Simultaneous Localization and Mapping: Through the Lens of Nonlinear Optimization

Amay Saxena, *Student Member, IEEE*, Chih-Yuan Chiu, *Graduate Student Member, IEEE*, Ritika Shrivastava, *Student Member, IEEE*, Joseph Menke, *Student Member, IEEE*, and Shankar Sastry, *Life Fellow, IEEE*

***Abstract*—Simultaneous Localization and Mapping (SLAM) algorithms perform visual-inertial estimation via filtering or batch optimization methods. Empirical evidence suggests that filtering algorithms are computationally faster, while optimization methods are more accurate. This work presents an optimization-based framework that unifies these approaches, and allows users to flexibly implement different design choices, e.g., the number and types of variables maintained in the algorithm at each time. We prove that filtering methods correspond to specific design choices in our generalized framework. We then reformulate the Multi-State Constrained Kalman Filter (MSCKF) and contrast its performance with that of sliding-window based filters. Our approach modularizes state-of-the-art SLAM algorithms to allow for adaptation to various scenarios. Experiments on the EuRoC MAV dataset verify that our implementations of these algorithms are competitive with the performance of off-the-shelf implementations in the literature. Using these results, we explain the relative performance characteristics of filtering and batch-optimization based algorithms in the context of our framework. We illustrate that under different design choices, our empirical performance interpolates between those of state-of-the-art approaches.**

***Index Terms*—SLAM, estimation, control, computer vision.**

I. Introduction

IN SIMULTANEOUS Localization and Mapping (SLAM), a robotic agent maps its uncharted environment while locating itself in the constructed map [1]. Applications include military map construction, search-and-rescue missions, augmented and virtual reality, and 3D scene capture [2]–[4].

Typical modern SLAM algorithms consist of *front* and *back ends*. The front end performs feature extraction, data association, and outlier rejection on raw sensor data. The back end then uses dynamics and measurement models for inference over the processed data, and produce compatible state estimates. Back end algorithms are often considered one of two classes—*Gaussian filtering* or *batch optimization* based. Filtering methods iteratively refine the distribution of recent states under a Gaussian prior [5]–[7], while optimization methods iteratively estimate states as solutions to an optimization problem, with objective constructed from inertial measurement unit (IMU) and image reprojection error terms. In particular, factor graph-based approaches efficiently solve optimization problems over past variables via factorization schemes that maintain the sparsity of the underlying least squares problem [8]–[11]. Keyframe-based methods are optimization-based approaches that retain only a small subset of maximally informative frames ("keyframes") in the optimization window arbitrarily spaced apart in time, while dropping all other poses [12]. Empirically, both classes of algorithms attain state-of-the-art performance, though the latter often attain higher accuracy at the cost of longer compute times [2], [12], [13].

Prior literature contrasted theoretical and empirical properties of filtering and batch optimization algorithms. Scaramuzza and Fraundorfer compared filtering and bundle adjustment-based methods for visual odometry [14], [15]. Frese *et al.* surveyed the use of grid-based and pose graph-based SLAM algorithms from a practitioner's perspective [16]. Huang and Dissanyake conducted a theoretical study of the consistency, accuracy, and computational speed of filtering, optimization-based, and pose-graph SLAM [17]. Khosoussi *et al.* exploited sparsity in SLAM problems by conditioning on estimates of robot orientations [18]. Strasdat *et al.* conducted Monte Carlo experiments on visual SLAM algorithms [19], revealed that including more features in the back end increased accuracy more (compared to including more frames), and concluded that bundle adjustment outperforms filtering, since its computation time increases less drastically with the number of features.

In this work, we build upon prior literature by formulating a unified optimization-based framework for the SLAM back end that encompasses a large class of existing, state-of-the-art SLAM algorithms. We use this unified framework to recast the Extended Kalman Filter (EKF), Multi-State Constrained Kalman Filter (MSCKF), and Open Keyframe Visual-Inertial SLAM algorithm (OKVIS) as optimization-based back-end algorithms, and compare the empirical performance of the reformulated MSCKF with that of sliding-window optimization-based back-end algorithms, including the keyframe-based approach of Open

Manuscript received February 24, 2022; accepted May 28, 2022. Date of publication June 10, 2022; date of current version June 15, 2022. This letter was recommended for publication by Associate Editor J. Guivant and Editor S. Behnke upon evaluation of the Reviewers' comments. This work was supported in part by NSF under Grant DMS 2013985 THEORINet: Transferable, Hierarchical, Expressive, Optimal, Robust and Interpretable Networks and and in part by the U.S. Office of Naval Research MURI under Grant N00014-16-1-2710. *(Corresponding author: Chih-Yuan Chiu.)*

The authors are with the Department of EECS at the University of California, Berkeley, CA 94720 USA (e-mail: amaysaxena@berkeley.edu; chihyuan_chiu@berkeley.edu; ritishri@berkeley.edu; joemenke@berkeley.edu; sastry@eecs.berkeley.edu).

Digital Object Identifier 10.1109/LRA.2022.3181409

Keyframe Visual-Inertial SLAM [12]. Somewhat surprisingly, the MSCKF outperforms sliding window filters of comparable sizes on several datasets, despite not performing multiple Gauss-Newton updates. We use our generalized framework to analyze these empirical findings.

II. SLAM: Formulation on Euclidean Spaces

A. SLAM on Euclidean Spaces

SLAM estimates two types of variables: *states* and *features*.

The state at each time t, denoted $x_t \in \mathbb{R}^{d_x}$, encodes information describing the robot, e.g., camera positions and orientations (poses). Feature positions available at time t in a global frame, denoted $\{f_{t,j} | j = 1, \ldots, p\} \subset \mathbb{R}^{d_f}$, can be obtained by analyzing information from image measurements $\{z_{t,j} | j = 1, \ldots, p\} \subset \mathbb{R}^{d_z}$ and state estimates; these describe the relative position of the robot in its environment.

States and features are described by an infinitely differentiable (i.e., C^∞) dynamics map $g : \mathbb{R}^{d_x} \to \mathbb{R}^{d_x}$ and a C^∞ measurement map $h : \mathbb{R}^{d_x} \times \mathbb{R}^{d_f} \to \mathbb{R}^{d_z}$, via additive noise models:

$$x_{t+1} = g(x_t) + w_t, \qquad w_t \sim \mathcal{N}(0, \Sigma_w), \tag{1}$$

$$z_{t,j} = h(x_t, f_{t,j}) + v_{t,j}, \qquad v_{t,j} \sim \mathcal{N}(0, \Sigma_v), \tag{2}$$

where $\Sigma_w \in \mathbb{R}^{d_x \times d_x}, \Sigma_w \succeq 0$ and $\Sigma_v \in \mathbb{R}^{d_z \times d_z}, \Sigma_v \succeq 0$.

For localization and mapping, SLAM algorithms maintain a *full state (vector)* $\overline{x_t} \in \mathbb{R}^d$, in which a number of past states and feature positions are concatenated. The exact number and time stamps of these states and features vary with the design choice of each SLAM algorithm. For example, sliding-window filters define the full state $\overline{x_t} := (x_{t-n+1}, \ldots, x_t, f_{t,p-q+1}, \ldots, f_{t,p}) \in \mathbb{R}^d$, with $d := d_x n + d_f q$, to be a sliding window of the most recent n states, consisting of one pose each, and the most recent estimates, at time t, of a collection of q features [5], [6]. Batch optimization methods, on the other hand, maintain all states and features encountered in the problem up to the current time [8]–[11].

Equations (1) and (2) do not involve overparameterized state variables, e.g., quaternion representations for poses, which are discussed in Section III.

B. SLAM as an Optimization Problem on Euclidean Spaces

SLAM estimates state and feature positions that best enforce constraints posed by given dynamics and measurement models, as well as noisy state and feature measurements collected over time. This is formulated as the minimization of the sum of weighted residual terms representing these constraints. For example, weighted residuals associated with the prior distribution over $\overline{x_t} \in \mathbb{R}^d$, the dynamics constraints between states $x_i, x_{i+1} \in \mathbb{R}^{d_x}$, and the reprojection error of feature $f_j \in \mathbb{R}^{d_f}$ corresponding to the state $x_i \in \mathbb{R}^{d_x}$ and image measurement $z_{t,j} \in \mathbb{R}^{d_z}$, may be given by $\Sigma_0^{-1/2}(\overline{x_t} - \mu_0) \in \mathbb{R}^d$, $\Sigma_w^{-1/2}(x_{i+1} - g(x_i)) \in \mathbb{R}^{d_x}$, and $\Sigma_v^{-1/2}(z_{i,j} - h(x_i, f_{t,j})) \in \mathbb{R}^{d_z}$, respectively (here, $1 \le i \le n-1$, $1 \le j \le q$). We define the running cost, $c : \mathbb{R}^{d_x n + d_f q} \to \mathbb{R}$, as the sum of weighted norm squares of these residuals. For example, for a sliding-window filtering algorithm for SLAM:

$$c(\overline{x_t}) := \|\overline{x_t} - \mu_0\|^2_{\Sigma_0^{-1}} + \sum_{i=t-n+1}^{t-1} \|x_{i+1} - g(x_i)\|^2_{\Sigma_w^{-1}} + \sum_{j=p-q+1}^{p} \sum_{i=t-n+1}^{t} \|z_{i,j} - h(x_i, f_{t,j})\|^2_{\Sigma_v^{-1}}, \tag{3}$$

where $\|v\|^2_A := v^\top A v$ for any real vector v and real matrix A of compatible dimension.

To formulate SLAM as a nonlinear least-squares problem, we stack all residual terms into one residual vector $C(\overline{x_t})$. For example, for the sliding-window filter given above:

$$\begin{aligned} C(\overline{x_t}) := \Bigg[& \left(\Sigma_0^{-1/2}(\overline{x_t} - \mu_0)\right)^\top \\ & \left(\Sigma_w^{-1/2}(x_{t-n+1} - g(x_{t-n}))\right)^\top \cdots \left(\Sigma_w^{-1/2}(x_t - g(x_{t-1}))\right)^\top \\ & \left(\Sigma_v^{-1/2}(z_{t-n+1,p-q+1} - h(x_{t-n+1}, f_{t,p-q+1}))\right)^\top \cdots \\ & \left(\Sigma_v^{-1/2}(z_{t-n+1,p} - h(x_{t-n+1}, f_{t,p}))\right)^\top \cdots \\ & \left(\Sigma_v^{-1/2}(z_{t,p-q+1} - h(x_t, f_{t,p-q+1}))\right)^\top \cdots \\ & \left(\Sigma_v^{-1/2}(z_{t,p} - h(x_t, f_{t,p}))\right)^\top \Bigg]^\top \in \mathbb{R}^{(2n-1)d_x + nqd_z}. \end{aligned}$$

Thus, $c(\overline{x_t}) = C(\overline{x_t})^\top C(\overline{x_t})$, and the SLAM problem is now reduced to the nonlinear least squares problem below:

$$\min_{\overline{x_t}} . c(\overline{x_t}) = \min_{\overline{x_t}} . C(\overline{x_t})^\top C(\overline{x_t}) \tag{4}$$

Section IV introduces the main algorithmic submodules used to find an approximate solution to (4).

III. SLAM: Formulation on Manifolds

Here, we generalize the SLAM formulation in Section II to the case where dynamical states are defined on smooth manifolds rather than Euclidean spaces. SLAM often involves the orientations of rigid bodies, which evolve on a smooth manifold embedded in an ambient space, e.g., rotation matrices expressed as unit quaternions. In such situations, we use *boxplus* ($\boxplus$) and *boxminus* ($\boxminus$) operators, defined below, to perform composition and difference operations in the iterative algorithm presented in Section IV, while enforcing constraints imposed by the manifold's geometric structure.

Suppose the full state x evolves on a smooth manifold $\mathcal{M}$, with $\dim(\mathcal{M}) = n$. For each $x \in \mathcal{M}$, let $\pi_x : U_x \to V_x$ be a diffeomorphic chart from an open neighborhood $U_x \subset \mathcal{M}$ of $x \in \mathcal{M}$ to an open neighborhood $V_x \subset \mathbb{R}^n$ of $0 \in \mathbb{R}^n$. Without loss of generality, suppose $\pi_x(x) = 0$. The operators $\boxplus : U_x \times V_x \to U_x$ and $\boxminus : U_x \times U_x \to V_x$ are defined by:

$$x \boxplus \delta = \pi_x^{-1}(\delta) \tag{5}$$

$$y \boxminus x = \pi_x(y) \tag{6}$$

In essence, $\boxplus$ adds a perturbation $\delta \in \mathbb{R}^n$, in local coordinates, to a state $x \in \mathcal{M}$, while $\boxminus$ extracts the difference $\delta \in \mathbb{R}^n$, in local coordinates, between states $x, x' \in \mathcal{M}$ covered by the same chart. Below, "δ" often describes an error or increment to a nominal state on the manifold.

A. Manifold Examples

This subsection gives examples of the $\boxplus$, $\boxminus$ and π operators for manifolds that occur widely in SLAM: the set of unit quaternions, $\mathbb{H}_u$, and the set of rotation matrices, $SO(3)$.

Each $q \in \mathbb{H}_u$ is expressed as $q = (q_u, \vec{q}_v)$ where $q_u \in \mathbb{R}$ and $\vec{q}_v \in \mathbb{R}^3$ denote the scalar and vector (imaginary) parts, respectively, with $\|q\| = \sqrt{q_u^2 + \|\vec{q}_v\|_2^2} = 1$ (JPL convention). Here, the coordinate map $\pi : \mathbb{H}_u \to \mathbb{R}^3$ is defined as the Log map on $\mathbb{H}_u$; its inverse π^{-1} is the Exp map. Specifically, we write each $q \in \mathbb{H}_u$ as $q = (\cos(\frac{\theta}{2}),\ \sin(\frac{\theta}{2})\vec{\omega})$ for some $\theta \in [0, \pi]$, $\vec{\omega} \in \mathbb{R}^3$ with $\|\vec{\omega}\| = 1$, i.e., the quaternion q implements a rotation about the axis $\vec{\omega}$ by θ radians counterclockwise. Then, $\pi : \mathbb{H}_u \to \mathbb{R}^3$ and $\pi^{-1} : B_\pi(0) \to \mathbb{H}_u$ are defined by: ($B_\pi(0) := \{x \in \mathbb{R}^3 : \|x\|_2 < \pi\}$ denotes the image of π)

$$\pi(q) = \mathrm{Log}(q) = \theta\vec{\omega},$$

$$\pi^{-1}(\theta\vec{\omega}) = \mathrm{Exp}(\theta\vec{\omega}) = (\cos(\theta/2),\ \sin(\theta/2)\vec{\omega}).$$

The $\boxplus$ and $\boxminus$ maps are then implemented via the standard quaternion product $\star : \mathbb{H}_u \times \mathbb{H}_u \to \mathbb{H}_u$:

$$q_a \boxplus \vec{\omega} = q_a \star \mathrm{Exp}(\vec{\omega})$$

$$q_a \boxminus q_b = \mathrm{Log}(q_b^{-1} \star q_a)$$

For $SO(3)$, we define $\boxplus$ and $\boxminus$ similarly, i.e.,

$$R_a \boxplus \vec{\omega} = R_a\ \mathrm{Exp}(\vec{\omega})$$

$$R_a \boxminus R_b = \mathrm{Log}(R_b^T R_a)$$

Often, the full state in a SLAM problem exists in the Cartesian product of a finite collection of manifolds, since it contains poses and features on their own manifolds. For a product manifold $\mathcal{M}_1 \times \mathcal{M}_2$, with projection, increment, and difference maps already defined on $\mathcal{M}_1$ and $\mathcal{M}_2$, we define $\boxplus$ and $\boxminus$ on $\mathcal{M}_1 \times \mathcal{M}_2$ by:

$$(g_1, g_2) \boxplus (\xi_1, \xi_2) = (g_1 \boxplus \xi_1, g_2 \boxplus \xi_2)$$

$$(g_1, g_2) \boxminus (h_1, h_2) = (g_1 \boxminus h_1, g_2 \boxminus h_2)$$

B. SLAM as an Optimization Problem on Manifolds

The SLAM problem can be formulated on manifolds using modified cost functions, where plus and minus operations are replaced with $\boxplus$ and $\boxminus$ when necessary. (Appendix A1).

IV. Main Algorithm

A. Algorithm Overview

This section details submodules for a general SLAM algorithm, using state variables and cost terms defined in Sections II and III. We first introduce a formulation on Euclidean spaces. Below, denote the state and concatenated cost vector by $\overline{x_t} \in \mathbb{R}^d$ and $C : \mathbb{R}^d \to \mathbb{R}^{d_C}$, respectively. (e.g., the sliding window filter in Section II would correspond to $d = d_x n + d_f q$ and $d_C = (2n-1)d_x + nqd_z$). The SLAM problem is then equivalent to solving the nonlinear least-squares problem (4), reproduced below:

$$\min_{\overline{x_t}} .c(\overline{x_t}) = \min_{\overline{x_t}} .\|C(\overline{x_t})\|_2^2.$$

B. Gauss-Newton Descent

Gauss-Newton descent minimizes $c(\overline{x_t})$ via Gauss-Newton steps (Alg. 3), by iteratively approximating $c(\overline{x_t})$ about a given linearization point $\overline{x_t}^\star$ as a linear least-squares cost term, i.e.,

$$\min_{\overline{x_t}} .c(\overline{x_t}) = \min_{\overline{x_t}} .\|\overline{x_t} - \mu_t\|_{\Sigma_t^{-1}}^2 + o(\overline{x_t} - \overline{x_t}^\star) \qquad (7)$$

for some $\mu_t \in \mathbb{R}^d$ and $\Sigma_t \in \mathbb{R}^{d \times d}$. The theorem below describes the linearization procedure required to obtain $\mu_t \in \mathbb{R}^d$ and $\Sigma_t \in \mathbb{R}^{d \times d}$, and the approximation involved.

Theorem 4.1: (*Gauss-Newton Step*) Let $\overline{x_t}^\star \in \mathbb{R}^d$ be a given linearization point, and suppose $J := \frac{\partial C}{\partial \overline{x_t}} \in \mathbb{R}^{d_C \times d}$ has full column rank. Applying a Gauss-Newton step (Alg. 3) to the cost $c(\overline{x_t})$, about $\overline{x_t}^\star \in \mathbb{R}^d$ yields the new cost:

$$c(\overline{x_t}) = \|\overline{x_t} - \mu_t\|_{\Sigma_t^{-1}}^2 + o(\overline{x_t} - \overline{x_t}^\star),$$

where $\mu_t \in \mathbb{R}^d$ and $\Sigma_t \in \mathbb{R}^{d \times d}$ are given by:

$$\Sigma_t \leftarrow (J^\top J)^{-1},$$

$$\mu_t \leftarrow \overline{x_t}^\star - (J^\top J)^{-1} J^\top C(\overline{x_t}^\star).$$

Proof: See Appendix (Section B2). ■

C. Marginalization of States

The marginalization step (Alg. 4) reduces the size of the SLAM problem by removing states that are no longer relevant, thus improving computational efficiency. First, we partition the overall state $\overline{x_t} \in \mathbb{R}^d = \mathbb{R}^{d_M + d_K}$ into a *marginalized component* $\overline{x_{t,M}} \in \mathbb{R}^{d_M}$, to be discarded from $\overline{x_t}$, and a *non-marginalized component* $\overline{x_{t,K}} \in \mathbb{R}^{d_K}$, to be kept. Then, we partition $c(\overline{x_t})$ into two cost terms: $c_1(\overline{x_{t,K}})$, which depends only on non-marginalized state components, and $c_2(\overline{x_{t,K}}, \overline{x_{t,M}})$ which depends on both marginalized and non-marginalized state components:

$$c(\overline{x_t}) = c(\overline{x_K}, \overline{x_M}) = c_1(\overline{x_K}) + c_2(\overline{x_K}, \overline{x_M})$$
$$= \|C_1(\overline{x_K})\|_2^2 + \|C_2(\overline{x_K}, \overline{x_M})\|_2^2.$$

Here, $C_1(\overline{x_K}) \in \mathbb{R}^{d_{C,1}}$ and $C_2(\overline{x_K}, \overline{x_M}) \in \mathbb{R}^{d_{C,2}}$ denote the concatenation of residuals associated with $c_1(\overline{x_K})$ and $c_2(\overline{x_K}, \overline{x_M})$ (with $d_C = d_{C,1} + d_{C,2}$). To remove $\overline{x_{t,M}} \in \mathbb{R}^{d_M}$ from the optimization problem, observe that:

$$\min_{\overline{x_t}} c(\overline{x_t}) = \min_{\overline{x_{t,K}}, \overline{x_{t,M}}} (c_1(\overline{x_{t,K}}) + c_2(\overline{x_{t,K}}, \overline{x_{t,M}}))$$
$$= \min_{\overline{x_{t,K}}} \left(\|C_1(\overline{x_{t,K}})\|_2^2 + \min_{\overline{x_{t,M}}} \|C_2(\overline{x_{t,K}}, \overline{x_{t,M}})\|_2^2 \right).$$

To remove $\overline{x_{t,M}}$, we approximate the solution to the inner minimization problem by a linear least-squares cost, i.e.:

$$\min_{\overline{x_{t,M}}} \|C_2(\overline{x_{t,K}}, \overline{x_{t,M}})\|_2^2 \approx \|\overline{x_{t,K}} - \overline{\mu}_{t,K}\|^2_{\overline{\Sigma}^{-1}_{t,K}}$$

for some $\overline{\mu}_{t,K} \in \mathbb{R}^{d_K}$ and $\overline{\Sigma}_{t,K} \in \mathbb{R}^{d_K \times d_K}$. Since $\|C_2(\overline{x_{t,K}}, \overline{x_{t,M}})\|_2^2$ is in general non-convex, we obtain $\overline{\mu}_{t,K}$ and $\overline{\Sigma}_{t,K}$ by minimizing the first-order Taylor expansion of $\|C_2(\overline{x_{t,K}}, \overline{x_{t,M}})\|_2^2$ about some linearization point. Below, Theorem 4.2 details the derivation of $\overline{\mu}_{t,K}$ and $\overline{\Sigma}_{t,K}$. (For the proof, see Appendix B3).

Theorem 4.2 (Marginalization Step): Let $\overline{x_t}^\star \in \mathbb{R}^d$ be a given linearization point, and suppose $J := \frac{\partial C}{\partial \overline{x_t}} \in \mathbb{R}^{d_C \times d}$ has full column rank. Define $J_K := \frac{\partial C}{\partial \overline{x_{t,K}}} \in \mathbb{R}^{d_C \times d_K}$, $J_M := \frac{\partial C}{\partial \overline{x_{t,M}}} \in \mathbb{R}^{d_C \times d_M}$. If $C(\overline{x_{t,M}}, \overline{x_{t,K}})$ were a linear function of $\overline{x_t} = (\overline{x_{t,M}}, \overline{x_{t,K}})$, then applying a Marginalization step (Alg. 4) to the cost $c(\overline{x_t})$, about the linearization point $\overline{x_t}^\star = (\overline{x^\star_{t,K}}, \overline{x^\star_{t,M}}) \in \mathbb{R}^d$, yields:

$$\min_{\overline{x_t}} c(\overline{x_{t,K}}, \overline{x_{t,M}}) = \min_{\overline{x_{t,K}}} . \left(c_1(\overline{x_{t,K}}) + \min_{\overline{x_{t,M}}} c_2(\overline{x_{t,K}}, \overline{x_{t,M}}) \right), \tag{8}$$

where $\Sigma_{t,K} \in \mathbb{R}^{d_K \times d_K}$ and $\mu_{t,K} \in \mathbb{R}^{d_K}$ are given by:

$$\Sigma_{t,K} := \left(J_K^\top \left[I - J_M (J_M^\top J_M)^{-1} J_M^\top \right] J_K \right)^{-1}, \tag{9}$$

$$\mu_{t,K} := \overline{x^\star_{t,K}} - \Sigma_{t,K} J_K^\top \left[I - J_M (J_M^\top J_M)^{-1} J_M^\top \right] C_2(\overline{x^\star_t}). \tag{10}$$

D. Main Algorithm on Manifolds

The Euclidean-space framework above can be directly extended to a formulation on manifolds, by using concepts in Section III to modify the dynamics and measurement maps in Section II, as well as the cost functions, Gauss-Newton steps, and marginalization steps in Sections IV-B, IV-C. When appropriate, plus and minus operations must be replaced with $\boxplus$ and $\boxminus$ (Appendix A2).

V. Equivalence of Filtering and Optimization Approaches

Here, we demonstrate the equivalence of filtering and batch optimization-based SLAM algorithms, using the Extended Kalman Filter (EKF, in Section V-A) and Multi-State Constrained Kalman Filter (MSCKF, in Section V-B), as examples. Although similar results exist in the optimization literature, they do not analyze algorithmic submodules unique to SLAM, e.g., feature incorporation and discarding [20]. For an introduction to the classical formulations of EKF and MSCKF SLAM, please see Appendices C2, C4.

A. Extended Kalman Filter (EKF), on Euclidean Spaces

At each time t, the EKF SLAM algorithm on Euclidean spaces maintains the full state vector $\tilde{x}_t := (x_t, f_{t,1}, \ldots, f_{t,p}) \in \mathbb{R}^{d_x + pd_f}$, consisting of the most recent state $x_t \in \mathbb{R}^{d_x}$ and feature position estimates $f_{t,1}, \ldots, f_{t,p} \in \mathbb{R}^{d_f}$. At initialization ($t = 0$), no feature has been detected ($p = 0$), and the EKF full state is simply the initial state $\tilde{x}_0 = x_0 \in \mathbb{R}^{d_x}$, with mean $\mu_0 \in \mathbb{R}^{d_x}$ and covariance $\Sigma_0 \in \mathbb{R}^{d_x \times d_x}$. Suppose, at the current time t, the running cost $c_{EKF,t,0} : \mathbb{R}^{d_x + pd_f} \to \mathbb{R}^{d_x + pd_f}$ is:

$$c_{EKF,t,0} = \|\tilde{x}_t - \mu_t\|^2_{\Sigma_t^{-1}},$$

where $\tilde{x}_t := (x_t, f_{t,1}, \ldots, f_{t,p}) \in \mathbb{R}^{d_x + pd_f}$ denotes the EKF full state at time t, with mean $\mu_t \in \mathbb{R}^{d_x + pd_f}$ and covariance $\Sigma_t \in \mathbb{R}^{(d_x + pd_f) \times (d_x + pd_f)}$. First, the *feature augmentation step* appends position estimates of new features $f_{p+1}, \ldots, f_{p+p'} \in \mathbb{R}^{d_f}$ to the EKF full state $\tilde{x}_t$, and updates its mean and covariance. In particular, feature measurements $z_{t,p+1}, \ldots, z_{t,p+p'} \in \mathbb{R}^{d_z}$ are incorporated by adding measurement residual terms to the current running cost $c_{EKF,t,0}$, creating a new cost $c_{EKF,t,1} : \mathbb{R}^{d_x + (p+p')d_f} \to \mathbb{R}$:

$$\begin{aligned} &c_{EKF,t,1}(\tilde{x}_t, f_{t,p+1}, \ldots, f_{t,p+p'}) \\ &:= \|\tilde{x}_t - \mu_t\|^2_{\Sigma_t^{-1}} + \sum_{k=p+1}^{p+p'} \|z_{t,k} - h(x_t, f_{t,k})\|^2_{\Sigma_v^{-1}}. \end{aligned}$$

In effect, $c_{EKF,t,1}$ appends positions of new features to $\tilde{x}_t$, and constrains it using feature measurements residuals.

Next, the *feature update* step uses measurements of features contained in $\tilde{x}_t$ to update the mean and covariance of $\tilde{x}_t$. More precisely, feature measurements $z_{t,1:p} := (z_{t,1}, \ldots, z_{t,p}) \in \mathbb{R}^{pd_z}$, of the p features $f_1, \ldots, f_p$ included in $\tilde{x}_t$, are introduced by incorporating associated measurement residuals

to the running cost $c_{EKF,t,0}$, creating a new cost $c_{EKF,t,2} : \mathbb{R}^{d_x + pd_f} \to \mathbb{R}$:

$$c_{EKF,t,2}(\tilde{x}_t) := \|\tilde{x}_t - \mu_t\|^2_{\Sigma_t^{-1}} + \sum_{k=1}^{p} \|z_{t,k} - h(x_t, f_{t,k})\|^2_{\Sigma_v^{-1}}.$$

A Gauss-Newton step then constructs an updated mean $\overline{\mu_t} \in \mathbb{R}^{d_x + pd_f}$ and covariance $\overline{\Sigma}_t \in \mathbb{R}^{(d_x + pd_f) \times (d_x + pd_f)}$ for $\tilde{x}_t$, creating a new cost $c_{EKF,t,3} : \mathbb{R}^{d_x + pd_f} \to \mathbb{R}$:

$$c_{EKF,t,3}(\tilde{x}_t) := \|\tilde{x}_t - \overline{\mu_t}\|^2_{\overline{\Sigma}_t^{-1}},$$

which returns the running cost to the form of $c_{EKF,t,0}$.

The *state propagation* step propagates the EKF full state forward by one time step, via the EKF state propagation map $g : \mathbb{R}^{d_x + pd_f} \to \mathbb{R}^{d_x + pd_f}$.

To propagate $\tilde{x}_t$ forward in time, we incorporate the dynamics residual to the running cost $c_{EKF,t,0}$ to create a new cost $c_{EKF,t,4} : \mathbb{R}^{2d_x + pd_f} \to \mathbb{R}$:

$$c_{EKF,t,4}(\tilde{x_t}, x_{t+1}) := \|\tilde{x}_t - \overline{\mu_t}\|^2_{\overline{\Sigma}_t^{-1}} + \|x_{t+1} - g(x_t)\|^2_{\Sigma_w^{-1}}.$$

In effect, $c_{EKF,t,4}$ appends the new state $x_{t+1} \in \mathbb{R}^{d_x}$ to $\tilde{x}_t$, while adding a new constraint posed by the dynamics residuals. A marginalization step, with $\tilde{x}_{t,K} := (x_{t+1}, f_{t,1}, \ldots, f_{t,p}) \in \mathbb{R}^{d_x + pd_f}$ and $\tilde{x}_{t,M} := x_t \in \mathbb{R}^{d_x}$, then removes the previous state $x_t \in \mathbb{R}^{d_x}$ from the running cost. This step produces a mean $\mu_{t+1} \in \mathbb{R}^{d_x + pd_f}$ and a covariance $\Sigma_{t+1} \in \mathbb{R}^{(d_x + pd_f) \times (d_x + pd_f)}$ for the new EKF full state, $\tilde{x}_{t+1} := \tilde{x}_{t,K}$. The running cost

is updated to $c_{EKF,t+1,0} : \mathbb{R}^{d_x + pd_f} \to \mathbb{R}$:

$$c_{EKF,t+1,0}(\tilde{x}_{t+1}) := \|\tilde{x}_{t+1} - \mu_{t+1}\|^2_{\Sigma_{t+1}^{-1}},$$

which returns the running cost to the form of $c_{EKF,t,0}$.

The theorems below establish that the feature augmentation, feature update, and state propagation steps of the EKF, presented above in our optimization framework, correspond precisely to those presented in the standard EKF SLAM algorithm (Alg. 5) [5], [21]. (For proofs, see Appendix C3).

Theorem 5.1: The feature augmentation step of standard EKF SLAM (Alg. 6) is equivalent to applying a Gauss-Newton step to $c_{EKF,t,1} : \mathbb{R}^{(d_x+pd_f)+p'd_f} \to \mathbb{R}$, with:

$$\begin{aligned} &c_{EKF,t,1}(\tilde{x}_t, f_{t,p+1}, \ldots, f_{t,p+p'}) \\ &= \|\tilde{x}_t - \mu_t\|^2_{\Sigma_t^{-1}} + \sum_{k=p+1}^{p+p'} \|z_{t,k} - h(x_t, f_{t,k})\|^2_{\tilde{\Sigma}_v^{-1}}. \end{aligned}$$

Theorem 5.2: The feature update step of standard EKF SLAM (Alg. 7) is equivalent to applying a Gauss-Newton step on $c_{EKF,t,2} : \mathbb{R}^{d_x+pd_f} \to \mathbb{R}$, with:

$$c_{EKF,t,2}(\tilde{x}_t) := \|\tilde{x}_t - \mu_t\|^2_{\Sigma_t^{-1}} + \sum_{k=1}^{p} \|z_{t,k} - h(x_t, f_{t,k})\|^2_{\Sigma_v^{-1}}.$$

Theorem 5.3: The state propagation step of standard EKF SLAM (Alg. 8) is equivalent to applying a Marginalization step to $c_{EKF,t,4} : \mathbb{R}^{2d_x+pd_f} \to \mathbb{R}$, with:

$$c_{EKF,t,4}(\tilde{x}_t, x_{t+1}) := \|\tilde{x}_t - \overline{\mu}_t\|^2_{\overline{\Sigma}_t^{-1}} + \|x_{t+1} - g(x_t)\|^2_{\Sigma_w^{-1}}.$$

where $\tilde{x}_{t,K} := (x_{t+1}, f_{t,1}, \ldots, f_{t,p}) \in \mathbb{R}^{d_x+pd_f}$ and $\tilde{x}_{t,M} = x_t \in \mathbb{R}^d$.

Remark V.1: In practice, Gauss-Newton steps for pose augmentation can be delayed and done with feature updates.

Algorithm 1: EKF SLAM on Euclidean Spaces, as an Iterative Optimization Problem.

Data: Prior $\mathcal{N}(\mu_0, \Sigma_0)$ on $x_0 \in \mathbb{R}^{d_x}$, noise covariances Σ_w, Σ_v, dynamics map g, measurement map h, time horizon T.
Result: Estimates $\hat{x}_t \in \mathbb{R}^{d_x}$, $\forall t \in \{1, \cdots, T\}$.

$f_0(x) \leftarrow \|x_0 - \mu_0\|^2_{\Sigma_0^{-1}}$
$p \leftarrow 0$.
for $t = 0, 1, \cdots T$ **do**
  $(z_{t,p+1}, \cdots, z_{t,p+p'}) \leftarrow$ Measurements of new features.
  $\text{cost}_t \leftarrow \text{cost}_t + \sum_{k=p+1}^{p+p'} \|z_{t,k} - h(x_t, f_k)\|^2_{\Sigma_v^{-1}}$
  $\bar{\mu}_t \leftarrow \left(\bar{\mu}_t, \ell(x_t, z_{t,p+1}), \cdots, \ell(x_t, z_{t,p+p'})\right) \in \mathbb{R}^{d_x+(p+p')d_f}$.
  $\bar{\mu}_t, \bar{\Sigma}_t, \text{cost}_t \leftarrow$ 1 Gauss-Newton step on cost_t, about $\overline{\mu_t}$ (Alg. 3).
  $p \leftarrow p + p'$
  $(z_{t,1}, \cdots, z_{t,p}) \leftarrow$ Measurements of existing features.
  $\text{cost}_t \leftarrow \text{cost}_t + \sum_{k=1}^{p} \|z_{t,k} - h(x_t, f_k)\|^2_{\Sigma_v^{-1}}$
  $\bar{\mu}_t, \bar{\Sigma}_t, \text{cost}_t \leftarrow$ 1 Gauss-Newton step on cost_t, about μ_t, (Alg. 3).
  $\hat{x}_t \leftarrow \bar{\mu}_t \in \mathbb{R}^{d_x+pd_f}$.
  if $t < T$ **then**
    $\text{cost}_t \leftarrow \text{cost}_t + \|x_{t+1} - g(x_t)\|^2_{\Sigma_w^{-1}}$
    $\mu_{t+1}, \Sigma_{t+1}, \text{cost}_t \leftarrow$ 1 Marginalization step on cost_{t+1} with $x_M = x_t$, about $(\overline{\mu_t}, g(\overline{\mu_t}))$ (Alg. 4).
    $\text{cost}_{t+1} \leftarrow \|x_{t+1} - \mu_{t+1}\|^2_{\Sigma_{t+1}^{-1}}$
  end
end
return $\hat{x}_0, \cdots, \hat{x}_T$

B. Multi-State Constrained Kalman Filter (MSCKF), on Manifolds

The MSCKF algorithm maintains a full state, $\tilde{x}_t \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$, containing the most recent IMU state, $x_{\text{IMU}} \in \mathcal{X}_{\text{IMU}}$ and n recent poses, $(x_1, \ldots, x_n) \in (\mathcal{X}_p)^n$:

$$\tilde{x}_t := (x_{t,\text{IMU}}, x_1, \ldots, x_n) \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n,$$

with mean $\mu_t \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$ and covariance $\Sigma_t \in \mathbb{R}^{(d_{\text{IMU}}+nd_x)\times(d_{\text{IMU}}+nd_x)}$. As new poses are introduced, old poses are discarded, and features are marginalized to update $\tilde{x}_t$, the mean μ_t, covariance Σ_t, and $n \in \mathbb{N}$ accordingly.

At initialization ($t = 0$), no pose has yet been recorded ($n = 0$), and the full state $\tilde{x}_0$ is the initial IMU state $\tilde{x}_{0,\text{IMU}} \in \mathcal{X}_{\text{IMU}}$, with mean $\mu_0 \in \mathcal{X}_{\text{IMU}}$ and covariance $\Sigma_0 \in \mathbb{R}^{d_{\text{IMU}} \times d_{\text{IMU}}}$. Thus, $\tilde{x}_0 = \mu_0$ optimizes the initial running cost $c_{MSCKF,0} : \mathcal{X}_{\text{IMU}} \to \mathbb{R}$ in our algorithm:

$$c_{MSCKF,0,0}(\tilde{x}_0) = \|\tilde{x}_0 \boxminus \mu_0\|^2_{\Sigma_0^{-1}}.$$

Suppose that, at the current time t, the running cost $c_{MSCKF,t,0} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \to \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$ is:

$$c_{MSCKF,t,0}(\tilde{x}_t) = \|\tilde{x}_t \boxminus \mu_t\|^2_{\Sigma_t^{-1}},$$

where $\mu_t \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$ and $\Sigma_t \in \mathbb{R}^{(d_{\text{IMU}}+nd_x)\times(d_{\text{IMU}}+nd_x)}$ denote the mean and covariance of the full state $\tilde{x}_t := (x_{t,\text{IMU}}, x_1, \ldots, x_n) \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$ at time t, consisting of the current IMU state and n poses. When a new image is received, the *pose augmentation step* adds a new pose $x_{n+1} \in \mathcal{X}_p$ (global frame) to $\tilde{x}_t$, derived from $x^{\text{IMU}}_{n+1} \in \mathcal{X}_{\text{IMU}}$, the IMU position estimate in the global frame, via the map $\psi : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \times \mathcal{X}_{\text{IMU}} \to \mathcal{X}_p$, i.e.,

$$x_{n+1} := \psi\left(\tilde{x}_t, x^{\text{IMU}}_{n+1}\right) \in \mathcal{X}_p.$$

The *feature update step* uses features measurements to update the mean and covariance of $\tilde{x}_t$. In MSCKF, features are discarded if (A) unobserved in the current pose, or (B) $n \geq N_{\max}$, a specified upper bound, in which case $\lfloor N_{\max} \rfloor / 3$ of the n poses, evenly spaced in time, are dropped after features common to these poses are marginalized. Let $S_{z,1}$ and $S_{z,2}$ denote sets of pose-feature pairs (x_i, f_j) from cases (A) and (B) above, respectively, and let S_f denote the set of features to be marginalized (Alg. 9). These constraints are then incorporated into the running cost, creating a new cost $c_{MSCKF,t,2} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \to \mathbb{R}$:

$$\begin{aligned} &c_{MSCKF,t,2}(\tilde{x}_t) \\ &:= \|\tilde{x}_t \boxminus \mu_t\|^2_{\Sigma_t^{-1}} + \sum_{(x_i,f_j)\in S_{z,1}\cup S_{z,2}} \|z_{i,j} \boxminus h(x_i, f_j)\|^2_{\Sigma_v^{-1}}, \end{aligned}$$

where $z_{i,j} \in \mathbb{R}^{d_z}$ denotes the feature measurement of feature j observed from pose $x_i \in \mathcal{X}_p$. By using Gauss-Newton linearization, we leverage constraints posed by the measurement residuals to construct an updated mean for $\tilde{x}_t$, denoted $\overline{\mu_t} \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$, and an updated covariance for $\tilde{x}_t$, denoted

$\overline{\Sigma}_t \in \mathbb{R}^{(d_{\text{IMU}}+nd_x)\times(d_{\text{IMU}}+nd_x)}$. As a result, our cost will be updated to $c_{MSCKF,t,3} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \to \mathbb{R}$:

$$c_{MSCKF,t,3}(\tilde{x}_t) := \|\tilde{x}_t \boxminus \overline{\mu_t}\|^2_{\overline{\Sigma}_t^{-1}},$$

which assumes the form of $c_{MSCKF,t,0}$.

The *state propagation* step propagates the full state by incorporating dynamics residuals into the running cost $c_{MSCKF,t,0}$, creating a new cost $c_{MSCKF,t,4} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \times \mathcal{X}_{\text{IMU}} \to \mathbb{R}$:

$$\begin{aligned} &c_{MSCKF,t,4}(\tilde{x}_t, x_{t+1,\text{IMU}}) \\ &:= \|\tilde{x}_t \boxminus \overline{\mu_t}\|^2_{\overline{\Sigma}_t^{-1}} + \|x_{t+1,\text{IMU}} \boxminus g_{\text{IMU}}(x_{t,\text{IMU}})\|^2_{\Sigma_t^{-1}}. \end{aligned}$$

In effect, $c_{MSCKF,t,4}$ appends the new IMU variable $x_{t+1,\text{IMU}} \in \mathcal{X}_{\text{IMU}}$ to the current full state $\tilde{x}_t \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$, and constrains this new full state via the dynamics residuals. A marginalization step, with $\tilde{x}_{t,K} := (x_{t+1,\text{IMU}}, x_1, \ldots, x_n) \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$ and $\tilde{x}_{t,M} := x_{t,\text{IMU}} \in \mathcal{X}_{\text{IMU}}$, then removes the previous IMU state, $x_{t,\text{IMU}}$, from the running cost. This produces a mean $\mu_{t+1} \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$ and a covariance $\Sigma_{t+1} \in \mathbb{R}^{(d_{\text{IMU}}+nd_x)\times(d_{\text{IMU}}+nd_x)}$ for the new MSCKF full state, $\tilde{x}_{t+1} := \tilde{x}_{t,K} = (x_{t+1,\text{IMU}}, x_1, \ldots, x_n) \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$. The running cost

is updated to $c_{MSCKF,t+1,0} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \to \mathbb{R}$:

$$c_{MSCKF,t+1,0}(\tilde{x}_{t+1}) := \|\tilde{x}_{t+1} \boxminus \mu_{t+1}\|^2_{\Sigma_{t+1}^{-1}},$$

which returns the running cost to the form of $c_{MSCKF,t,0}$.

The theorems below establish that the feature augmentation, feature update, and state propagation steps of the MSCKF, presented above in our optimization framework, correspond precisely to those presented in the standard MSCKF (Alg. 9) [6]. (For proofs, see Appendix C5).

Theorem 5.4: The pose augmentation step of the standard MSCKF (Alg. 10) is equivalent to applying a Gauss-Newton step to $c_{MSCKF,t,1} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \times \mathcal{X}_{\text{IMU}} \to \mathbb{R}$, with:

$$\begin{aligned} &c_{MSCKF,t,1}(\tilde{x}_t, x_{n+1}) \\ &= \|\tilde{x}_t \boxminus \mu_t\|^2_{\Sigma_t^{-1}} + \epsilon^{-1}\|x_{n+1} \boxminus \psi(\tilde{x}_t, x^{\text{IMU}}_{n+1})\|^2_2, \end{aligned}$$

and taking $\epsilon \to 0$ in the resulting (augmented) mean μ_t and covariance Σ_t.

Theorem 5.5: The feature update step of the standard MSCKF (Alg. 11) is equivalent to applying a Marginalization step to $c_{MSCKF,t,2} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \times \mathbb{R}^{|S_f|d_f} \to \mathbb{R}$, with:

$$\begin{aligned} &c_{MSCKF,t,2}(\tilde{x}_t, f_{S_f}) \\ &:= \|\tilde{x}_t \boxminus \mu_t\|^2_{\Sigma_t^{-1}} + \sum_{(x_i,f_j)\in S_{z,1}\cup S_{z,2}} \|z_{i,j} \boxminus h(x_i, f_j)\|^2_{\Sigma_v^{-1}}, \end{aligned}$$

where $f_{S_f} \in \mathbb{R}^{|S_f|d_f}$ denotes the stacked vector of all feature positions in S_f (see Alg. 9).

Theorem 5.6: The state propagation step of the standard MSCKF (Alg. 11) is equivalent to applying a Marginalization step to $c_{MSCKF,t,4} : \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n \times \mathcal{X}_{\text{IMU}} \to \mathbb{R}$, with:

$$\begin{aligned} &c_{MSCKF,t,4}(\tilde{x}_t, x_{t+1,\text{IMU}}) \\ &:= \|\tilde{x}_t \boxminus \overline{\mu_t}\|^2_{\overline{\Sigma}_t^{-1}} + \|x_{t+1,\text{IMU}} \boxminus g_{\text{IMU}}(x_{t,\text{IMU}})\|^2_{\Sigma_t^{-1}}. \end{aligned}$$

Algorithm 2: Multi-State Constrained Kalman Filter (MSCKF) on Manifolds, as Iterative Optimization.

Data: Prior $\mathcal{N}(\mu_0, \Sigma_0)$ on $x_{\text{IMU},0} \in \mathcal{X}_{\text{IMU}}$, noise covariances Σ_w, Σ_v, dynamics g_{IMU}, measurement map h, time horizon T, Pose transform ψ (IMU $\to$ global) , $\epsilon > 0$.
Result: Estimates $\hat{x}_t$ for all desired timesteps $t \in \{1, \cdots, T\}$.

$\text{cost}_t \leftarrow \|x_0 \boxminus \mu_0\|^2_{\Sigma_0}$. (Initialize objective function).
$S_z, S_x, S_{z,1}, S_{z,2} \leftarrow \phi$
$(n, p) \leftarrow (0, 0)$
for $t = 0, \cdots, T$ **do**
  while new pose $x_{n+1} \in \mathcal{X}_p$ recorded, new IMU measurement not received **do**
    $\text{cost}_t \leftarrow \text{cost}_t + \epsilon^{-1}\|x_{n+1} \boxminus \psi(\tilde{x}_t, x^{\text{IMU}}_{n+1})\|^2_2$.
    $\mu_t, \Sigma_t, \text{cost}_t \leftarrow$ 1 Gauss-Newton cost_t (Alg. 3), about $(\mu_t, \psi(\mu_t, x^{\text{IMU}}_{n+1}))$ with $\epsilon \to 0$.
    $\{z_{n+1,j}\} \leftarrow$ Feature measurements at x_{n+1}
    $S_z \leftarrow S_z \cup \{(x_{n+1}, f_j) | f_j \text{ observed at } n+1\}$
    $n \leftarrow n + 1$
    if $n \geq N_{\max} - 1$ **then**
      $S_x \leftarrow \{x_i | i \bmod 3 = 2, \text{ and } 1 \leq i \leq n.\}$
      $S_{z,1} \leftarrow \{(x_i, f_j) \in S_z | x_i \in S_x, \text{feature } j \text{ observed at each pose in } S_x\}$
    end
    $S_{z,2} \leftarrow \{(x_i, f_j) \in S_z | f_j \text{ not observed at } x_n\}$.
    $\text{cost}_t \leftarrow \text{cost}_t + \sum_{(x_i,f_j)\in S_{z,1}\cup S_{z,2}} \|z_{i,j} \boxminus h(x_i, f_{t,j})\|_{\Sigma_v^{-1}}$
    $\overline{\mu_t}, \overline{\Sigma_t}, \text{cost}_t \leftarrow$ 1 Gauss-Newton step on cost_t, about μ_t (Alg. 3)
    $\hat{x}_t \leftarrow \overline{\mu_t} \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$.
    $S_z \leftarrow S_z \backslash (S_{z,1} \cup \{(x_i, f_j) | x_i \in S_x\})$
    Reindex poses and features in ascending order.
    $(p, n) \leftarrow (p - |S_f|, n - |S_x|)$
  end
  if $t < T$ **then**
    $\text{cost}_t \leftarrow \text{cost}_t + \|x_{t+1,\text{IMU}} \boxminus g_{\text{IMU}}(x_{t,\text{IMU}})\|^2_{\Sigma_w^{-1}}$.
    $\mu_{t+1}, \Sigma_{t+1}, \text{cost}_t \leftarrow$ 1 Marginalization step on cost_t, about $(\overline{\mu_t}, g(\mu_{t,\text{IMU}}))$ (Alg. 4)
  end
end
return $\hat{x}_0, \cdots \hat{x}_T \in \mathcal{X}_{IMU} \times (\mathcal{X}_p)^n$

with $\tilde{x}_{t,K} := (x_{t+1,\text{IMU}}, x_1, \ldots, x_n) \in \mathcal{X}_{\text{IMU}} \times (\mathcal{X}_p)^n$ and $\tilde{x}_{t,M} = x_{t,\text{IMU}} \in \mathcal{X}_{\text{IMU}}$.

C. State-of-The-Art SLAM Algorithms

Our framework balances the need for computational efficiency, estimation accuracy, and map precision, tradeoffs observed in design choices of existing SLAM algorithms.

- *Extended Kalman Filter (EKF):* [5], [21], [22] –The EKF iteratively updates position estimates of the current pose and all observed features; all past poses are marginalized. This design favors computational speed over localization precision. A variant, the *iterated Extended Kalman Filter* (iEKF), takes multiple Gauss-Newton steps before marginalization to tune the linearization point. This improves mapping and localization accuracy but increases computation time.
- *Multi-State Constrained Kalman Filter:* [6], [7], [23]—The MSCKF iteratively updates a full state, with the current

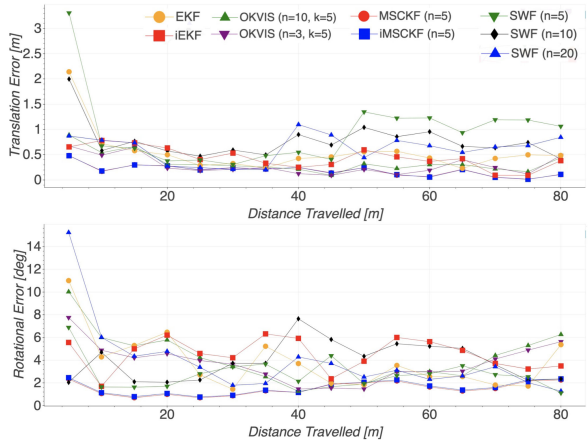

Fig. 1. Localization on Vicon Room 2 (medium). Drift from ground-truth is plotted against the distance traveled along the ground-truth trajectory, sampled at 5 m intervals. (Note: MSCKF and iMSCKF curves almost overlap). We apply trajectory alignment as in [29].

IMU state and n past poses; here $n \leq N_{\max}$, a specified upper bound that trades off accuracy and computational speed.Features are stored separately.

- *Sliding Window Smoother, Fixed-Lag Smoother:* [10], [24], [25]—The fixed-lag smoother resembles the MSCKF, but performs multiple steps of Gauss-Newton descent before the marginalization step, to tune the linearization point.This improves mapping and localization accuracies at the cost of increasing computation time.
- *Open Keyframe Visual-Inertial SLAM (OKVIS):* [12]—OKVIS updates a sliding window of "keyframes", poses deemed most informative,which may be arbitrarily spaced in time.Keyframe poses leaving the sliding window, and associated landmarks, are marginalized. This design aims to improve estimation accuracy by maximizing information encoded by the stored poses, without increasing computation time.
- *GraphSLAM and Bundle Adjustment:* [21], [26] –These algorithms solve the full SLAM problem, with no marginalization. Their state estimation can be more accurate than the above algorithms, but also far slower.

VI. Experiments

This section describes the empirical performance of different marginalization schemes on pose tracking of real-world data. We examine the MSCKF [6], a standard sliding window filter, and the keyframe-based OKVIS algorithm [12], each implemented as an incremental optimization algorithm of the form presented in this letter.

A. Simulation Settings

Experiments are performed on the EuRoC MAV dataset of stereo image sequences and IMU data [27]. We standardize the front-end across all experiments and implementations, using BRISK keypoint features with brute-force matching. Outlier rejection between the two cameras in the stereo setup is performed using a simple epipolar constraint test, and outlier rejection between stereo frames taken at subsequent timesteps is performed with reprojection distance test using the latest estimate of the feature position and camera pose. We use GTSAM in C++ in the back-end to construct and update costs, compute Jacobians, and implement Gauss-Newton and marginalization steps [8], [27]. To construct dynamics and measurement maps, we collect on-board IMU odometry measurements, and apply the IMU pre-integration scheme in [28]. (Appendix D). We apply trajectory alignment as in [29].

B. Results and Discussion

Localization root-mean-squared error on Vicon Room and Machine Hall sequences from the Euroc MAV dataset are presented in Table I. Due to space constraints, only the estimator drift on the V2_02 sequence is plotted (Fig 1). First, we analyze standard sliding window filters of window size $n = 5, 10, 20$ frames. Features are marginalized when they are only visible in the oldest frame in the optimization window. EKF and iEKF are also included, and are implemented as sliding window filters with window size 1. For the former, only 1 Gauss-Newton step is taken, and in the latter, steps are taken until convergence. Next, we implement MSCKF as an incremental optimization algorithm (Section V-B), with window size $n = 5$, and with two optimization schemes: (1) the standard formulation, with one Gauss-Newton step after marginalization, and (2) a version that takes multiple Gauss-Newton steps until convergence ("Iterated MSCKF," or iMSCKF). Finally, we implement OKVIS with IMU window size $n = 3, 10$, keyframe window size $k = 5$, and marginalization and keyframe selection schemes identical to those in Leutenegger *et al.* [12].

Our experiments show that, overall, OKVIS outperforms baseline sliding window filters, even when the latter has a larger window size. Moreover, our MSCKF implementation outperforms sliding window filters and OKVIS, even under challenging camera motions, despite the latter maintaining nonlinear constraints between camera poses and landmarks, and taking multiple Gauss-Newton steps per iteration. This persists even for sliding window filters with larger window sizes. Taking multiple Gauss-Newton steps in the iMSCKF estimator did not noticeably improve performance over the standard MSCKF, as illustrated in Fig. 1.

In contrast with sliding window filter and OKVis implementations of comparable sizes, the MSCKF recovers better from localization errors, by employing a marginalization scheme that always maintains poses arbitrarily far in the past. This is because older poses represent higher baselines and thus supply better localization information [7]. For instance, the MSCKF maintains the first pose in the estimator for a long time, rendering subsequent estimates more consistent with the initial pose, and thus minimizing drift at the start of the trajectory. In contrast, although OKVIS allows keyframes to be maintained arbitrarily far in the past, keyframes are usually roughly evenly spaced and form a sliding temporal window in camera motions. Thus, earlier poses are quickly marginalized, causing estimates to drift more at the start of the trajectory. Furthermore, the MSCKF includes features in the optimization window only after they

TABLE I
ROOT-MEAN-SQUARED ERROR IN TRANSLATION AND ROTATION ON VICON ROOM AND MACHINE HALL SEQUENCES FROM THE EUROC MAV DATASET. WE APPLY TRAJECTORY ALIGNMENT AS IN [30]

Data	MSCKF(n=5)	iMSCKF(n=5)	OKVIS(n=10,k=5)	OKVIS(n=3,k=5)	SWF(n=10)	SWF(n=20)	SWF(n=5)	EKF	iEKF
V1_01	0.09m, 2.87°	0.09m, 2.87°	0.20m, 3.62°	0.23m, 3.68°	0.36m, 3.12°	0.36m, 4.72°	1.00m, 8.22°	1.10m, 8.79°	0.04m, 59.10°
V1_02	0.15m, 1.52°	0.16m, 1.55°	-	-	0.33m, 3.90°	1.16m, 4.69°	0.30m, 3.68°	0.75m, 8.13°	0.42m, 5.48°
V1_03	1.00m, 4.84°	1.12m, 4.90°	0.42m, 6.49°	0.46m, 6.99°	-	6.36m, 25.85°	0.79m, 6.47°	14.84m, 24.58°	1.83m, 8.69°
V2_01	0.14m, 0.83°	0.14m, 0.80°	0.45m, 2.88°	0.39m, 2.58°	0.75m, 5.23°	0.23m, 1.71°	0.26m, 1.96°	0.79m, 3.67°	0.48m, 3.58°
V2_02	0.24m, 1.67°	0.24m, 1.74°	0.44m, 4.69°	0.38m, 4.13°	0.96m, 3.71°	0.75m, 4.49°	1.42m, 2.99°	0.83m, 4.32°	0.47m, 4.14°
MH01	0.07m, 1.03°	0.07m, 1.03°	0.63m, 8.44°	0.72m, 11.61°	0.19m, 1.31°	0.11m, 1.14°	0.43m, 4.02°	0.46m, 3.09°	0.42m, 3.11°
MH02	0.14m, 1.05°	0.19m, 1.86°	0.76m, 8.79°	0.93m, 9.99°	0.23m, 1.89°	0.23m, 1.58°	0.28m, 2.80°	0.34m, 2.21°	0.50m, 2.68°
MH03	0.26m, 1.36°	0.25m, 1.35°	0.64m, 4.29°	0.86m, 5.79°	0.32m, 1.77°	0.54m, 1.75°	0.25m, 1.60°	1.19m, 4.62°	1.33m, 5.23°
MH04	1.11m, 1.62°	1.06m, 1.52°	-	-	0.79m, 1.75°	0.86m, 1.79°	0.59m, 1.28°	4.10m, 4.58°	5.21m, 6.32°

have matured, and thus maximally utilizes localization information with fewer updates. Finally, incorporating only matured features ensures that each feature is always initialized through multiple-view triangulation instead of merely stereo triangulation. This minimizes the linearization error when features are marginalized.

VII. CONCLUSION

This letter presents a framework formulating and analyzing optimization and filtering-based SLAM approaches as the iterative application of key algorithm submodules, and proves that it encompasses state-of-the-art filtering algorithms as special cases. Experimental analysis indicate our formulation is useful for analyzing various design choices inherent in these existing SLAM algorithms, and implementing them in a modular fashion for a wide range of robotics applications, which we are eager to test on hardware.

As future work, we wish to apply our analysis to the *dynamic SLAM* problem, which concerns highly mobile features [2], [30] in practical multi-agent interactions, e.g., real-life traffic scenarios [31], by designing marginalization strategies for estimators that jointly track moving and stationary landmarks. We expect good performance on the dynamic SLAM problem, since it enables flexible user-selected design choices.

APPENDIX

Please use the following link to access an ArXiV version with the appendix (https://arxiv.org/pdf/2112.05921.pdf). The authors will ensure that this link stays active.

REFERENCES

[1] J. Leonard and H. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1991, vol. 3, pp. 1442–1447.

[2] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[3] A. J. Davison, "FutureMapping: The computational structure of spatial AI systems," 2018, *arXiv:1803.11288(cs.AI)*.

[4] A. J. Davison and J. Ortiz, "FutureMapping 2: Gaussian belief propagation for spatial AI," 2019, *arXiv:1910.14139(cs.AI)*..

[5] J. Solà, "Simultaneous localization and mapping with the extended Kalman filter," 2014, *arXiv:1803.11288*.

[6] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2007, pp. 3565–3572.

[7] M. Li and A. I. Mourikis, "Improving the accuracy of EKF-based visual-inertial odometry," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 828–835.

[8] F. Dellaert, "Factor graphs and GTSAM: A hands-on introduction," Tech. Rep. GT-RIM-CP&R-2012-002, 2012.

[9] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Trans. Robot.*, vol. 24, no. 6, pp. 1365–1378, Dec. 2008.

[10] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *Int. J. Robot. Res.*, vol. 25, no. 12, pp. 1181–1203, 2006.

[11] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the bayes tree," *Int. J. Robot. Res.*, vol. 31, pp. 217–236, Feb. 2012.

[12] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, pp. 314–334, 2015.

[13] K. Eckenhoff, P. Geneva, and G. Huang, "Closed-form preintegration methods for graph-based visual-inertial navigation," *Int. J. Robot. Res.*, vol. 38, pp. 563–586, 2019.

[14] D. Scaramuzza and F. Fraundorfer, "Visual odometry, Part I: The first 30 years and fundamentals," *IEEE Robot. Automat. Mag.*, vol. 18, no. 4, pp. 80–92, Dec. 2011.

[15] F. Fraundorfer and D. Scaramuzza, "Visual odometry, Part II: Matching, robustness, optimization, and applications," *IEEE Robot. Automat. Mag.*, vol. 19, no. 2, pp. 78–90, Jun. 2012.

[16] U. Frese, R. Wagner, and T. Röfer, "A SLAM overview from a user's perspective," *Künstliche Intelligenz*, vol. 24, pp. 191–198, 2010.

[17] S. Huang and G. Dissanayake, "A critique of current developments in simultaneous localization and mapping," *Int. J. Adv. Robot. Syst.*, vol. 13, no. 5, 2016, Art. no. 1729881416669482.

[18] K. Khosoussi, S. Huang, and G. Dissanayake, "A sparse separable SLAM back-end," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1536–1549, Dec. 2016.

[19] H. Strasdat, J. M. M. Montiel, and A. J. Davison, "Visual SLAM: Why filter?," *Image Vis. Comput.*, vol. 30, no. 2, pp. 65–77, 2012.

[20] B. Bell, "The iterated Kalman smoother as a Gauss-Newton method," *SIAM J. Optim.*, vol. 4, pp. 626–636, 1994.

[21] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics.* Cambridge, MA, USA: MIT Press, 2005.

[22] "Parameter estimation techniques: A tutorial with application to conic fitting," *Image Vis. Comput.*, vol. 15, no. 1, pp. 59–76, 1997.

[23] M. Li and A. I. Mourikis, "Optimization-based estimator design for vision-aided inertial navigation: Supplemental materials," in *Robotics: Science and Systems*, Cambridge, MA, USA: MIT Press, 2012, pp. 241–248.

[24] P. S. Maybeck, *Stochastics Models, Estimation, and Control: Introduction*. Cambridge, MA, USA: Academic Press, 1979.

[25] G. Sibley, L. H. Matthies, and G. S. Sukhatme, "Sliding window filter with application to planetary landing," *J. Field Robot.*, vol. 27, no. 5, pp. 587–608, 2010.

[26] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intell. Trans. Syst. Mag.*, vol. 2, no. 4, pp. 31–43, Winter 2010.

[27] F. Dellaert, "Factor graphs for robot perception," *Foundations Trends Robot.*, vol. 6, no. 1/2, pp. 1–139, 2017.

[28] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," *Robot.: Sci. Syst.*, 2015.

[29] S. Umeyama, "Least-squares estimation of transformation parameters between two point patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, pp. 376–380, Apr. 1991, doi: 10.1109/34.88573.

[30] J. Zhang, M. Henein, R. Mahony, and V. Ila, "VDO-SLAM: A Visual dynamic object-aware SLAM system," 2020, *arXiv:2005.11052[cs.RO]*.

[31] D. Fridovich-Keil, E. Ratner, A. Dragan, and C. Tomlin, "Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 1475–1481, doi: 10.1109/ICRA40945.2020.9197129.

[32] J. Solà, "Quaternion kinematics for the error-state KF," 2014, *arXiv:1711.02508*.