Is Iteration Worth It? Revisit Its Impact in Sliding-Window VIO

Chuchu Chen, Yuxiang Peng, and Guoquan Huang

***Abstract*— Visual-inertial odometry (VIO), which fuses noisy inertial readings and camera measurements to provide 3D motion tracking, is a foundational component in many autonomous applications. With the increasing use of next-generation edge devices (e.g., AR/VR devices, nano drones, and mobile robotics) that are constrained by limited power, resources, and multi-tasking demands, balancing computational efficiency and accuracy in VIO estimators has become more critical than ever. Historically, state estimation algorithms have been developed using either optimization or filtering-based methods, with the key distinction being the ability to relinearize measurements and correct state estimates iteratively. It has been widely claimed that iterative methods improve accuracy by allowing for the reduction of error through relinearization at a higher computational demand. Conversely, filtering methods are more efficient but may suffer from significant linearization errors. However, these trade-offs have not been thoroughly examined in the context of visual-inertial motion tracking. In this paper, we conduct the first comprehensive study on the impact of iterative algorithms in sliding-window VIO. We analyze the relinearization of IMU and camera measurements separately, providing insights into how each affects system performance. By considering key factors such as system observability and measurement processes, we offer a deeper understanding of VIO estimator behavior. Our findings, backed by real-world tests, offer practical guidelines for balancing accuracy and efficiency, helping practitioners determine when to prioritize iterative methods or simpler filtering approaches while encouraging researchers and engineers to rethink VIO design for optimal resource allocation.**

I. Introduction

Visual-inertial navigation systems (VINS) combine inertial readings and camera data to track 3D motion, forming a critical backbone for autonomous systems and next-generation devices [1]–[5]. Historically, VINS estimator development has been driven by two main approaches: optimization and filtering-based methods. The primary distinction between these approaches is the ability to relinearize nonlinear measurements to correct the state *iteratively*. There are also partial-iteration approaches, such as iterative Kalman filters, relinearize only a subset of the measurements. It is widely claimed that iterative methods offer better accuracy by reducing errors through relinearization at the cost of higher computational demands. Filter-based methods are more efficient but may have large linearization errors.

Unfortunately, these trade-offs have not been thoroughly evaluated. For next-generation edge devices—operating under strict constraints on processing power, memory, and energy resources while managing multiple tasks simultaneously—it is essential to determine when iterative methods are necessary and when simpler filtering approaches suffice to optimize VIO algorithms for robust performance without overloading the system's computational capacity. In this work, we aim to bridge this gap. We focus on the VIO problem, where estimation is conducted over a fixed-size sliding window of recent poses, by marginalizing past states and measurements—without incorporating long-term loop closures and global maps. We will explore the impact of re-linearizing IMU and visual measurements on estimation performance while considering the observability properties of the estimator. Our work encourages the community to rethink VIO estimator design by emphasizing the balance between accuracy and computational efficiency, providing insights, and offering practical guidance for resource-constrained environments. In summary, our main contribution includes:

- We present the first comprehensive study on the impact of iterative algorithms in sliding-window VIO, analyzing IMU and camera measurement relinearization to demonstrate how each influences system performance, supported by proof-of-concept real-world tests.
- Our study incorporates key factors such as system observability and measurement processes, providing a deeper understanding of VIO estimator behavior.
- We provide practical guidelines and valuable insights on when to prioritize accuracy over computational efficiency in the application of iterative algorithms.

II. Related Work

Inertial navigation systems (INS) have long been pivotal in estimating 6DOF poses in GPS-denied environments using low-cost, lightweight IMUs. While it provides high-accuracy localization, they are prone to noise and bias, necessitating the integration of cameras to provide visual information. One of the earliest and most successful filter-based VINS algorithms is the Multi-State Constrained Kalman Filter (MSCKF) [6]. Instead of adding a large number of detected features directly to the state vector, MSCKF projects visual bearing measurements onto the null space of the feature Jacobian matrix, preserving only the motion constraints relevant to the cloned camera poses [7], reduces computational complexity while maintaining essential information. It has also been extended to the iterative algorithm, which allows for the relinearization of camera measurements [8]–[11]. Filter-based VINS estimators often suffer from inconsistency due to spurious information gain in unobservable directions caused by linearization. To address this issue, several "observability-aware" approaches have been proposed [12]–[15], with the First-Estimates Jacobian (FEJ) technique [16]–

This work was partially supported by the University of Delaware (UD) College of Engineering, the NSF (SCH-2014264, IIS-2410019), Google ARCore, and Meta Reality Labs.

The authors are with the Department of Mechanical Engineering, University of Delaware, Newark, DE 19716, USA. Email: {ccchu,yxpeng,ghuang}@udel.edu

[18] gaining popularity for its simplicity and significant performance improvements. The full optimization method, which formulates a nonlinear least-squares (NLS) problem by using all available measurements and relinearizing them to estimate the entire state history, is generally expected to achieve the highest accuracy. However, the computational burden grows high as the trajectory and the map grow over time. Especially when incorporating high-rate IMU readings into the optimization framework, it requires the re-integration of high-rate readings between consecutive frames in the local window and highly increases the computational demand. Therefore, the pre-integration theory is introduced to formulate the relative motion constraints between frames and avoid repeated integration to save computation [19]–[23]. Another widely used technique is state marginalization [23]–[26], which reduces the state size while preserving key information. This is done by selectively marginalizing and permanently fixed certain states and computing a corresponding prior for the remaining states. Similar to filter-based systems, marginalization introduces inconsistencies and leads to the erroneous belief that it has gained information in unmeasurable directions, resulting in overconfident estimates. The FEJ method has also been applied to mitigate these issues and has demonstrated improved performance [27], [28]. Strasdat et al. [29] analyzed the trade-offs between filtering and BA but focused solely on visual SLAM. However, the combination of inertial and camera sensing can lead to significantly different performance outcomes, as IMU data provides essential metric and dynamic motion information, offering more accurate initial guesses for the NLS problem. More recently, [30] introduced a theoretical framework that clarifies the connections and distinctions among various estimators. To the best of the authors' knowledge, no comprehensive study has thoroughly compared and investigated the differences and relationships between filter-based and iterative algorithms across varying contexts.

III. Problem Statement

We formulate the estimation problem over the entire trajectory until the current time t_k with batch least squares. The system state consists of the current navigation states, $\mathbf{x}_k$, and 3D features, $\mathbf{x}_f$:

$$\mathbf{x} = \begin{bmatrix}\mathbf{x}_0^\top & \dots & \mathbf{x}_k^\top & \mathbf{x}_f^\top\end{bmatrix}^\top, \mathbf{x}_f = \begin{bmatrix}\dots & \mathbf{f}_g^\top & \dots\end{bmatrix}^\top$$
$$\mathbf{x}_k = \begin{bmatrix}{}^{I_k}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_k}^\top & {}^{G}\mathbf{v}_{I_k}^\top & \mathbf{b}_g^\top & \mathbf{b}_a^\top\end{bmatrix}^\top \tag{1}$$

where ${}^{I}_{G}\bar{q}$ is the unit quaternion that represents the rotation ${}^{I}_{G}\mathbf{R}$ from the global frame $\{G\}$ to the IMU frame $\{I\}$; ${}^{G}\mathbf{p}_I$ and ${}^{G}\mathbf{v}_I$ are the IMU position and velocity in $\{G\}$, $\mathbf{b}_g$ and $\mathbf{b}_a$ are the gyroscope and accelerometer biases, and the feature state, $\mathbf{x}_f$, comprises the global position of landmarks.

At timestep t_k, the *batch maximum a posteriori* (MAP) seeks to solve for the history of the state estimate $\mathbf{x}_{0:k}$ by maximizing the posterior PDF leveraging: 1) prior information $\mathcal{N}(\hat{\mathbf{x}}_0, \mathbf{P}_0)$, 2) IMU motion constraints $\mathbf{u}$, and 3) camera observation measurements $\mathbf{z}$:

$$p \propto p(\mathbf{x}_0)\prod_{i=0}^{k-1} p(\mathbf{x}_{i+1}|\mathbf{x}_i, \mathbf{u}_i)\prod_{\mathbf{z}_{i,j}\in\mathcal{Z}_{0:k}} p(\mathbf{z}_{i,j}|\mathbf{x}_i, \mathbf{f}_j) \tag{2}$$

where the set $\mathcal{Z}_{0:k}$ denotes all measurements between $[t_0, t_k]$. Under the Gaussian distribution assumption, maximizing the above PDF is equivalent to minimizing:

$$\mathcal{C}(\mathbf{x}) = \mathcal{C}_{p_0} + \sum_{i=0}^{k-1}\mathcal{C}_{I_i} + \sum_{\mathbf{z}_{i,j}\in\mathcal{Z}_{0:k}}\mathcal{C}_{f_{ij}} \tag{3}$$

where we define the following nonlinear cost terms:

$$\text{Prior:}\quad \mathcal{C}_{p_0} = \frac{1}{2}||\mathbf{x}_0 \boxminus \hat{\mathbf{x}}_0||^2_{\mathbf{P}_0} \tag{4}$$
$$\text{Inertial:}\quad \mathcal{C}_{I_i} = \frac{1}{2}||\mathbf{x}_{i+1} \boxminus \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)||^2_{\mathbf{W}_i} \tag{5}$$
$$\text{Camera:}\quad \mathcal{C}_{f_{ij}} = \frac{1}{2}||\mathbf{z}_{ij} \boxminus \mathbf{h}(\mathbf{x}_i, \mathbf{f}_j)||^2_{\mathbf{R}_{ij}} \tag{6}$$

Note that the cost can be formulated in various ways. For instance, IMU readings can be expressed directly between two IMU states, or they can be formulated using preintegration as a relative motion constraint to avoid recomputation when the linearization point changes. The state and error states can also be represented with different formulations [15].

IV. Estimation Algorithms

The estimation problem is typically represented as a factor graph [see Figure 1], where state variables are depicted as circles (nodes) and measurements are represented as edges connecting related states. In the following sections, we will discuss the key aspects of the VIO estimator.

A. Optimization vs. Filtering-based Estimators

With a slight abuse of notation, let the current linearization point for this problem be denoted as $\hat{\mathbf{x}}^\ominus = [\hat{\mathbf{x}}_0^\ominus, \hat{\mathbf{x}}_1^\ominus, \hat{\mathbf{x}}_2^\ominus, \hat{\mathbf{f}}^\ominus]$.

1) Full-BA (BA) : The full optimization utilizes all measurements to formulate the nonlinear least squares problem with the cost we introduced in the previous section and solves iteratively. Specifically, given the l-th iteration with linearization point $\hat{\mathbf{x}}^l$, for each iteration we have the linearized cost:

$$\mathcal{C}_B(\delta\mathbf{x}) = \frac{1}{2}||\delta\mathbf{x}_0^l - \mathbf{r}_0||^2_{\mathbf{P}_0} + \frac{1}{2}\sum_{k=1}^{2}||\mathbf{\Phi}_k^l\delta\mathbf{x}^l - \mathbf{r}_{I_k}||^2_{\mathbf{W}_k} + \frac{1}{2}\sum_{m=1}^{3}||\mathbf{H}_m^l\delta\mathbf{x}^l + \mathbf{H}_{f_m}^l\delta\mathbf{f}^l - \mathbf{r}_m||^2_{\mathbf{R}_m} \tag{7}$$

where $\mathbf{r}_{I_k}$ and $\mathbf{r}_m$ are residuals for IMU and camera cost, $\mathbf{\Phi}_k^l$ denotes the Jaocbians for IMU cost. $\mathbf{H}_m^l$ and $\mathbf{H}_f^l$ are the Jacobians for camera measurements. Minimizing the above cost, $\delta\mathbf{x}^l$ is computed to update the state estimate:

$$\hat{\mathbf{x}}^{l+1} = \hat{\mathbf{x}}^l \boxplus \delta\mathbf{x}^l \tag{8}$$

The updated state $\hat{\mathbf{x}}^\oplus \leftarrow \hat{\mathbf{x}}^{l+1}$ will be set once converge or reach the maximum number of iterations at $l+1$'th iteration.

2) Iterative Filtering (IF): In the iterative filter-based method (e.g., an iterative Kalman filter), the state is propagated using inertial readings, with iterative updates performed to relinearize the camera measurements. This process can be visualized in a simplified graph, as shown in Figure 1, middle. Specifically, the nonlinear IMU cost is linearized at the state estimate $\hat{\mathbf{x}}^\ominus$, creating a new prior factor ${}^{I}\mathbf{p}_I$ as:

$$\mathcal{C}_{p_I}(\delta\mathbf{x}) = \frac{1}{2}||\delta\mathbf{x}_0 - \mathbf{r}_0||^2_{\mathbf{P}_0} + \frac{1}{2}\sum_{k=1}^{2}||\mathbf{\Phi}_k^\ominus\delta\mathbf{x} - \mathbf{r}_{I_k}||^2_{\mathbf{W}_k}$$

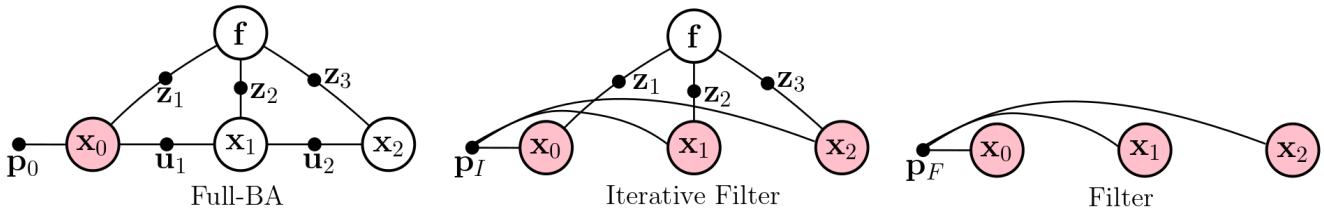

Fig. 1: Example graph for full-BA optimizer (left), iterative filter-based estimator (middle) and filter-based estimator (right).

where $\mathbf{\Phi}_k^{\ominus}$ denotes the IMU measurement Jacobian linearized with initial state $\hat{\mathbf{x}}^{\ominus}$. With this linearized prior factor (cost), and the nonlinear camera measurements, the new cost given the l-th iteration with linearization point $\hat{\mathbf{x}}^l$ is:

$$\mathcal{C}_{IF}(\delta\mathbf{x}^l) = \mathcal{C}_{p_I}(\delta\mathbf{x}^l) + \frac{1}{2}\sum_{m=1}^{3} ||\mathbf{H}_m^l \delta\mathbf{x}^l + \mathbf{H}_{f_m}^l \delta\mathbf{f}^l - \mathbf{r}_m||^2_{\mathbf{R}_m}$$

Similarly, minimizing the above cost, $\delta\mathbf{x}^l$ is computed to update for the new state estimate with Eq. (8).

3) Filtering (F): Finally, filter-based methods (Figure 1, right) linearize all measurements only once and end with a new prior connect to all the states:

$$\mathcal{C}_F(\delta\mathbf{x}) = \frac{1}{2}||\delta\mathbf{x}_0 - \mathbf{r}_0||^2_{\mathbf{P}_0} + \frac{1}{2}\sum_{m=1}^{3} ||\mathbf{H}_m^{\ominus}\delta\mathbf{x} - \mathbf{r}_m^*||^2_{\mathbf{R}_m} + \frac{1}{2}\sum_{k=1}^{2} ||\mathbf{\Phi}_k^{\ominus}\delta\mathbf{x} - \mathbf{r}_{I_k}||^2_{\mathbf{W}_k} \quad (9)$$

where $\mathbf{H}_m^{\ominus}$ and $\mathbf{r}_m^*$ are obtained by nullspace projection [6]. With all Jacobians computed with respect to $\mathbf{x}^{\ominus}$, the new state is computed from a single update:

$$\mathbf{x}^{\oplus} = \hat{\mathbf{x}}^{\ominus} + \delta\mathbf{x} \quad (10)$$

Specifically, only the linearization point—typically obtained through feature triangulation—is used to compute the measurement Jacobian, while the feature mean is not updated.

B. Marginalization, Consistency, and FEJ

Different strategies are used to manage state and feature size to enable real-time VIO [27], [31]. In this work, we focus on the sliding-window method, which keeps recent IMU states while removing older ones. We adopt the common approach of marginalizing a feature when its connected IMU state is removed from the sliding window [27], similar to the MSCKF feature in filter-based methods. For example, in Figure 2, assume the sliding window size is 3, when $\mathbf{x}_3$ is added, $\mathbf{x}_0$ is to be marginalized, the associated feature $\mathbf{f}_1$ is also removed. This approach is widely used in both filter-based and optimization frameworks.

VINS have been shown to suffer from inconsistency issues related to system observability, linearization, and state

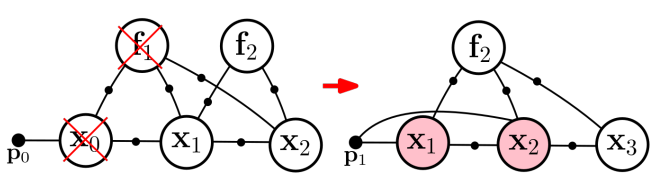

Fig. 2: Example of the marginalization and FEJ process, pink nodes indicating where linearization points are fixed.

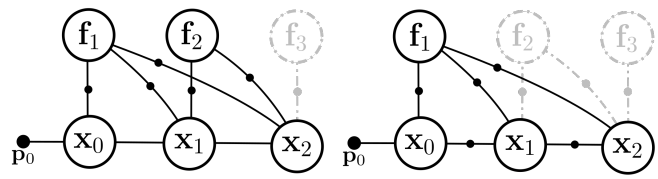

Fig. 3: Example graph illustrating the sequential (left) and batch (right) methods, with a sliding window size of 3.

marginalization in both optimization and filter-based estimators [27]. These can be addressed using the First-Estimates Jacobian (FEJ) method, which fixes the state linearization point during Jacobian evaluation. In short, states connected to a prior factor must have their linearization points fixed during Jacobian evaluation. For instance, in Figure 2, when $\mathbf{x}_0$ and $\mathbf{f}_0$ are marginalized, the new prior factor $\mathbf{p}_1$ connects to $\mathbf{x}_1$ and $\mathbf{x}_2$, thus these two states are being fixed, shown in pink in the Figure. More details refer to our previous works [3]. In comparison with full BA, iterative filters, and filter-based methods, FEJ states are also represented as pink nodes in Figure 1. In full BA, only the states connected to the prior need to be fixed. In iterative filters, since IMU measurements are used to propagate the state only once and are not relinearized, the corresponding IMU states are fixed during the iterative update process. Simiarly, in the filter, as features are used only once and never updated, their triangulation values are treated as FEJ values. The propagated IMU states, rather than the updated one, will be used to compute Jacobian [18]. It is important to clarify that the states are fixed only when evaluating the Jacobian; the state corrections (i.e., $\delta\mathbf{x}$) will still be computed and the state estimates will be updated accordingly.

C. Batch vs. Sequential Measurement Processing

Even though the states remain the same, there are two common methods for processing measurements. The first, known as the sequential method, corrects the states as soon as measurements become available. In contrast, the batch method waits until the measurements span the entire sliding window before using them for state estimation. Figure 3 illustrates this process with the sliding window of size 3. Specifically, the sequential method (left) will use all measurements for $\mathbf{f}_1$ and $\mathbf{f}_2$ to perform state estimate but ignores those for $\mathbf{f}_3$ because there is not enough data to triangulate its 3D position. In the batch method (right), the measurements for $\mathbf{f}_2$ are not processed yet, as they do not span the oldest pose (to be marginalized pose), which will not incur information loss if not processed immediately. However, the measurements for $\mathbf{f}_1$ have measurement at the to-be-marginalized pose; if not processed, this measurement will

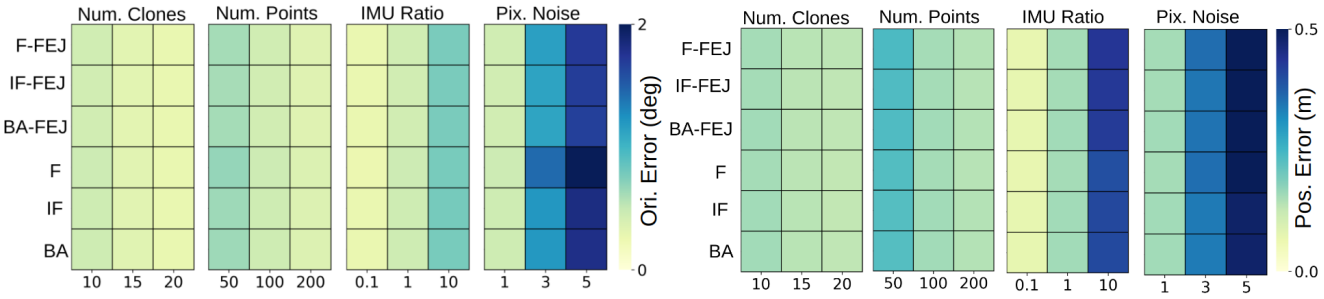

Fig. 4: Orientation (left) and position error (right) for different estimators and parameters. The darker the color the error is larger. Darker colors indicate larger errors. This figure illustrates the error trend in estimation performance; detailed results can be found in [32].

TABLE I: Estimator configurations

Relinearize	Full-BA (**BA**)	Iterative Filter (**IF**)	Filter (**F**)
IMU mea.	✓	✗	✗
CAM mea.	✓	✓	✗

be dropped. Note that this distinction refers only to which measurements are used, and different estimators (e.g., filters or BA) can still be employed to solve the problem. Sequential updates allow more updates at the cost of extra computation, which might be able to minimize the chance when there is no feature to update. Batch update has the advantage of getting enough parallax for good feature triangulation and better efficiency, but it might encounter issues when the estimator runs out of features for update.

V. Experiments

We develop an iterative estimation system to support filter (F), iterative filter (IF), and full optimization (BA) approaches, all with and without the FEJ method applied. Specifically, we begin by modifying the estimator from an Extended Kalman Filter (EKF) to a square-root inverse filter (SRIF). Inspired by [30], we perform QR-based marginalization as state information (SI) update to maintain upper-triangular square-root marginalized prior, and all the other measurements are processed as state-only (SO) update with a QR-based decomposition together with square-root prior for state update. During SO update, the measurements can be relinearized, and the state can be updated iteratively, which is equivalent to the optimization-based approach. To efficiently incorporate IMU readings, we utilize CPI [21] to compute Jacobians for preintegration measurements. We also applied the Levenberg-Marquardt approach and Huber loss in the iterative solver to ensure a robust nonlinearization optimizaiton. Table I summarize the key different for each of them for the convenience of the reader. We leverage OpenVINS [33] to simulate a realistic indoor dataset with visual bearings and inertial measurements, with details available in the supplementary material [32] and report the Absolute Trajectory Error (ATE) for each method. We maintain a sliding window of IMU states, marginalizing the oldest state at each time step. Features are marginalized when their associated IMU states are removed from the window as described in Section IV-B. All the reported results are based on 50 Monte-Carlo runs.

A. Parameter Sensitivity Analysis in Standard Cases

We first evaluate the performance of each algorithm under standard conditions, varying key parameters. Specifically, we test different sliding window sizes from 10 to 20, and the number of feature points from 50 to 200. Additionally, we adjust IMU noise levels by applying different scaling factors (i.e., IMU ratios) to either inflate or deflate the original noise. Lastly, we test with different camera pixel noise levels, ranging from 1 to 5 pixels. The results are presented in Figure 4, where the estimation errors for both orientation and position are color-coded for each algorithm. Due to the space constraints, more detailed results can be found in [32].

From these results, several key observations can be made. First, as the number of clones and feature points increases, estimation accuracy improves due to the added constraints within each window. We then compare the performance of iterative and non-iterative algorithms. Interestingly, in most cases, the difference between iterative and non-iterative methods is minimal. When IMU and camera noise levels are unrealistically large (e.g., 5 pixels), iterative methods begin to show improvement though the noise conditions in these scenarios are unlikely to occur in practical applications. We then examine the performance of the FEJ. In most cases, their performance is similar, likely due to the minimal difference between the first and current state estimates, as there are no long-tracked SLAM features. This aligns with the findings from our previous work [27].

B. Challenging but Practical Cases

We now explore several challenging real-world scenarios to assess the performance of each algorithm and analyze the impact of iteration. In the following section, different colors are used to represent FEJ (pink) and no-FEJ (blue) estimators, with darker shades indicating more measurements being relinearized (e.g., dark blue for full-BA and light blue for filter), an example can be found in Figure 5.

1) Lose Feature Tracks: In real-world mobile robotics, losing camera features is common in challenging conditions like featureless or dynamic environments, high motion blur, temporary obstructions, or sudden lighting changes during indoor-outdoor transitions. Fortunately, IMU readings can help to prevent the system from immediate failure. To demonstrate this, we simulate cases where the system relies solely on IMU data for a specific duration. The results are

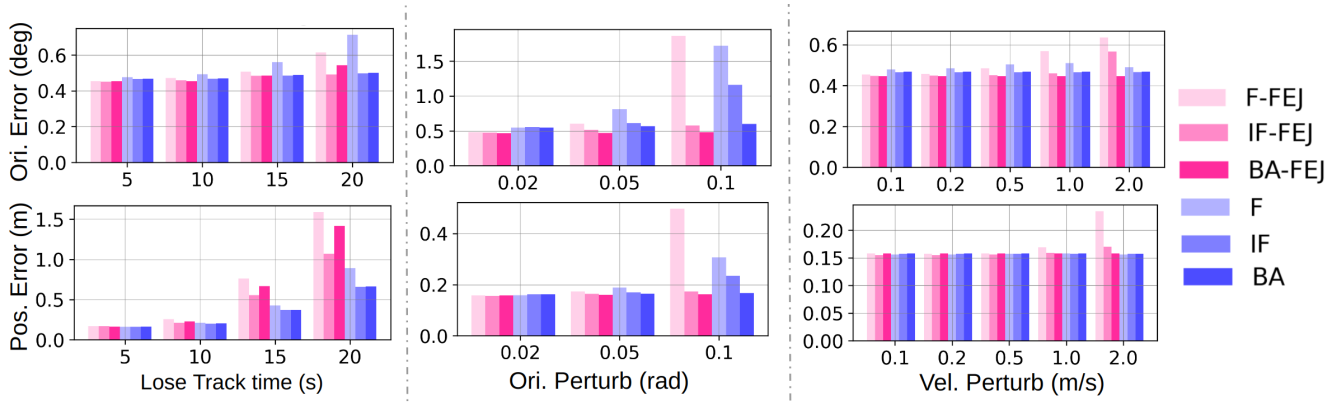

Fig. 5: Orientation (top) and position (bottom) estimation errors for various estimators. The left figure displays performance across different time periods of feature track loss, while the middle and right figures show performance with varying initial orientation and velocity perturbations.

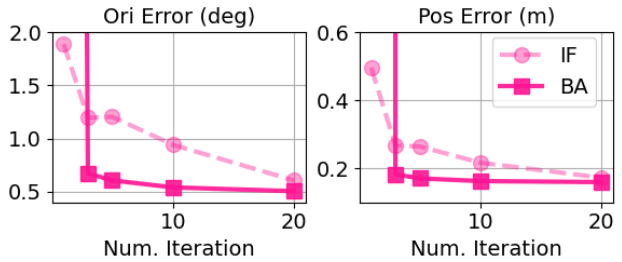

Fig. 6: Converegence rate comparison for IF-FEJ and BA-FEJ when perturbing initial orientation with 0.1 rad.

TABLE II: Average ATE for various estimators across different levels of camera noise and feature processing methods.

Noise pix.	Method	Sequential (deg/m)	Batch (deg/m)
3	F-FEJ	1.477 / 0.469	1.152 / 0.357
	IF-FEJ	1.468 / 0.429	1.132 / 0.344
	BA-FEJ	1.420 / 0.454	1.122 / 0.348
	F	1.390 / 0.394	1.441 / 0.354
	IF	1.385 / 0.392	1.210 / 0.340
	BA	1.375 / 0.393	1.198 / 0.338

shown in Figure 5 (left), with the x-axis representing the time without feature updates. The results clearly demonstrate that iteration significantly enhances accuracy, especially when the camera cannot detect features for extended periods (e.g., 20 seconds). This is likely due to significant IMU drift during periods without camera tracking, where a single iteration is insufficient to correct the accumulated errors. Additionally, IF and BA show similar performance, indicating that in this context, relinearizing camera measurements has a more significant impact than relinearizing IMU data. In comparing FEJ and non-FEJ estimators, we observe that FEJ-based estimators perform worse than their non-FEJ counterparts. Similar to the previous example where IMU noise was high, the error from IMU pose dominates performance. This is further evidenced by the fact that BA-FEJ performs worse than IF-FEJ.

2) Inaccurate System Initialization: Successful operation of VINS typically requires good initial conditions. While extensive work has addressed the initialization problem, low-excitation scenarios can still pose challenges, leading to inaccurate initial conditions [5], [34]. We thus conduct simulations where the initial states are perturbed from the simulated ground truth and evaluate the performance of each algorithm. Due to space constraints, Figure 5, middle and right, show the estimation orientation and position error with perturbed initial orientation and initial velocity; more complete results are referred to [32]. Starting with the middle figure, which shows results for perturbed initial orientation, we observe that F-FEJ performs worse than F, likely due to the incorrect initial conditions. In contrast, IF-FEJ shows significant improvement over F-FEJ, indicating the benefit of iterative updates in this case. Finally, the BA methods, while computationally more demanding, show that BA-FEJ offers only limited improvement over IF-FEJ, consistent with our earlier observations that relinearizing inertial measurements does not yield substantial gains. Next, we examine the results of perturbing the initial velocity. Minor errors have little impact, but larger errors (1 m/s) show significant improvement with iteration. IF and BA also show minimal differences. To further clarify these findings, we report the convergence rates for FEJ-based estimators, as shown in Figure 6. While IF and BA eventually achieve similar error reductions, full-BA demonstrates a faster convergence rate. As seen, full-BA reduces errors quickly within just a few iterations, whereas IF requires more iterations to reach convergence.

C. Sequential vs. Batch Measurement Processing

Next, we evaluate different feature measurement processing methods as discussed in Section IV-C. Results are reported in Table II. Similar to earlier findings, in standard cases, iterative and non-iterative methods show comparable performance. Interestingly, the sequential method demonstrates worse performance, likely because it uses fewer measurements to triangulate features, leading to less accurate linearization points and degraded estimation performance. It is also worth noting that the sequential method imposes a higher computational burden than the batch method, as the

TABLE III: Average ATE in degrees/meters. VINS-Mono(1) indicates the estimator is modified to a single iteration.

Algo.	V101	V102	V103	V201	V202	V203	MH01	MH02	MH03	MH04	MH05
VINS-Mono	0.82 / 0.07	2.74 / 0.10	5.15 / 0.15	2.13 / 0.09	2.57 / 0.13	3.43 / 0.29	0.78 / 0.20	0.86 / 0.18	1.84 / 0.22	2.51 / 0.41	0.94 / 0.29
VINS-Mono(1)	0.84 / 0.07	2.78 / 0.10	4.95 / 0.17	1.92 / 0.08	2.49 / 0.13	3.40 / 0.30	0.76 / 0.21	0.99 / 0.19	1.82 / 0.22	2.00 / 0.42	0.91 / 0.28

same measurements are relinearized multiple times compared to BA. Additionally, when the FEJ method is used, its performance is highly dependent on the accuracy of the initial state estimates (i.e., feature triangulation). If these initial values are inaccurate, the performance of the FEJ estimator can degrade.

D. Real-world Test

As a proof-of-concept experiment, we employ VINS-Mono [23], a widely-used open-source optimization-based VINS system that leverages the Ceres Solver [35], to evaluate its performance on the EurocMAV dataset. We use the Levenberg-Marquardt solver with an initial trust region to $1e8$ for proper convergence and allow only 1 maximal iteration in the open-source codebase, which is easily reproducible. The results of the modified code and the original code are reported in Table III, showing a single iteration demonstrates remarkably strong performance compared to multiple iterations in this indoor and standard dataset, consistent with our findings in the simulation.

VI. Iteration or Not: A Discussion

It is important to highlight that our study is centered on a straightforward yet essential configuration—sliding-window VIO, a common approach for 3D motion tracking. We also acknowledge that testing every estimator under all potential conditions is unrealistic, especially given the unpredictable nature of real-world systems. Nonetheless, our experiments and findings are intended to offer meaningful insights, prompting the community to reconsider algorithm design on resource-constrained edge devices.

A. Filter May Be Enough for Most Common Cases

Based on our extensive analysis, we found that under practical and realistic conditions, iterative methods provide minimal improvement over standard filtering approaches. This is largely because, within the short sliding window used in VIO, both inertial and camera measurements offer only relative constraints. Without global information like GPS, maps, or loop closures, relinearizing nonlinear measurements does little to reduce errors effectively. In filtering-based VIO, high-frequency inertial readings first propagate IMU states, which are then treated as accurate to form a linear system for feature triangulation. Since the linear solution is nearly optimal, a single update is typically sufficient in most scenarios. In other words, the drift caused by the relative nature of VIO sensing is difficult to significantly reduce by performing iterative updates.

B. Good Initials are Important for Filter

In our analysis, we identified three key scenarios where filter-based estimators struggle, all tied to the same root: poor initial state estimates. In Section V-B, we show the case with a lost feature track and perturb the initial states; both cases will cause IMU states to be inaccurate for filter update. In these cases, incorrect initial states lead directly to estimation errors without iteration, as only one update is not sufficient for good convergence. Section V-B emphasizes the importance of accurate feature initialization in enhancing performance—even with fewer updates, accurate initial values consistently outperform less reliable ones. Moreover, when the FEJ method is used to ensure system consistency, the initial values (or FEJ values) become even more critical. If the initial values are highly inaccurate, the FEJ can actually degrade performance. Thus, we recommend focusing on the quality of initial values, which may be even more crucial than repeatedly relinearizing measurements.

C. Iteration Can Help!

Unfortunately, in real-world scenarios, systems often face challenges, and good initial conditions are not always guaranteed. In such cases, iteration has been shown to improve performance. For instance, while accurately initializing the system can be challenging, iterative updates when the system is just initalized can enhance robustness to poor initial conditions. Moreover, although relinearizing only partial measurements (e.g., camera measurements) effectively reduces error, enabling full-BA can speed up convergence.

VII. Conclusion and Future Work

In this work, we have presented a comprehensive study on the impact of iterative algorithms in sliding-window VIO, focusing on the relinearization of IMU and camera measurements. We compare the performance of full-BA, iterative filter, and filter algorithms in both standard simulation scenarios and challenging but practical examples, such as lost feature measurements during a time and bad initial conditions to evaluate the impact of iteration algorithms for different measurements. In our study, we also compare and discuss key factors such as system observability and measurement processes, we provided a deeper understanding of VIO estimator behavior, offering valuable insights to guide the design and optimization of VIO systems. Our real-world validation through open-sourced systems reinforces the practical relevance of these findings, helping practitioners make informed decisions about when to prioritize iterative methods or simpler filtering approaches. Ultimately, this work encourages a rethinking of VIO design for resource-constrained platforms, where balancing computational efficiency and accuracy is crucial for achieving robust performance. In the future, we will extend our analysis to visual SLAM with delayed measurements and loop closure. Additionally, we plan to incorporate SLAM features and explore methods for establishing safety checks to enable iterative refinements, aiming to balance accuracy and computational efficiency more effectively.

References

[1] G. Huang, "Visual-inertial navigation: A concise review," in *Proc. International Conference on Robotics and Automation*, Montreal, Canada, May 2019.

[2] Y. Peng, C. Chen, and G. Huang, "Ultrafast square-root filter-based VINS," in *Proc. International Conference on Robotics and Automation*, Yokohama, Japan, May 2024.

[3] C. Chen, Y. Yang, P. Geneva, W. Lee, and G. Huang, "Visual-inertial-aided online mav system identification," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Kyoto, Japan., 2022.

[4] C. Chen, P. Geneva, Y. Peng, W. Lee, and G. Huang, "Monocular visual-inertial odometry with planar regularities," in *Proc. of the IEEE International Conference on Robotics and Automation*, London, UK., 2023.

[5] N. Merrill, P. Geneva, S. Katragadda, C. Chen, and G. Huang., "Fast and robust learned single-view depth-aided monocular visual-inertial initialization," *International Journal of Robotics Research*, May 2024.

[6] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 3565–3572.

[7] S. Roumeliotis and J. Burdick, "Stochastic cloning: a generalized framework for processing relative state measurements," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 2, 2002, pp. 1788–1795 vol.2.

[8] D. G. Kottas and S. I. Roumeliotis, "An iterative kalman smoother for robust 3d localization on mobile and wearable devices," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 6336–6343.

[9] G. Huang, A. I. Mourikis, and S. I. Roumeliotis, "An observability constrained sliding window filter for SLAM," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, Sept. 2011, pp. 65–72.

[10] T.-C. Dong-Si and A. I. Mourikis, "Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 5655–5662.

[11] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[12] Z. Huai and G. Huang, "Robocentric visual-inertial odometry," *International Journal of Robotics Research*, Apr. 2019.

[13] A. Barrau and S. Bonnabel, "The invariant extended kalman filter as a stable observer," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1797–1812, 2016.

[14] Y. Yang, C. Chen, W. Lee, and G. P. Huang, "Decoupled right invariant error states for consistent visual-inertial navigation," *IEEE Robotics and Automation Letters*, 2022.

[15] C. Chen, Y. Peng, and G. Huang, "Visual-inertial state estimation with decoupled error and state representations," in *Proc. of International Workshop on the Algorithmic Foundations of Robotics*, Chicago, IL, 2024.

[16] G. Huang, A. I. Mourikis, and S. I. Roumeliotis, "A quadratic-complexity observability-constrained unscented Kalman filter for SLAM," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1226–1243, Oct. 2013.

[17] M. Li and A. I. Mourikis, "High-precision, consistent ekf-based visual-inertial odometry," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[18] C. Chen, Y. Yang, P. Geneva, and G. Huang, "FEJ2: A consistent visual-inertial state estimator design," in *International Conference on Robotics and Automation (ICRA)*, Philadelphia, USA, 2022.

[19] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 61–76, 2011.

[20] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration theory for fast and accurate visual-inertial navigation," *IEEE Transactions on Robotics*, pp. 1–18, 2015.

[21] K. Eckenhoff, P. Geneva, and G. Huang, "Closed-form preintegration methods for graph-based visual-inertial navigation," *International Journal of Robotics Research*, vol. 38, no. 5, pp. 563–586, 2019. [Online]. Available: https://github.com/rpng/cpi

[22] Y. Yang, B. P. W. Babu, C. Chen, G. Huang, and L. Ren, "Analytic combined imu integrator for visual-inertial navigation," in *Proc. of the IEEE International Conference on Robotics and Automation*, Paris, France, 2020.

[23] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[24] E. D. Nerurkar, K. J. Wu, and S. I. Roumeliotis, "C-KLAM: Constrained keyframe-based localization and mapping," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 3638–3643.

[25] H. Liu, M. Chen, G. Zhang, H. Bao, and Y. Bao, "ICE-BA: Incremental, consistent and efficient bundle adjustment for visual-inertial slam," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1974–1982.

[26] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[27] C. Chen, P. Geneva, Y. Peng, W. Lee, and G. Huang, "Optimization-based vins: Consistency, marginalization, and fej," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[28] C. Chen, Y. Peng, and G. Huang, "Fast and consistent covariance recovery for sliding-window optimization-based vins," in *Proc. International Conference on Robotics and Automation*, Yokohama, Japan, May 2024.

[29] H. Strasdat, J. M. Montiel, and A. J. Davison, "Visual slam: why filter?" *Image and Vision Computing*, vol. 30, no. 2, pp. 65–77, 2012.

[30] K. Wu, "On the efficiency and consistency of visual-inertial localization and mapping," Ph.D. dissertation, Department of Computer Science and Engineering, University of Minnesota, 2024. [Online]. Available: https://hdl.handle.net/11299/262002

[31] C. Chen, P. Geneva, Y. Peng, W. Lee, and G. Huang, "Technical report: Optimization-based VINS: Consistency, marginalization, and fej," University of Delaware, Tech. Rep. RPNG-2023-GRAPH, 2023. [Online]. Available: https://udel.edu/~ghuang/papers/tr_graph.pdf

[32] C. Chen, Y. Peng, and G. Huang, "Supplementary material: Is iteration worth it? revisit its impact in sliding-window vio," University of Delaware, Tech. Rep. RPNG-2024-COV, 2024. [Online]. Available: https://udel.edu/~ghuang/papers/tr_iterative.pdf

[33] P. Geneva, N. Merrill, Y. Yang, C. Chen, W. Lee, and G. Huang, "Versatile 3d multi-sensor fusion for lightweight 2d localization," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Las Vegas, NV, 2020.

[34] N. Merrill, P. Geneva, S. Katragadda, C. Chen, and G. Huang, "Fast monocular visual-inertial initialization leveraging learned single-view depth," in *Proc. Robotics: Science and Systems (RSS)*, Daegu, Republic of Korea, July 2023.

[35] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres Solver," https://github.com/ceres-solver/ceres-solver, 2022.