsymbol{\eta}_i, \\ \mathbf{J}_{ij} &= \boldsymbol{\Phi}_i^\top\left(\mathbf{I} + \mathbf{J}_j\mathbf{O}_i\right)^{-1}\mathbf{J}_j\boldsymbol{\Phi}_i + \mathbf{J}_i. \end{aligned} \tag{15}$$

If we define the t-th prefix sum as

$$\begin{aligned} &(\boldsymbol{\Phi}_t^*, \boldsymbol{\omega}_t^*, \mathbf{O}_t^*, \boldsymbol{\eta}_t^*, \mathbf{J}_t^*) = (\boldsymbol{\Phi}_1, \boldsymbol{\omega}_1, \mathbf{O}_1, \boldsymbol{\eta}_1, \mathbf{J}_1) \\ &\quad \otimes (\boldsymbol{\Phi}_2, \boldsymbol{\omega}_2, \mathbf{O}_2, \boldsymbol{\eta}_2, \mathbf{J}_2) \ldots \otimes (\boldsymbol{\Phi}_t, \boldsymbol{\omega}_t, \mathbf{O}_t, \boldsymbol{\eta}_t, \mathbf{J}_t), \end{aligned} \tag{16}$$

then the filtering probability density is given as

$$p(\mathbf{x}_t \mid \overline{\mathbf{y}}_{1:t}) = \mathcal{N}(\mathbf{x}_t \mid \boldsymbol{\omega}_t^*, \mathbf{O}_t^*). \tag{17}$$

See [14, Theorem 2] for details. Because the operator $\otimes$ is associative, we can use a parallel scan algorithm [17] to compute all the prefix sums in parallel (using, e.g., `associative_scan` function in JAX [27]).

For parallelizing the Kalman smoother (i.e., RTS smoother), as in [14], we define

$$p(\mathbf{x}_t \mid \overline{\mathbf{y}}_{1:t}, \mathbf{x}_{t+1}) = \mathcal{N}\left(\mathbf{x}_t; \mathbf{E}_t\mathbf{x}_{t+1} + \mathbf{g}_t, \mathbf{L}_t\right), \tag{18}$$

for $t < T$

$$\begin{aligned} \mathbf{E}_t &= \hat{\mathbf{P}}_t\mathbf{A}_{t+1}^\top\left(\mathbf{A}_{t+1}\hat{\mathbf{P}}_t\mathbf{A}_{t+1}^\top + \mathbf{Q}_{t+1}\right)^{-1}, \\ \mathbf{g}_t &= \hat{\mathbf{x}}_t - \mathbf{E}_t\left(\mathbf{A}_{t+1}\hat{\mathbf{x}}_t + \mathbf{b}_{t+1}\right), \\ \mathbf{L}_t &= \hat{\mathbf{P}}_t - \mathbf{E}_t\mathbf{A}_{t+1}\hat{\mathbf{P}}_t, \end{aligned} \tag{19}$$

and for $t = T$

$$\begin{aligned} \mathbf{E}_T &= \mathbf{0}, \\ \mathbf{g}_T &= \hat{\mathbf{x}}_T, \\ \mathbf{L}_T &= \hat{\mathbf{P}}_T. \end{aligned} \tag{20}$$

Here, $\hat{\mathbf{x}}_t = \boldsymbol{\omega}_t^*$ and $\hat{\mathbf{P}}_t = \mathbf{O}_t^*$ are the Kalman filter means and covariances computed via the parallel Kalman filter. We then define an associative operator $\otimes$ via

$$(\mathbf{E}_i, \mathbf{g}_i, \mathbf{L}_i) \otimes (\mathbf{E}_j, \mathbf{g}_j, \mathbf{L}_j) = (\mathbf{E}_{ij}, \mathbf{g}_{ij}, \mathbf{L}_{ij}), \tag{21}$$

where

$$\begin{aligned} \mathbf{E}_{ij} &= \mathbf{E}_i\mathbf{E}_j, \\ \mathbf{g}_{ij} &= \mathbf{E}_i\mathbf{g}_j + \mathbf{g}_i, \\ \mathbf{L}_{ij} &= \mathbf{E}_i\mathbf{L}_j\mathbf{E}_i^\top + \mathbf{L}_i. \end{aligned} \tag{22}$$

We then write the t-th backward prefix sum as

$$(\mathbf{E}_t^*, \mathbf{g}_t^*, \mathbf{L}_t^*) = (\mathbf{E}_t, \mathbf{g}_t, \mathbf{L}_t) \otimes \ldots \otimes (\mathbf{E}_T, \mathbf{g}_T, \mathbf{L}_T). \tag{23}$$

Correspondingly, the smoother probability density can then be computed as

$$p(\mathbf{x}_t \mid \overline{\mathbf{y}}_{1:T}) = \mathcal{N}(\mathbf{x}_t \mid \mathbf{g}_t^*, \mathbf{L}_t^*), \tag{24}$$

where again the parameters can be computed in parallel via a parallel scan algorithm [17].

The performance advantage of the proposed approach becomes increasingly evident as the dataset grows. For larger numbers of time steps, the parallel algorithm significantly reduces computation time compared to the sequential algorithm. This enhanced efficiency makes it well-suited for real-time applications and large-scale state estimation problems. The computation steps of the proposed method are summarized in Algorithm 1.

Algorithm 1 Parallel Kalman smoothing augmented Lagrangian method

Input: $\mathbf{y}_t$, $\mathbf{A}_t$, $\mathbf{H}_t$, $\mathbf{C}_t$, $\mathbf{e}_t$, $\mathbf{b}_t$, $\mathbf{d}_t$, $\mathbf{Q}_t$, $\mathbf{R}_t$, $t = 1, \ldots, T$; parameter ρ; $\mathbf{m}_0$ and $\mathbf{P}_0$

Output: $\mathbf{x}_{1:T}$

1: **while** not converged **do**
2: $\quad\mathbf{x}_{1:T}^{(l)}$ is computed in parallel by all-prefix-sums operation on (16) and (23);
3: $\quad\mathbf{v}_{1:T}^{(l)}$ is computed in parallel by (8b);
4: $\quad\boldsymbol{\zeta}_{1:T}^{(l)}$ is computed in parallel by (8c);
5: **end while**

IV. Experimental Results

This section presents an experimental evaluation of the algorithm's performance through a simulated application.

We consider a four-dimensional linear tracking model [1], which includes positions (x_1, x_2) and velocities (v_1, v_2). The kind of tracking problem often arises in autonomous and semi-autonomous shipping applications [2]. The system state is $\mathbf{x}_t = \begin{bmatrix} x_{1,t} & x_{2,t} & v_{1,t} & v_{2,t} \end{bmatrix}^\top$. We simulate two sensors, each providing measurements of the positions. To distinguish different sensors, we denote the measurement matrix and the covariance matrix as $\mathbf{H}_{t,i}$ and $\mathbf{R}_{t,i}$, where i indicates the sensor number. The measurement matrices and covariance matrices corresponding to the two sensors are then set as

$$\mathbf{H}_{t,1} = \mathbf{H}_{t,2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$
$$\mathbf{R}_{t,1} = \begin{bmatrix} 0.5^2 & 0 \\ 0 & 0.5^2 \end{bmatrix},$$
$$\mathbf{R}_{t,2} = \begin{bmatrix} 0.4^2 & 0 \\ 0 & 0.4^2 \end{bmatrix}.$$

We then construct the measurement matrix $\mathbf{H}_{t,m}$ and covariance matrix $\mathbf{R}_{t,m}$ for the multi-sensor dynamic system by integrating the two sensors mentioned above. The measurement matrix is given by $\mathbf{H}_{t,m} = \begin{bmatrix} \mathbf{H}_{t,1}; & \mathbf{H}_{t,2} \end{bmatrix}$ and the covariance matrix by $\mathbf{R}_{t,m} = \begin{bmatrix} \mathbf{R}_{t,1} & \mathbf{0}; & \mathbf{0} & \mathbf{R}_{t,2} \end{bmatrix}$. Additionally, we impose the constraint matrix $\mathbf{C}_t = \begin{bmatrix} -1 & 0 & 0 & 0; & 0 & -1 & 0 & 0 \end{bmatrix}$ into the model, along with the constraint vector $\mathbf{d}_t = \begin{bmatrix} 0; & 0 \end{bmatrix}$ to ensure that displacements at every time step remain strictly positive. The other parameters $\mathbf{A}_t$ and $\mathbf{Q}_t$ are consistent with [2]. The hardware employs an Intel Core i9 $-$ 13900K CPU and an NVIDIA RTX 4090 GPU, providing a high-performance computing environment. The experiments were implemented using JAX [27].

We compare the estimation results from individual sensors and two fusing sensors in Fig. 1. It is observed that individual sensor and multi-sensor estimation can integrate the discrete predicted points into a smooth trajectory. To further evaluate the performance of the proposed method, we computed the relative error between the estimated trajectory and the true trajectory over the entire simulation period during the algorithm iterations. The relative error values during the augmented Lagrangian iteration are presented in Table I, which compares the results for sensor noise levels of the values 0.5 and 0.4. The variable l represents the iteration count of our method in Algorithm 1, and the symbol "$-$" indicates that the convergence condition has been satisfied, halting further iterations. The multi-sensor fusion model has a faster convergence rate, achieving the convergence condition in only 4 iterations, and the resulting error is lower.

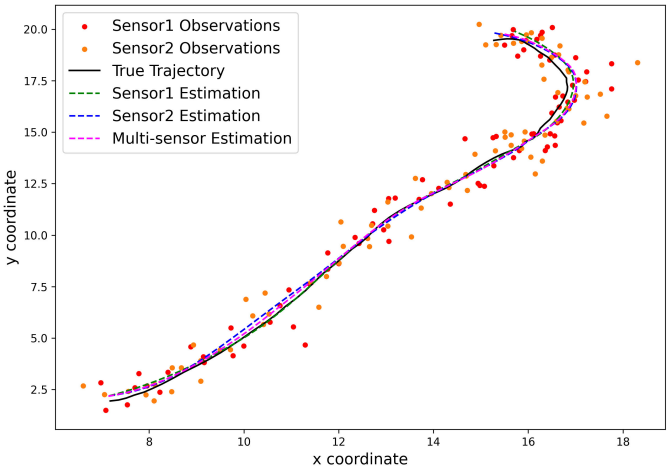
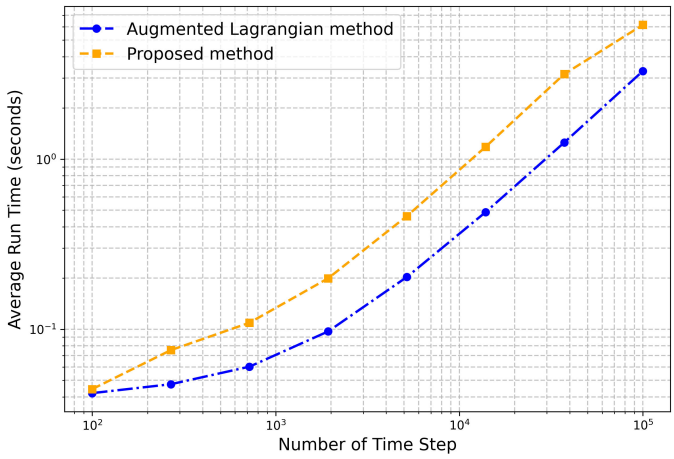
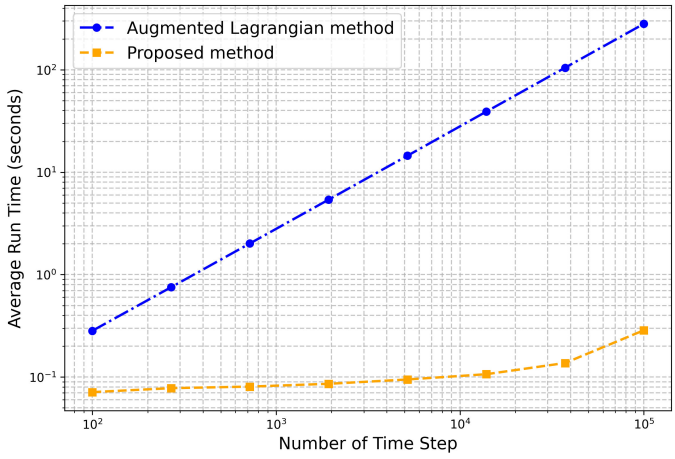


Fig. 1. Comparison of multi-sensor estimation and single sensor estimation.



(a) Runtime in CPU



(b) Runtime in GPU

Fig. 2. Runtime of the Kalman smoothing augmented Lagrangian method and its (proposed) parallel version on CPU and GPU.

TABLE I
RELATIVE ERROR OF DIFFERENT SENSOR SETTINGS

Iteration l	1	2	3	4	5	6
$\mathbf{R}_{t,1}$	0.1835	0.0467	0.0366	0.0365	0.0366	0.0366
$\mathbf{R}_{t,2}$	0.1281	0.0350	0.0322	0.0323	0.0323	–
$\mathbf{R}_{t,m}$	0.0832	0.0295	0.0294	0.0294	–	–

We tested the runtime of the sequential and parallel versions of the Kalman smoothing augmented Lagrangian method on the CPU and GPU for time steps ranging from 10^2 to 10^5. As depicted in Fig. 2(a), the parallel algorithm is actually slower on the CPU than the sequential one. This is because the CPU has too few cores to benefit from the parallelization in the method.

As shown in Fig. 2(b), on GPU, the performance of the proposed parallel method is significantly better than that of the sequential method. The runtime of the parallel algorithm on the GPU is significantly reduced, especially before the time step count reaches the number of GPU cores (16384), showing roughly a logarithmic growth trend until then. However, the runtime gradually increases after the time step count exceeds the number of GPU cores. This is because when the parallel computational load exceeds the number of our GPU cores, the data for parallel computation needs to be divided into several sequential steps. Nevertheless, the overall computation speed of the parallel method is still significantly higher than that of the sequential algorithm.

V. CONCLUSION

This paper proposes a parallel Kalman smoothing augmented Lagrangian method for constrained state estimation. The proposed method effectively combines the augmented Lagrangian framework with a parallel Kalman smoother method, enabling the decomposition of complex optimization problems into multiple computationally efficient subproblems. Our method demonstrates significant computational efficiency improvements in the experimental evaluations, particularly in large-scale state estimation scenarios.

REFERENCES

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

[2] R. Gao, S. Särkkä, R. Claveria-Vega, and S. Godsill, "Autonomous tracking and state estimation with generalized group Lasso," *IEEE Transactions on Cybernetics*, vol. 52, no. 11, pp. 12 056–12 070, Jun. 2021.

[3] S. J. Godsill and P. J. Rayner, *Digital Audio Restoration: A Statistical Model Based Approach*. New York: Springer-Verlag, 1998.

[4] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2024.

[5] R. Gao and S. Särkkä, "Augmented sigma-point Lagrangian splitting method for sparse nonlinear state estimation," in *Processdings of 28th European Signal Processing Conference (EUSIPCO)*. IEEE, Jan. 2021, pp. 2090–2094.

[6] Q. Legros and D. Fourer, "Estimation of instantaneous frequency and amplitude of multi-component signals using sparse modeling of signal innovation," in *Processdings of 32nd European Signal Processing Conference (EUSIPCO)*. IEEE, Oct. 2024, pp. 2502–2506.

[7] G. Joseph and P. K. Varshney, "State estimation of linear systems with sparse inputs and Markov-modulated missing outputs," in *Processdings of 30th European Signal Processing Conference (EUSIPCO)*. IEEE, Oct. 2022, pp. 837–841.

[8] K. Li and J. Príncipe, "Functional Bayesian filter," *IEEE Transactions on Signal Processing*, vol. 70, pp. 57–71, Dec. 2021.

[9] C. K. Thomas and D. Slock, "Generalized swept approximate message passing based Kalman filtering for dynamic sparse Bayesian learning," in *Processdings of 28th European Signal Processing Conference (EUSIPCO)*. IEEE, Jan. 2021, pp. 2065–2069.

[10] D. W. Kim, M. Park, and Y. Park, "Probabilistic modeling and Bayesian filtering for improved state estimation for soft robots," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1728–1741, Mar. 2021.

[11] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[12] H. E. Rauch, F. Tung, and C. T. Striebel, "Maximum likelihood estimates of linear dynamic system," *AIAA Journal*, vol. 3, no. 8, pp. 1445–1450, Aug. 1965.

[13] A. Bellés, D. Medina, P. Chauchat, S. Labsir, and J. Vilà-Valls, "Robust m-type error-state Kalman filters for attitude estimation," in *Processdings of 31st European Signal Processing Conference (EUSIPCO)*. IEEE, Sep. 2023, pp. 840–844.

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

[15] B. Ait-El-Fquih and I. Hoteit, "Parallel-and cyclic-iterative variational Bayes for fast Kalman filtering in large-dimensions," *IEEE Transactions on Signal Processing*, vol. 70, pp. 5871–5884, Dec. 2022.

[16] D. Moratuwage, B. N. Vo, B. T. Vo, and C. Shim, "Multi-scan multi-sensor multi-object state estimation," *IEEE Transactions on Signal Processing*, vol. 70, pp. 5429–5442, Oct. 2022.

[17] G. E. Blelloch, "Prefix sums and their applications," School of Computer Science, Carnegie Mellon University, Tech. Rep., 1990.

[18] T. Goldstein and S. Osher, "The split Bregman method for L1-regularized problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 323–343, Apr. 2009.

[19] D. W. Peaceman and H. H. Rachford, "The numerical solution of parabolic and elliptic differential equations," *Journal of the Society for Industrial and Applied Mathematics*, vol. 3, no. 1, pp. 28–41, Mar. 1955.

[20] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, Jul. 2011.

[21] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, Feb. 2020.

[22] S. J. Wright and J. Nocedal, *Numerical Optimization*. Springer Verlag, 2006.

[23] R. Gao, F. Tronarp, and S. Särkkä, "Variable splitting methods for constrained state estimation in partially observed Markov processes," *IEEE Signal Processing Letters*, vol. 27, pp. 1305–1309, Jul. 2020.

[24] R. Gao, F. Tronarp, Z. Zhao, and S. Särkkä, "Regularized state estimationand parameter learning via augmented Lagrangian Kalman smoother method," in *Processdings of IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*, Pittsburgh, PA, USA, Oct. 2019.

[25] Y. B. Shalom, X. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. Wiley, 2001.

[26] R. Gao, F. Tronarp, and S. Särkkä, "Iterated extended Kalman smoother-based variable splitting for L1-regularized state estimation," *IEEE Transactions on Signal Processing*, vol. 97, no. 19, pp. 5078–5092, Oct. 2019.

[27] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, "JAX: composable transformations of Python+NumPy programs," 2018. [Online]. Available: http://github.com/jax-ml/jax