$\sqrt{\text{VINS}}$: Robust and Ultrafast Square-Root Filter-based 3D Motion Tracking

Yuxiang Peng, *Student Member, IEEE,* Chuchu Chen, *Member, IEEE,* Kejian Wu, *Member, IEEE,* and Guoquan Huang, *Senior Member, IEEE*

***Abstract*—In this paper, we develop and open-source, for the first time, a robust and efficient square-root filter (SRF)-based visual-inertial navigation system (VINS), termed $\sqrt{\text{VINS}}$, which is ultra-fast, numerically stable, and capable of dynamic initialization even under extreme conditions (i.e., extremely small time window). Despite recent advancements in VINS, resource constraints and numerical instability on embedded (robotic) systems with limited precision remain critical challenges. A square-root covariance-based filter offers a promising solution by providing numerical stability, efficient memory usage, and guaranteed positive semi-definiteness. However, canonical SRFs suffer from inefficiencies caused by disruptions in the triangular structure of the covariance matrix during updates. The proposed method significantly improves VINS efficiency with a novel Cholesky decomposition (LLT)-based SRF update, by fully exploiting the system structure and the SRF to preserve the upper triangular structure of square-root covariance. Moreover, we design a fast, robust, dynamic initialization method, which first quickly recovers the minimal states without triangulating 3D features and then efficiently performs *iterative* SRF update to refine the full states, enabling seamless VINS operation even in challenging scenarios. The proposed LLT-based SRF is extensively verified through numerical studies, demonstrating superior numerical stability under challenging conditions and achieving robust efficient performance on 32-bit single-precision floats, operating at *twice* the speed of state-of-the-art (SOTA) methods. Our initialization method, tested on both mobile workstations and Jetson Nano computers achieving a high success rate of initialization even within a $100ms$ window under minimal conditions. Finally, the proposed $\sqrt{\text{VINS}}$ is extensively validated across diverse scenarios, demonstrating strong efficiency, robustness, and reliability. The full open-source implementation is released to support future research and applications.**

$\sqrt{\text{VINS}}$**: https://github.com/rpng/sqrtVINS**

***Index Terms*—Visual-Inertial Systems, State Estimation, SLAM, VIO, Motion Tracking, Initialization, Sensor Fusion**

I. Introduction

Visual-inertial navigation systems (VINS), which utilize a (single) camera and an inertial measurement unit (IMU) for 3D motion tracking, hold significant potential and are widely used in autonomous robots and mobile devices (e.g., see [1]–[4]). Numerous VINS algorithms have been developed in recent years [5] and can be broadly categorized into covariance-based and information-based methods. The former, such as the multi-state constraint Kalman filter (MSCKF) [6], update estimates using a dense covariance matrix but can lose positive definiteness, causing instability and divergence. The latter, such as the sliding-window optimization [7], exploit the sparse structure of the information matrix for efficiency, which, however, may become ill-conditioned. While double-precision arithmetic could mitigate the numerical error, this problem is especially pronounced on embedded platforms, where single-precision floating-point arithmetic is only available on the computation unit or is required to speed up and achieve real-time performance.

The square-root filter (SRF), which tracks the square-root covariance matrix, offers promising benefits such as numerical stability, guaranteed positive-definiteness, and reduced memory requirements [8]. However, applying SRF to VINS is challenging due to inefficiencies in maintaining the triangular structure of the covariance matrix during updates, particularly with large measurement sizes. This limitation has prevented SRF from being widely used in VINS. In the recent work [9], a novel permuted-QR (P-QR) decomposition of SRF was proposed to efficiently utilize the upper-triangular structure during matrix factorization, which has enabled seamless integration of SRF into VINS. In this work, building upon [9], we develop an enhanced Cholesky decomposition (LLT)-based SRF update to further improve computational efficiency while maintaining numerical stability.

Another critical module for VINS is dynamic initialization, which plays a vital role in recovering the initial states *on the fly* to enable seamless and continuous operation. Existing methods typically construct a linear system using image features and inertial measurements to obtain a closed-form solution, followed by nonlinear optimization to refine the estimated states. Although the minimal case of dynamic initialization requires 3 frames and 2 features, corresponding to only $100ms$ with a 20Hz camera [10], existing methods often require over 1 second to achieve robust initialization [10]–[30]. In this work, we develop a fast initialization method that, for the first time, demonstrates the ability to robustly initialize the system, even in the minimal case.

In particular, the main contributions of this paper are summarized as follows:

- We propose square-root VINS ($\sqrt{\text{VINS}}$), achieving exceptional efficiency and remarkable numerical stability on 32-bit computing platforms, with performance more than twice as fast as state-of-the-art (SOTA) algorithms for 3D motion tracking.
- Our $\sqrt{\text{VINS}}$ introduces a novel Cholesky decomposition

This work was partially supported by the University of Delaware (UD) College of Engineering, Delaware NASA/EPSCoR Seed Grant, NSF (SCH-2014264, MRI-2018905, IIS-2410019), Meta Reality Labs, and Google. Chen was also supported by the UD Doctoral Fellowship.

Peng and Huang are with the Department of Mechanical Engineering, University of Delaware, Newark, DE, USA. {yxpeng,ghuang}@udel.edu.

Chen is with the Department of Department of Mechanical and Aerospace Engineering, George Washington University, Washington, DC, USA. chuchu.chen@gwu.edu.

Wu is with XREAL Inc, Beijing, China. kejian@xreal.com.

(LLT)-based SRF update method, which significantly outperforms existing approaches by fully leveraging and preserving the upper-triangular structure of the SRF.
- Our $\sqrt{\text{VINS}}$ introduces a novel dynamic initialization module that, for the first time, robustly achieves initialization under the theoretical minimal case ($100ms$ with just 3 keyframes).
- We perform extensive numerical studies to highlight potential numerical challenges in VINS and underscore the advantages of the proposed $\sqrt{\text{VINS}}$. Comprehensive real-world experiments validate the notable efficiency boost of the proposed method while maintaining high accuracy.

The rest of the paper is organized as follows: After reviewing the literature focusing on VINS estimation and dynamic initialization in Section II, we describe the proposed improvement on the SRF efficiency in Section III. Section IV presents the proposed $\sqrt{\text{VINS}}$ while Section V details the proposed ultrafast dynamic initialization. The comprehensive evaluation of the proposed methods is conducted in Sections VI, VII, and VIII. Finally, the paper concludes in Section IX.

II. Related Work

A. Visual-Inertial State Estimation

An accurate, efficient and reliable state estimation algorithm is the foundation of successful VINS. From the perspective of iterative update and relinearization, VINS can be categorized into optimization-based method [7], [14], [31]–[33], and filter-based method [34]–[40]. The former performs iterative update and relinearization to solve the non-linear optimization problems in VINS and can achieve potential accuracy gain at the cost of more required computation, while the latter only performs one-time linearization and achieves superior efficiency, especially desirable for low-end platforms, where computation power is in severe constraint.

From the other perspective, VINS state estimation algorithms can also be categorized into covariance and information forms. In the former such as the extended Kalman filter (EKF) and its variants, the estimator keeps tracking the dense covariance matrix to update the estimate [6], [38], [39], [41]–[44]. In contrast, the information estimators such as extended Information filters (EIF) [45] or optimization-based methods [7], [14], [15], [31], [46], maintain the information (Hessian) matrix and exploit its sparse structure in solving for estimates. However, both covariance and information filters face challenging numerical issues, in particular on resource-constrained edge platforms [47], [48], when limited word length (32-bit float, instead of 64-bit double) is available or it is required to achieve real-time performance. In the covariance form, the covariance matrix tends to lose its positive definiteness and cause the estimator to diverge. In the information form, as the information matrix can easily become ill-conditioned (e.g., condition number larger than 10^9 [47]), naively inverting it during optimization would lead to large numerical errors (see Chapter 3.5.1 in [49]).

To address this numerical instability, there exist methods that use the square root of the information matrix instead of its full matrix and were shown to be effective to some extent in VINS [37], [48], [50]–[59]. For example, the method in [48] maintains an upper triangular square root of prior information and uses QR-decomposition to incorporate new measurements into the prior, then invert it to solve for the state update. While this estimator achieves the same accuracy with the half of the word length, it still has the concerning numerical issue with a relatively high condition number (10^5) over time, especially when paired with a high-precision IMU [60], [61], lead to substantial numerical inaccuracies for long-term operations. This numerical issue was addressed recently in [62] by preconditioning with a square root information filter.

On the contrary, VINS estimators in the covariance form tend to offer better numerical stability. For instance, in the EKF-based VINS, the only matrix that typically requires inversion, the innovation covariance, usually possesses a good condition number [47]. This makes the use of the square-root covariance matrix highly appealing for VINS, as it combines the advantages of the covariance form with the benefits of square-root properties. Surprisingly, this idea of SRF remains largely unexplored in VINS. Looking into history, the SRF has undergone significant improvements over the decades. Back in the 1960s, the initial SRF formulation was proposed by Potter and played a significant role in the Apollo project's success [63], [64], which has been extended to account for propagation (process) noise [65]. While several update methods have been developed to enhance SRF efficiency [66]–[68], these methods are generally less efficient than conventional KFs due to the disruption of the square-root covariance's triangular structure during updates. Agee [69] and Carlson [70] proposed methods that preserve the triangular structure and offer similar efficiency to KF, but these are limited to sequential updates. Modern systems, however, prefer batch updates with vector operations that enable more efficient level-3 BLAS operations [71]. These limitations of the SRF make it unsuitable for VINS applications that demand quick, real-time estimates from large-size measurements.

To address the aforementioned issues and fully harness the benefits of the square-root covariance, the recent work [9] introduced a novel (P-)QR-based SRF update and applied it to VINS, gaining computational efficiency and numerical stability. In this work, we significantly extend [9] by designing a novel LLT-based update method for SRF to achieve even greater efficiency and an ultrafast dynamic initialization module, as well as performing substantially more comprehensive experimental evaluations.

B. Visual-Inertial System Initialization

In addition to the state estimation algorithm, VINS relies on accurate initial conditions (e.g., velocity and gravity) for a successful operation. Although initial conditions can be recovered by assuming static motion, this often involves risky assumptions in practical systems. Dynamic initialization, on the other hand, enables VINS to recover key parameters while in motion, eliminating the need for the platform to remain stationary at startup. This is also crucial for practical applications, as it enables the system to re-initialize after failures and significantly enhances robustness

Dynamic initialization methods are broadly categorized into tightly coupled and loosely coupled approaches. Tightly coupled methods (closed-form solutions) recover initial states directly by integrating both visual and inertial measurements [10], [22]–[30], [72]. The first closed-form solution

for gravity direction, velocity, scale, and accelerometer bias was introduced in [72] and has been extended to include unknown camera-IMU calibration [23] with the minimal case and degenerate motion analysis [10]. In contrast to formulating a linear system, [24] formulates a MAP (Maximum A Posteriori) problem that incorporates sensor noise to further improve accuracy. Later on, [25] demonstrated the impact of gyro bias on initialization accuracy and proposed its estimation via nonlinear optimization. This was later improved by using a rotation-only constraint [73] and adding line constraints [20], [21].

On the other hand, loosely coupled methods first perform visual structure-from-motion (SfM), followed by visual-inertial alignment to match IMU measurements and recover the initial states [11]–[21]. For example, Qin et al. [13], [14] leverage a simplified SfM pipeline to obtain the up-to-scale trajectory, and then formulate a linear system that recovers scale, gravity, and velocity. The IMU uncertainty is further being considered in visual-inertial alignment to improve performance [31]. A more recent method showed that up-to-scale SfM can be formulated in a maximum-likelihood framework and constrain gravity magnitude, improving accuracy and avoiding issues with iterative solutions [17].

A key challenge for the aforementioned initialization methods is their requirement to solve for 3D feature positions during the initialization process. Given their cubic complexity with respect to the number of features, this can significantly impact efficiency. Various approaches have been proposed to address this issue. For example, [28] enhances efficiency in tightly coupled initialization by marginalizing (projecting) the depth of each feature bearing and the redundant 3DoF features in a reference frame. However, this still requires triangulating the 3D positions of features, which can make the marginalization numerically unstable under low parallax conditions, where some of the features are close to rank deficiency. Recently, [30], [74] incorporated learned image depth and leverage affine-invariant single-image depth to reduce feature parameters but relies on a depth network. The most related work is by He et al. [19], which employs the LiGT constraint [75], a linear constraint that uses known rotation and camera measurements to formulate the linear system without the need to estimate 3D features. However, their tightly coupled approach incorporates all visual measurements, resulting in a large measurement size. Their loosely coupled method solves the LiGT constraints for positions up to scale before solving for inertial states with additional velocity parameters, increasing the state size.

In contrast, we introduce a novel linear system formulation using inertial and bearing measurements to solve for initial velocity and gravity. We neither recover 3D feature positions nor retain unnecessary unknown scales, making our solution both highly efficient and robust. To ensure successful VINS operation after initialization, we also introduce an SRF-based refinement that improves state accuracy and efficiently computes the initial covariance. Unlike the Visual-inertial Bundle Adjustment (VI-BA)approach, which relies on potentially unstable matrix inversion, our method directly computes the covariance, ensuring robustness for VINS. Given these benefits, our initialization method is the first among all existing approaches to achieving the theoretical minimum condition, requiring only 3 sequential frames in just 100 ms initialization window.

III. Improving SRF Computational Efficiency

The SRF has existed for decades and potentially has better numerical stability and efficiency [76]. Specifically, unlike a standard EKF (and its variants) that estimates the (dense) covariance matrix $\mathbf{P}$, the SRF propagates and updates the corresponding upper-triangular square-root matrix $\mathbf{U}$ which is given by [76]:

$$\mathbf{U}^\top \mathbf{U} = \mathbf{P} \tag{1}$$

Because of this special structure, the SRF can represent a much broader dynamic range of numbers and reduce the condition numbers of the involved matrices (especially, the condition number of $\mathbf{U}$ as compared to that of $\mathbf{P}$). It also implicitly guarantees the symmetry and positive semi-definiteness of the covariance matrix. These numerical advantages of the SRF ensure better numerical stability (over the EKF). Furthermore, the SRF can potentially be more efficient in computation and memory usage, as it can operate with shorter word lengths (e.g., 32-bit floating-point instead of 64-bit) without compromising accuracy.

However, we have not seen its wide adoption in VINS, because it is challenging to fully capitalize on these benefits in practice. In particular, the canonical update form in the SRF is computationally expensive. For example, the Potter SRF [63], the earliest SRF method, destroys the upper triangular structure of $\mathbf{U}$ during the update, necessitating an additional costly matrix triangulation step to restore the structure. The Carlson SRF [70] improved upon the Potter form by preserving the upper triangular structure during updates, resulting in significant speedups. However, both the Potter and Carlson forms are derived for scalar measurements. While they can be extended to vector measurements through sequential scalar updates [67], [68], they fail to fully leverage Single Instruction/Multiple Data (SIMD) in modern computers to accelerate matrix operations. There are SRF update forms that support direct updates for vector measurements [59], [67], [68]. However, they either fail to maintain upper-triangular $\mathbf{U}$ or require a theoretically higher number of floating-point operations during updates, which diminish the benefit of vectorization. To address these limitations in existing SRFs, we propose a novel Cholesky decomposition (LLT)-based SRF update method. This approach overcomes computational bottlenecks by doing vector updates and properly maintaining upper-triangular $\mathbf{U}$ during the update, and also has a smaller number of operations when the measurement size is large. Moreover, we also fully exploit the upper-triangular structure of the SRF and further optimize its computation in both propagation and update.

In the following section, we first briefly introduce the standard SRF propagation step for completeness. We then present our novel update formulation, which is the core contribution of this work.

A. QR-based SRF Propagation

As the SRF propagates and updates the state estimate (mean) in the same way as the EKF [76], we here will focus on deriving EKF-equivalent propagation and update of square-root covariance matrix $\mathbf{U}$. In particular, given $\mathbf{P}_{k-1|k-1} = \mathbf{U}_{k-1|k-1}^\top \mathbf{U}_{k-1|k-1}$ leveraging QR decomposition, we can

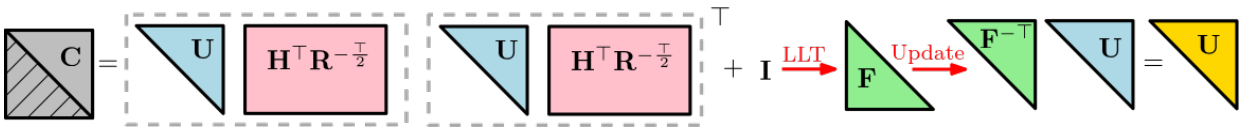


Fig. 1. Visualization of the matrix computation and its structure evolution during the proposed LLT-based SRF update.

derive the covariance and its square root propagation as follows:

$$\begin{aligned}
\mathbf{P}_{k|k-1} &= \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k-1}^\top + \mathbf{W}_{k-1} && (2)\\
&= \mathbf{\Phi}_{k-1}\mathbf{U}_{k-1|k-1}^\top\mathbf{U}_{k-1|k-1}\mathbf{\Phi}_{k-1}^\top + \mathbf{W}_{k-1}^{\frac{\top}{2}}\,\mathbf{W}_{k-1}^{\frac{1}{2}} && (3)\\
&= \begin{bmatrix}\mathbf{W}_{k-1}^{\frac{\top}{2}} & \mathbf{\Phi}_{k-1}\mathbf{U}_{k-1|k-1}^\top\end{bmatrix}\begin{bmatrix}\mathbf{W}_{k-1}^{\frac{1}{2}}\\ \mathbf{U}_{k-1|k-1}\mathbf{\Phi}_{k-1}^\top\end{bmatrix} && (4)\\
&\overset{\text{QR}}{=:} \begin{bmatrix}\mathbf{U}_{k|k-1}^\top & \mathbf{0}\end{bmatrix}\mathbf{Q}_{k-1}^\top\mathbf{Q}_{k-1}\begin{bmatrix}\mathbf{U}_{k|k-1}\\ \mathbf{0}\end{bmatrix} && (5)\\
&= \mathbf{U}_{k|k-1}^\top\mathbf{U}_{k|k-1} && (6)
\end{aligned}$$

where $\mathbf{\Phi}_{k-1}$ is the state transition or system Jacobian matrix, and we have employed the Cholesky decomposition (LLT) on the system noise covariance $\mathbf{W}_{k-1}$, i.e., $\mathbf{W}_{k-1} \overset{\text{LLT}}{=} \mathbf{W}_{k-1}^{\frac{\top}{2}}\mathbf{W}_{k-1}^{\frac{1}{2}}$, as well as the fact that $\mathbf{Q}_{k-1}$ is orthonormal. It is clear from the above derivations that the square root covariance $\mathbf{U}_{k|k-1}$ propagates from time t_{k-1} to t_k via efficient QR decomposition:

$$\begin{bmatrix}\mathbf{W}_{k-1}^{\frac{1}{2}}\\ \mathbf{U}_{k-1|k-1}\mathbf{\Phi}_{k-1}^\top\end{bmatrix} \overset{\text{QR}}{=} \mathbf{Q}_{k-1}\begin{bmatrix}\mathbf{U}_{k|k-1}\\ \mathbf{0}\end{bmatrix} \quad (7)$$

B. LLT-based SRF Update

As discussed earlier, the update is a computational bottleneck if a canonical SRF form is used. Although a recent work [9] introduced the permuted-QR (P-QR)-based square-root update to mitigate this issue, there is still room to further improve efficiency with Cholesky decomposition, which leads to our novel LLT-based efficient SRF update algorithm:

Lemma 1. *Let the measurement model be given by* $\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{n}_k$*, with measurement residual as:*[1]

$$\mathbf{r}_k := \mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1} \simeq \mathbf{H}_k\tilde{\mathbf{x}}_{k|k-1} + \mathbf{n}_k \quad (8)$$

where the measurement noise assumes $\mathbf{n}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$ *and* $\mathbf{H}_k$ *is the measurement Jacobian, the state and square-root covariance update of the SRF can be written as:*

$$\begin{aligned}
\mathbf{U}_{k|k} &= \mathbf{F}_k^{-\top}\mathbf{U}_{k|k-1} && (9)\\
\hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{U}_{k|k}^\top\mathbf{U}_{k|k}\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{r}_k && (10)
\end{aligned}$$

where $\mathbf{F}_k$ *is computed via Cholesky decomposition of:*

$$\mathbf{C}_k := \mathbf{I} + \mathbf{U}_{k|k-1}\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{U}_{k|k-1}^\top \overset{\text{LLT}}{=} \mathbf{F}_k^\top\mathbf{F}_k \quad (11)$$

Note that a similar permutation is applied prior to the Cholesky decomposition, following the approach in [9], to ensure that the resulting matrix $\mathbf{F}_k$ *is lower triangular. Consequently,* $\mathbf{F}_k^{-\top}$ *is upper triangular, as required in the update.*

[1]Throughout the paper $\hat{\mathbf{x}}$ is used to denote the estimate of a random variable $\mathbf{x}$, while $\tilde{\mathbf{x}}$ is the corresponding error state. The subscript $a|b$ denotes the estimate at time t_a by fusing all the measurements up to time t_b.

TABLE I
FLOPS OF THE DIFFERENT SRF MEASUREMENT UPDATE FORMS ASSUMING UNCORRELATED MEASUREMENTS AND CONSIDERING ONLY THE DOMINANT COMPUTATION TERMS (m MEASUREMENTS, n STATES).

Methods	Potter [63]	Carlson [70]	Kaminski [68] [59]	P-QR [9]	Proposed
Flops	$6mn^2$	$\frac{7}{2}mn^2$	$2m^2n + 5mn^2 + \frac{4}{3}n^3$	$3mn^2 + \frac{1}{3}n^3$	$2mn^2 + \frac{2}{3}n^3$

The SRF update proceeds by first computing $\mathbf{U}_{k|k}$ *in* (9)*, followed by the state update in* (10)*.*

Proof. We now show the proposed LLT-based SRF update is equivalent to the EKF:

$$\begin{aligned}
&\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1}\mathbf{H}_k^\top\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{H}_k\mathbf{P}_{k|k-1}\\
&= \mathbf{U}_{k|k-1}^\top\left(\mathbf{I} - \mathbf{U}_{k|k-1}\mathbf{H}_k^\top\left(\mathbf{H}_k\mathbf{U}_{k|k-1}^\top\mathbf{U}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{H}_k\mathbf{U}_{k|k-1}^\top\right)\mathbf{U}_{k|k-1}\\
&= \mathbf{U}_{k|k-1}^\top\left(\underbrace{\mathbf{I} + \mathbf{U}_{k|k-1}\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{U}_{k|k-1}^\top}_{\mathbf{C}_k = \mathbf{F}_k^\top\mathbf{F}_k}\right)^{-1}\mathbf{U}_{k|k-1}\\
&= \mathbf{U}_{k|k-1}^\top\mathbf{F}_k^{-1}\mathbf{F}_k^{-\top}\mathbf{U}_{k|k-1} =: \mathbf{U}_{k|k}^\top\mathbf{U}_{k|k}
\end{aligned}$$

$$\begin{aligned}
&\mathbf{P}_{k|k-1}\mathbf{H}_k^\top\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{r}_k\\
&= \mathbf{P}_{k|k-1}\mathbf{H}_k^\top\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{R}_k\mathbf{R}_k^{-1}\mathbf{r}_k\\
&= (\mathbf{P}_{k|k-1}\mathbf{H}_k^\top - \mathbf{P}_{k|k-1}\mathbf{H}_k^\top\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^\top)\mathbf{R}_k^{-1}\mathbf{r}_k\\
&= (\mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1}\mathbf{H}_k^\top\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^\top + \mathbf{R}_k\right)^{-1}\mathbf{H}_k\mathbf{P}_{k|k-1})\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{r}_k\\
&= \mathbf{P}_{k|k}\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{r}_k\\
&= \mathbf{U}_{k|k}^\top\mathbf{U}_{k|k}\mathbf{H}_k^\top\mathbf{R}_k^{-1}\mathbf{r}_k
\end{aligned}$$

□

In the above derivations, we fully exploit the *upper triangular* structure of $\mathbf{F}_k^{-\top}$. Figure 1 illustrates the matrix operations and the structural evolution throughout this process. While algebraically equivalent to the EKF, the proposed LLT-based SRF update is significantly more efficient than other square-root update methods. It avoids redundant factorizations and explicitly leverages the triangular structure for faster and more scalable computation. A detailed discussion of these efficiency gains is provided below.

We stress that efficient computation of the lower-triangular matrix $\mathbf{F}_k$ is critical to enable efficient update of the square-root covariance (9), because $\mathbf{F}_k^{-\top}$ and $\mathbf{U}_{k|k-1}$ are both upper triangular and their product (9) would be trivial and become upper-triangular by structure. To this end, when computing $\mathbf{C}_k$ (11), we exploit the upper-triangular structure of $\mathbf{U}_{k|k}$ and the symmetry of $\mathbf{C}_k$. When computing $\mathbf{U}_{k|k}$ (9), we utilize the upper triangular structure of $\mathbf{F}_k^\top$ and $\mathbf{U}_{k|k-1}$ and efficiently solve $\mathbf{F}_k^\top\mathbf{U}_{k|k} = \mathbf{U}_{k|k-1}$ via back substitution. To quantify

the computational cost, we calculate the number of arithmetic operations (i.e., floating point operations or FLOPs) required in the update under the common assumptions: (i) measurements are uncorrelated, and (ii) only the dominant (highest) order of computation is considered. Table I summarizes the computational complexity of various SRF update methods for m measurements and n states. As evident, the proposed LLT-based update method requires fewer operations when $m > \frac{4}{9}n$, as compared to the Carlson update [70]. In contrast to the P-QR-based method [9], this new update has a smaller coefficient for the term involving mn^2 while a slightly larger coefficient for the term involving n^3. As a result, the proposed method is more efficient when $m > \frac{1}{3}n$, which typically holds true for VINS, where the measurement numbers are significantly larger than the state dimensions. Theoretically, when decomposing a matrix to obtain its square root, the Cholesky-based method requires fewer FLOPs than the QR-based method, albeit with slightly reduced numerical stability [49]. However, due to the relatively small condition number of $\mathbf{C}$ in practical VINS, Cholesky-based decomposition can be employed here without encountering numerical issues.

IV. Square-Root VINS

We now apply the proposed SRF presented in the preceding section to the VINS problem and design an ultrafast and numerically stable visual-inertial state estimator, termed *Square-Root VINS (sqrt-VINS)* or $\sqrt{\text{VINS}}$. The proposed $\sqrt{\text{VINS}}$ is developed within an efficient sliding-window filtering framework by fully exploiting the specific block triangular structure of the system to achieve *faster-than-ever* performance.

First of all, we propose a special ordering of state vector to avoid unnecessary computations in $\sqrt{\text{VINS}}$. Specifically, at time t_k, the state vector $\mathbf{x}_k$ consists of the current navigation states $\mathbf{x}_{I_k}$, the calibration state $\mathbf{x}_{cb}$, historical IMU pose clones $\mathbf{x}_C$, and a set of 3D environmental (SLAM) features $\mathbf{x}_f$, in the following order:

$$\mathbf{x}_k = \begin{bmatrix}\mathbf{x}_{I_k}^\top & \mathbf{x}_{cb}^\top \mid \mathbf{x}_C^\top & \mathbf{x}_f^\top\end{bmatrix}^\top =: \begin{bmatrix}\mathbf{x}_{x_k}^\top & \mid & \mathbf{x}_C^\top & \mathbf{x}_f^\top\end{bmatrix}^\top \tag{12}$$

$$\mathbf{x}_{I_k} = \begin{bmatrix}{}^{I_k}_{G}\bar{q}^\top & {}^G\mathbf{p}_{I_k}^\top & {}^G\mathbf{v}_{I_k}^\top & \mathbf{b}_g^\top & \mathbf{b}_a^\top\end{bmatrix}^\top \tag{13}$$

$$\mathbf{x}_{cb} = \begin{bmatrix}t_d & {}^I_C\bar{q}^\top & {}^C\mathbf{p}_I^\top & \boldsymbol{\zeta}^\top\end{bmatrix} \tag{14}$$

$$\mathbf{x}_C = \begin{bmatrix}\mathbf{x}_{T_k}^\top \dots \mathbf{x}_{T_{k-c}}^\top\end{bmatrix}^\top \tag{15}$$

$$\mathbf{x}_f = \begin{bmatrix}{}^G\mathbf{f}_1^\top \dots {}^G\mathbf{f}_L^\top\end{bmatrix}^\top \tag{16}$$

where ${}^I_G\bar{q}$ is the JPL unit quaternion corresponding to rotation matrix ${}^I_G\mathbf{R}$ that represents the rotation from the global $\{G\}$ to the IMU frame $\{I\}$; We use the JPL convention to maintain consistency with the majority of prior work in filter-based VIO. For a detailed comparison between the JPL and Hamiltonian conventions, we refer readers to [77]. ${}^G\mathbf{p}_I$, ${}^G\mathbf{v}_I$, and ${}^G\mathbf{f}_i$ are the IMU position, velocity, and i-th feature position in $\{G\}$; $\mathbf{b}_g$ and $\mathbf{b}_a$ are the gyroscope and accelerometer biases; $\mathbf{x}_{T_i} = [{}^{I_i}_G\bar{q}^\top \ {}^G\mathbf{p}_{I_i}^\top]^\top$ is the i-th cloned pose; t_d and $\{{}^I_C\bar{q}^\top, {}^C\mathbf{p}_I^\top\}$ are respectively the time offset and extrinsic calibration between the camera and IMU, while $\boldsymbol{\zeta}$ is the camera intrinsic parameters.

Inspired by [59], the special ordering of the state variables (12) is discovered based on our computational complexity analysis and follows the principle: The states to be frequently marginalized are placed at the end (i.e., IMU clones and feature positions), while the variables that will be kept in the state vector (such as the current navigation states and calibration parameters) are stored in the front. By doing so, we can leverage the triangular structure of the SRF to efficiently insert and delete (marginalize) variables from the state vector without expensive re-ordering, which will become clear in the ensuing $\sqrt{\text{VINS}}$ operations (see Sections IV-B and IV-C).

A. IMU Propagation and Integration

A standard 6-axis IMU provides local linear acceleration and angular velocity measurements $\mathbf{a}_m$ and $\boldsymbol{\omega}_m$ at time t_k:

$$\mathbf{a}_m(t_k) = \mathbf{a}(t_k) - {}^I_G\mathbf{R}(t_k){}^G\mathbf{g} + \mathbf{b}_a(t_k) + \mathbf{n}_a(t_k) \tag{17}$$

$$\boldsymbol{\omega}_m(t_k) = \boldsymbol{\omega}(t_k) + \mathbf{b}_g(t_k) + \mathbf{n}_g(t_k) \tag{18}$$

where $\boldsymbol{\omega}(t_k)$ and $\mathbf{a}(t_k)$ are the true angular velocity and linear acceleration in the IMU local frame $\{I\}$, ${}^G\mathbf{g} \simeq [0, 0, -9.8]^\top$ is the known global gravitational acceleration, and $\mathbf{n}_g$ and $\mathbf{n}_a$ are the zero-mean white Gaussian noise. These IMU measurements are used to drive the inertial navigation system (INS) whose continuous-time kinematics is given by [78]:

$${}^I_G\dot{\bar{q}}(t) = \frac{1}{2}\boldsymbol{\Omega}(\boldsymbol{\omega}(t)){}^I_G\bar{q}(t) \tag{19}$$

$${}^G\dot{\mathbf{p}}_I(t) = {}^G\mathbf{v}_I(t) \tag{20}$$

$${}^G\dot{\mathbf{v}}_I(t) = {}^{I(t)}_G\mathbf{R}^\top\mathbf{a}(t) \tag{21}$$

$$\dot{\mathbf{b}}_g(t) = \mathbf{n}_{wg}(t) \tag{22}$$

$$\dot{\mathbf{b}}_a(t) = \mathbf{n}_{wa}(t) \tag{23}$$

where $\boldsymbol{\Omega}(\boldsymbol{\omega}) = \begin{bmatrix}-\lfloor\boldsymbol{\omega}\rfloor & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^\top & 0\end{bmatrix}$ and here $\lfloor\cdot\rfloor$ is the skew-symmetric matrix. We have modeled the gyroscope and accelerometer biases as random walk, with their time derivatives represented as white Gaussian processes, denoted by $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$, respectively. Integration of the above continuous-time inertial kinematics from t_k to t_{k+1} yields the following discrete-time motion model:

$${}^{I_{k+1}}_{G}\mathbf{R} = {}^{I_{k+1}}_{I_k}\Delta\mathbf{R}\ {}^{I_k}_{G}\mathbf{R} \tag{24}$$

$${}^G\mathbf{p}_{I_{k+1}} = {}^G\mathbf{p}_{I_k} + {}^G\mathbf{v}_{I_k}\Delta T + \frac{1}{2}{}^G\mathbf{g}\Delta T^2 + {}^{I_k}_G\mathbf{R}^\top{}^{I_k}\boldsymbol{\alpha}_{I_{k+1}} \tag{25}$$

$${}^G\mathbf{v}_{I_{k+1}} = {}^G\mathbf{v}_{I_k} + {}^G\mathbf{g}\Delta T + {}^{I_k}_G\mathbf{R}^\top{}^{I_k}\boldsymbol{\beta}_{I_{k+1}} \tag{26}$$

$$\mathbf{b}_{g_{k+1}} = \mathbf{b}_{g_k} + \mathbf{n}_{g_k} \tag{27}$$

$$\mathbf{b}_{a_{k+1}} = \mathbf{b}_{a_k} + \mathbf{n}_{a_k} \tag{28}$$

where $\Delta T = t_{k+1} - t_k$, $\mathbf{n}_{g_k} = \int_{t_k}^{t_{k+1}}\mathbf{n}_{wg}(u)\,du$, and $\mathbf{n}_{a_k} = \int_{t_k}^{t_{k+1}}\mathbf{n}_{wa}(u)\,du$. Importantly, ${}^{I_{k+1}}_{I_k}\Delta\mathbf{R}$ is the gyro preintegration in $[t_k, t_{k+1}]$, while ${}^{I_k}\boldsymbol{\alpha}_{k+1}$ and ${}^{I_k}\boldsymbol{\beta}_{k+1}$ are the preintegration of acceleration measurements. Due to space limitations, we do not provide full derivations of the preintegration terms. However, readers can refer to related works on IMU preintegration theory (see [79]–[81]). In particular, [81] and its associated technical report offer detailed derivations of these terms:

$${}^{I_{k+1}}_{I_k}\Delta\mathbf{R} = -\int_{t_k}^{t_{k+1}}{}^I\boldsymbol{\omega}(t_\tau)d\tau$$

$${}^{I_k}\boldsymbol{\alpha}_{I_{k+1}} = \int_{t_k}^{t_{k+1}}\int_{t_k}^{s}{}^k_u\Delta\mathbf{R}\left(\mathbf{a}_m(u) - \mathbf{b}_a(u) - \mathbf{n}_a(u)\right)duds$$

$${}^{I_k}\boldsymbol{\beta}_{I_{k+1}} = \int_{t_k}^{t_{k+1}}{}^k_u\Delta\mathbf{R}\left(\mathbf{a}_m(u) - \mathbf{b}_a(u) - \mathbf{n}_a(u)\right)du$$

With the above model model (24)-(28), we can perform the SRF propagation as described in Section III-A.

B. Stochastic Cloning and Marginalization

The proposed $\sqrt{\text{VINS}}$ adopts the sliding-window filtering methodology to control the ever-growing computational cost over time while (sub-)optimally fusing all the measurements available in the current window. This is achieved by performing efficient stochastic cloning [82] and state marginalization, which is akin to that in the multi-state constraint Kalman filter (MSCKF) [6], [83] *but* operating on the square-root covariance matrix $\mathbf{U}$ [see (1)].

To that end, we first partition the state vector (12) into the most recent navigation state $\mathbf{x}_I$, the state to be marginalized $\mathbf{x}_M$, and the remaining state $\mathbf{x}_R$, and their corresponding columns of the square-root covariance matrix are $\mathbf{U}_I$, $\mathbf{U}_M$ and $\mathbf{U}_R$, respectively. As the system propagates forward, we perform stochastic cloning [82] to augment the state and square-root covariance with those of the most recent or cloned camera/IMU pose in order to better process its corresponding measurements at a later time. As the square-root covariance for each state corresponds to the respective column blocks in $\mathbf{U}$ [59], [84], we specifically copy the cloned pose $\mathbf{x}_T$ and its corresponding column $\mathbf{U}_T$, to get the augmented state and square-root covariance as follows:

$$\begin{bmatrix}\mathbf{x}_I^\top & \mathbf{x}_M^\top & \mathbf{x}_R^\top\end{bmatrix} \xrightarrow{\text{cloning}} \begin{bmatrix}\mathbf{x}_I^\top & \mathbf{x}_T^\top & \mathbf{x}_M^\top & \mathbf{x}_R^\top\end{bmatrix} \tag{29}$$

$$\begin{bmatrix}\mathbf{U}_I & \mathbf{U}_M & \mathbf{U}_R\end{bmatrix} \xrightarrow{\text{cloning}} \begin{bmatrix}\mathbf{U}_I & \mathbf{U}_T & \mathbf{U}_M & \mathbf{U}_R\end{bmatrix} \tag{30}$$

This can be proven by demonstrating that $\mathbf{U}^\top\mathbf{U} = \mathbf{P}$ through the aforementioned process for $\mathbf{U}$ and the covariance augmentation in the conventional MSCKF. The detailed proof can be found in Section 2.3 of [84].

To bound the size of the state so as to control computational cost, we marginalize and remove the unwanted state $\mathbf{x}_M$ from (29) and its corresponding block $\mathbf{U}_M$ from (30). At the same time, because this marginalization destroys the upper triangular structure of the square-root covariance, we perform QR decomposition to ensure the resulting upper triangular $\mathbf{U}^*$:

$$\begin{bmatrix}\mathbf{U}_I & \mathbf{U}_T & \mathbf{U}_M & \mathbf{U}_R\end{bmatrix} \xrightarrow{\text{marginalization}}$$

$$\begin{bmatrix}\mathbf{U}_I & \mathbf{U}_T & \mathbf{U}_R\end{bmatrix} \overset{\text{QR}}{=} \mathbf{Q}^* \begin{bmatrix}\mathbf{U}^* \\ \mathbf{0}\end{bmatrix} \tag{31}$$

At this point, we stress the significance of the proposed special ordering of the state [see (12)]. During marginalization, by ordering the state with non-marginalized variables (e.g., $\mathbf{x}_I$ and $\mathbf{x}_{cb}$) at the top and marginalized variables (e.g., features) at the bottom, the resulting square-root covariance remains close to upper-triangular, leading to significant computational savings. Figure 2 illustrates this process: The left side shows an extreme and most ideal case where the marginalized state $\mathbf{x}_M$ is at the end of the state vector. After removing its corresponding column $\mathbf{U}_M$ from the square-root covariance, we directly obtain the upper-triangular square root covariance $\mathbf{U}^*$ without further operations. By contrast, the right side of Figure 2 shows a general case where the marginalized state in the middle of the state vector, requiring an additional QR operation to restore the upper-triangular structure. Note that, although we would like to place the marginalized state at the very bottom so as to completely avoid QR operations, in practice, this is hard to guarantee – for example, when dealing with lost track features, the marginalized state might not always be at the very end – and QR is still necessary. However, we only need to perform QR on a sub-block of the square root covariance matrix. In comparison to the square-root information filter [48] that requires processing the entire matrix, our approach remains significantly more efficient thanks to the nature of SRF marginalization.

C. Structure-Aware Efficient Measurement Update

Assume that at time t_k we have established a visual feature track within the current sliding window in which we assume M features in total being detected and tracked in the time window of $[t_{k-c}, \cdots, t_k]$. A feature $\mathbf{f}_i$ $(i = 1, \cdots, M)$ observed at time t_κ has the following nonlinear bearing measurement model by projecting its 3D position onto the 2D image plane, which is further linearized for SRF update [see (12)]:

$$\mathbf{z}_{i,\kappa} = h_d(h_p({}^{C_\kappa}\mathbf{f}_i), \boldsymbol{\zeta}) + \mathbf{n}_{i,\kappa} \quad \Rightarrow \tag{32}$$

$$\begin{aligned}\mathbf{r}_{i,\kappa} &\simeq \mathbf{H}_{I,\kappa}\tilde{\mathbf{x}}_{I,\kappa|\kappa-1} + \mathbf{H}_{cb,\kappa}\tilde{\mathbf{x}}_{cb,\kappa|\kappa-1} + \mathbf{H}_{f_i,\kappa}\tilde{\mathbf{f}}_{i,\kappa|\kappa-1} + \mathbf{n}_{i,\kappa} \\ &= \mathbf{H}_{x,\kappa}\tilde{\mathbf{x}}_{x,\kappa|\kappa-1} + \mathbf{H}_{f,\kappa}\tilde{\mathbf{x}}_{f,\kappa|\kappa-1} + \mathbf{n}_{i,\kappa}\end{aligned} \tag{33}$$

where h_d and h_p are the intrinsic distortion and projection functions, respectively. Note that h_d is general and can support any camera model (e.g., radial-tangential and equidistant [85]). In a typical visual tracking scenario, M can be large (in the order of thousands) and easily result in prohibitive computational cost if processing them without discrimination. As such, we categorize them into the SLAM features that are included in the state and the MSCKF features that will not be kept, and accordingly, partition the measurement set into different subsets depending on the observed features' categories similar as [41], [59]:

$$\{\mathbf{z}_{i,\kappa}\}_{i=1,\cdots,M}^{\kappa=k-c,\cdots,k} = \underbrace{\mathcal{Z}_N \cup \mathcal{Z}_S}_{\mathcal{Z}_{SLAM}} \cup \underbrace{\mathcal{Z}_M \cup \mathcal{Z}_U}_{\mathcal{Z}_{VIO}} \tag{34}$$

where $\mathcal{Z}_{SLAM}$ includes all the measurements available in the window related to the SLAM features that either are already in the state vector (with respect to $\mathcal{Z}_S$) or are to be initialized into the state (using $\mathcal{Z}_N$), and $\mathcal{Z}_{VIO}$ contains the remaining measurements related to the MSCKF features, some of which are to be processed at the current time ($\mathcal{Z}_M$) while the others are delayed for future update ($\mathcal{Z}_U$). Except $\mathcal{Z}_U$, we perform different update strategies for these measurement subsets to ensure efficiency and accuracy.

Without loss of generality (w.l.o.g), we first consider using the measurement set $\mathcal{Z}_N$ [see (34)] to initialize a new SLAM feature $\mathbf{f}_N$ into the state vector (12). Although we can use the delayed initialization to efficiently estimate the feature position $\hat{\mathbf{f}}_{N,\kappa|\kappa}$, its square-root covariance would be more challenging. To this end, we first stack all the linearized measurement residuals of $\mathcal{Z}_N$ [see (33)] and have:[2]

$$\mathbf{r}_N = \mathbf{H}_{N,x}\tilde{\mathbf{x}}_x + \mathbf{H}_{N,f}\tilde{\mathbf{f}}_N + \mathbf{n}_N \tag{35}$$

Note that $\mathcal{Z}_N$ typically has more than the minimum required measurements for feature initialization in order to obtain better accuracy; that is, the initialization is over constrained and $\mathbf{H}_{N,f}$ is tall. This implies that over-constrained $\mathcal{Z}_N$ also

[2]We here have dropped off the time index for brevity and employed the subscript symbol "N" to refer to the association with $\mathcal{Z}_N$. Note that similar notations are used for the other measurement sets.

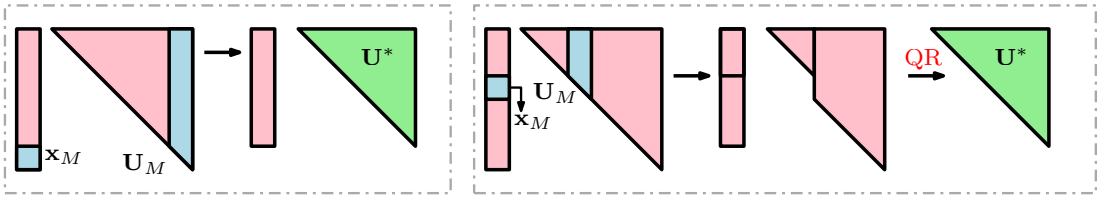


Fig. 2. An illustration of the marginalization process of $\sqrt{\text{VINS}}$ with different state ordering. The blue blocks represent the marginalized state $\mathbf{x}_M$ and its covariance block $\mathbf{U}_M$, while the pink blocks represent the other states. The left side shows an extreme case where the marginalized state is at the bottom, allowing direct extraction of the upper-triangular square-root covariance $\mathbf{U}^*$. The right side shows the marginalized state in the middle, requiring an additional QR operation. Placing marginalized states further toward the bottom improves structure preservation and reduces the QR operation cost.

encompasses information that constrains the existing state $\mathbf{x}_x$ (not just the new feature $\mathbf{f}_N$), which we seek to optimally utilize too. As evident from (35) that the range space of $\mathbf{H}_{N,f}$ essentially constrains the new feature while its nullspace constrains the existing state, we perform QR decomposition on $\mathbf{H}_{N,f}$ in order to initialize the square-root covariance as well as extract constraints on $\mathbf{x}_x$ from $\mathcal{Z}_N$:

$$\mathbf{H}_{N,f} \overset{\text{QR}}{=} \begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2\end{bmatrix} \begin{bmatrix}\mathbf{0} \\ \mathbf{J}_{N,f}\end{bmatrix} \tag{36}$$

where $\mathbf{Q}_1$ spans the nullspace of $\mathbf{H}_{N,f}$ and $\mathbf{Q}_2$ spans the range space. It is important to note that during the above QR decomposition, we use the P-QR decomposition introduced in [9] to permute $\mathbf{J}_{N,f}$ into a lower triangular matrix, instead of upper triangular. This is different from the traditional SLAM initialization [86] and can better preserve the structure of the SRF to improve efficiency. Left multiplication (33) by $[\mathbf{Q}_1 \ \mathbf{Q}_2]^\top$ yields:

$$\begin{aligned}\begin{bmatrix}\mathbf{Q}_1^\top \\ \mathbf{Q}_2^\top\end{bmatrix} \mathbf{r}_N &= \begin{bmatrix}\mathbf{Q}_1^\top \\ \mathbf{Q}_2^\top\end{bmatrix} \begin{bmatrix}\mathbf{H}_{N,f} & \mathbf{H}_{N,x}\end{bmatrix} \begin{bmatrix}\tilde{\mathbf{f}}_N \\ \tilde{\mathbf{x}}_x\end{bmatrix} + \begin{bmatrix}\mathbf{Q}_1^\top \\ \mathbf{Q}_2^\top\end{bmatrix} \mathbf{n}_N \\ \Rightarrow \begin{bmatrix}\boldsymbol{\gamma}_N \\ \boldsymbol{\varsigma}_N\end{bmatrix} &:= \begin{bmatrix}\mathbf{0} & \boldsymbol{\Gamma}_{N,x} \\ \mathbf{J}_{N,f} & \mathbf{J}_{N,x}\end{bmatrix} \begin{bmatrix}\tilde{\mathbf{f}}_N \\ \tilde{\mathbf{x}}_x\end{bmatrix} + \begin{bmatrix}\boldsymbol{\eta}_N \\ \boldsymbol{\xi}_N\end{bmatrix}\end{aligned} \tag{37}$$

where $\begin{bmatrix}\boldsymbol{\Gamma}_{N,x} \\ \mathbf{J}_{N,x}\end{bmatrix} := \begin{bmatrix}\mathbf{Q}_1^\top \\ \mathbf{Q}_2^\top\end{bmatrix} \mathbf{H}_{N,x}$. Clearly, as the top linear system: $\boldsymbol{\gamma}_N = \boldsymbol{\Gamma}_{N,x}\tilde{\mathbf{x}}_x + \boldsymbol{\eta}_N$, depends only on the existing state, it should be utilized as normal measurements like $\mathcal{Z}_S$ to update the state. On the other hand, the bottom linear system: $\boldsymbol{\varsigma}_N = \mathbf{J}_{N,f}\tilde{\mathbf{f}}_N + \mathbf{J}_{N,x}\tilde{\mathbf{x}}_x + \boldsymbol{\xi}_N$, $\boldsymbol{\xi}_N \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Omega})$, is used to efficiently initialize the square-root covariance with the new feature being included in the state as follows:

$$\mathbf{U}' = \begin{bmatrix}\mathbf{U} & -\mathbf{U}\mathbf{J}_{N,x}^{-\top}\mathbf{J}_{N,f}^{-\top} \\ \mathbf{0} & \boldsymbol{\Omega}^{\frac{1}{2}}\mathbf{J}_{N,f}^{-\top}\end{bmatrix} \tag{38}$$

This can easily be proved by performing $\mathbf{U}'^\top\mathbf{U}' = \mathbf{P}'$ to show its equivalence to the SLAM feature initialization in EKF [87].

For $\mathcal{Z}_S$, as all the measurements relate to SLAM features $\mathbf{x}_f$ that are already in the state vector, we simply stack all the measurement residuals (33) for batch update:

$$\mathbf{r}_S = \mathbf{H}_{S,x}\tilde{\mathbf{x}}_x + \mathbf{H}_{S,f}\tilde{\mathbf{x}}_f + \mathbf{n}_S \tag{39}$$

In contrast, $\mathcal{Z}_M$ correspond to the MSCKF features $\mathbf{f}_M$ which are not in the state but we still want to utilize their constraints on the state. To this end, we project the linearized measurement residual onto the left nullspace $\mathbf{N}$ of the feature Jacobian $\mathbf{H}_{M,f}$, effectively eliminating the feature dependency and improving efficiency by avoiding the need to keep these features in the state vector [6].

$$\mathbf{r}_M = \mathbf{H}_{M,x}\tilde{\mathbf{x}}_x + \mathbf{H}_{M,f}\tilde{\mathbf{f}}_M + \mathbf{n}_M \tag{40}$$

$$\mathbf{N}^\top\mathbf{r}_M = \mathbf{N}^\top\mathbf{H}_{M,x}\tilde{\mathbf{x}}_x + \mathbf{N}^\top\mathbf{n}_M \tag{41}$$

$$\Rightarrow \ \boldsymbol{\gamma}_M := \boldsymbol{\Gamma}_{M,x}\tilde{\mathbf{x}}_x + \boldsymbol{\eta}_M \tag{42}$$

We have thus far built the linearized measurement residuals [see (37), (39) and (42)] for different measurement sets (34) available in the current sliding window, which are stacked in a compact form for the SRF batch update:

$$\begin{bmatrix}\boldsymbol{\gamma}_M \\ \boldsymbol{\gamma}_N \\ \mathbf{r}_S\end{bmatrix} = \begin{bmatrix}\boldsymbol{\Gamma}_{M,x} & \mathbf{0} & \mathbf{0} \\ \boldsymbol{\Gamma}_{N,x} & \mathbf{0} & \mathbf{0} \\ \mathbf{H}_{S,x} & \mathbf{H}_{S,f} & \mathbf{0}\end{bmatrix} \begin{bmatrix}\tilde{\mathbf{x}}_x \\ \tilde{\mathbf{x}}_f \\ \tilde{\mathbf{f}}_N\end{bmatrix} + \begin{bmatrix}\boldsymbol{\eta}_M \\ \boldsymbol{\eta}_N \\ \mathbf{n}_S\end{bmatrix} \tag{43}$$

Using (43), we perform the efficient LLT-based SRF update [see (9) and (10)] for the proposed $\sqrt{\text{VINS}}$.

In particular, we take full advantage of the sparse structure of the measurement Jacobian matrix in (43) when computing the matrix $\mathbf{C}$ and its Cholesky decomposition to obtain $\mathbf{F}$ [see (11)]. Specifically, as the Jacobian of (43) has zeros (the third block column) corresponding to the new SLAM feature $\mathbf{f}_N$, we group the first two block columns corresponding to $\mathbf{x}_x$ and $\mathbf{x}_f$ as $\mathbf{H}_{xf}$. In computing $\mathbf{C}$, as $\mathbf{U}^\top$ is lower-triangular, we *only* need to multiply the non-zero blocks of $\mathbf{H}_{xf}$ with its corresponding $\mathbf{U}_{xf}^\top$, which naturally leads to *sparse* $\mathbf{C} = \begin{bmatrix}\mathbf{C}_{xf} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\end{bmatrix}$, where $\mathbf{C}_{xf} = \mathbf{U}_{xf}\mathbf{H}_{xf}^\top\mathbf{R}^{-1}\mathbf{H}_{xf}\mathbf{U}_{xf}^\top + \mathbf{I}$. As a result, we perform LLT *only* on $\mathbf{C}_{xf}$ rather than the full matrix, allowing us to efficiently solve for $\mathbf{F}_{xf}$ and thus *sparse* $\mathbf{F} = \begin{bmatrix}\mathbf{F}_{xf} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\end{bmatrix}$. We also exploit this special structure of $\mathbf{F}$ when performing the SRF update [see (9)]; that is, we only need to compute $\mathbf{F}_{xf}^{-\top}$ along with its corresponding upper triangular matrix, avoiding processing the entire matrix. Note that this saving can be significant in particular when multiple new SLAM features are initialized in the current window which is often the case in practice.

1) Outlier Rejection: Measurement outliers are inevitable in practice. Compared with an information-form estimator, we have covariance available at each update, thus, we employ the standard χ^2 test to reject them by computing the following Mahalanobis distance:

$$d_m = \mathbf{r}^\top\left(\mathbf{H}\mathbf{U}^\top\mathbf{U}\mathbf{H}^\top + \mathbf{R}\right)^{-1}\mathbf{r} \tag{44}$$

where $\mathbf{r}$ and $\mathbf{R}$ generically refer to the measurement residual and noise covariance [see (8)]. Thanks to the special structure of the measurement Jacobian $\mathbf{H}$ in our case [e.g., see (43)], we are able to compute the Mahalanobis distance (44) very efficiently. In particular, as the measurement residual of the MSCKF features $\boldsymbol{\gamma}_M$ is not related to features [see (43)], the following most expensive operation is carried out as:

$$\mathbf{U}\mathbf{H}^\top = \begin{bmatrix}\mathbf{U}_1 & \mathbf{U}_2 \\ \mathbf{0} & \mathbf{U}_3\end{bmatrix} \begin{bmatrix}\boldsymbol{\Gamma}_{M,x}^\top \\ \mathbf{0}\end{bmatrix} = \begin{bmatrix}\mathbf{U}_1\boldsymbol{\Gamma}_{M,x}^\top \\ \mathbf{0}\end{bmatrix} \tag{45}$$

Clearly, given the upper-triangular structure of $\mathbf{U}$ and the unique structure of the measurement Jacobian $\mathbf{\Gamma}_{M,x}$, we only need to compute $\mathbf{U}_1\mathbf{\Gamma}_{M,x}^\top$, instead of multiplying the measurement Jacobian with the full $\mathbf{U}$. This holds exactly the same for the measurement residual of the new SLAM features $\boldsymbol{\gamma}_N$ [see (43)]. For the SLAM feature measurement residual $\mathbf{r}_S$ (39), the sparsity of the measurement Jacobian which only relates to the corresponding camera pose and the observed features, also allows us to leverage the upper-triangular structure of $\mathbf{U}$ to compute the Mahalanobis distance d_m (44) very efficiently. Note that the computed $\mathbf{U}\mathbf{H}^\top$ (45) can also be *re-used* in the proposed SRF update to avoid repeated computation.

2) SLAM Feature Propagation: Anchor feature representation, which parameterizes features in a moving anchor frame $\{A\}$, has demonstrated robust performance in VINS [14], [37], [83], and thus is also used in this work. The anchor frame in principle can be selected as any camera frame within the sliding window that observes the feature. When the anchor frame is moving out of the current sliding window, a new anchor frame is selected and the features are transformed or anchored to this new frame. Though it appears to be straightforward, proper handling of the feature's covariance in the new anchor frame is often overlooked. To this end, we build the relation between the feature represented in the *old* anchor frame $\{A_1\}$, denoted by ${}^{A_1}\mathbf{f}$, and in the new anchor frame $\{A_2\}$, denoted by ${}^{A_2}\mathbf{f}$. This can be derived based on the fact that the global position of the static feature, ${}^{G}\mathbf{f}$, remains unchanged regardless of the choice of anchor frame:

$$ {}^{G}\mathbf{f} = {}^{A_1}_{G}\mathbf{R}^\top {}^{A_1}\mathbf{f} + {}^{G}\mathbf{p}_{A_1} = {}^{A_2}_{G}\mathbf{R}^\top {}^{A_2}\mathbf{f} + {}^{G}\mathbf{p}_{A_2} \tag{46} $$

where $\mathbf{x}_{A_i} = \{{}^{G}_{A_i}\mathbf{R}, {}^{G}\mathbf{p}_{A_i}\}$ $(i = 1, 2)$ denotes the orientation and the position of the i-th anchor frame in the global frame. Linearization of the above equations yields:

$$ \begin{aligned} & {}^{G}\tilde{\mathbf{f}} = \mathbf{H}_{A_1}\tilde{\mathbf{x}}_{A_1} + \mathbf{H}_{f_{A_1}} {}^{A_1}\tilde{\mathbf{f}} = \mathbf{H}_{A_2}\tilde{\mathbf{x}}_{A_2} + \mathbf{H}_{f_{A_2}} {}^{A_2}\tilde{\mathbf{f}} \\ \Rightarrow \quad & {}^{A_2}\tilde{\mathbf{f}} = \mathbf{H}_{f_{A_2}}^{-1}(\mathbf{H}_{f_{A_1}} {}^{A_1}\tilde{\mathbf{f}} + \mathbf{H}_{A_1}\tilde{\mathbf{x}}_{A_1} - \mathbf{H}_{A_2}\tilde{\mathbf{x}}_{A_2}) \end{aligned} \tag{47} $$

where $\mathbf{H}_{A_i}$ denotes the Jacobians with respect to anchor poses and $\mathbf{H}_{f_{A_i}}$ denotes the Jacobians with respect to anchor feature represented in different anchors. Leveraging the Jacobians in equation above and covariance of the old anchor feature, old anchor pose and new anchor pose, we perform QR-based covariance propagation introduced in Section III-A to obtain the covariance matrix of the new anchor feature ${}^{A_2}\mathbf{f}$.

D. Online Calibration

To make the proposed $\sqrt{\text{VINS}}$ more robust and easy-to-use, we also perform online calibration of the camera-IMU spatiotemporal parameters and camera intrinsics. To calibrate the extrinsic and intrinsic parameters, we include them in the state vector and build the measurement model that also depends on them (32). With that, we perform SRF update of these parameters along with the other states (see [83]). However, calibrating the time offset t_d between the IMU and camera is not that straightforward and requires more care [88]. Specifically, when a new image is available at time t_k, given the prior estimate of the time offset $\hat{t}_d$, we propagate the IMU/camera pose up to time $(t_k + \hat{t}_d)$ with the IMU readings and obtain the prior pose estimate $\hat{\mathbf{x}}_T(t_k + \hat{t}_d)$ [see (12)]. Note that the proposed $\sqrt{\text{VINS}}$ clones the *true* pose $\mathbf{x}_T(t_k + t_d)$ in order to process the visual measurements of this image at a later time (along with the other measurements in the current window). It is clear that the estimation error $\tilde{t}_d$ of the time offset contributes to the error of this cloned pose $\tilde{\mathbf{x}}_T(t_k + t_d)$, which should be carefully compensated in its covariance via SRF propagation (which is often overlooked in practice). As this is not the main contribution of our work, we refer the reader to [88] and our open-source implementation for further details.

E. Remarks

At this point, we have presented the proposed $\sqrt{\text{VINS}}$ and its main steps are summarized in Algorithm 1. We here highlight a few design choices to take advantage of the structure of the system so as to improve the efficiency.

- *Special state ordering:* The order of the state variables [see (12)] is especially designed to speed up update and marginalization process. For example, $\mathbf{x}_I$ and $\mathbf{x}_{cb}$ are prioritized at the top as they would not be marginalized, while the clones $\mathbf{x}_C$ are ordered from the latest to oldest for easy marginalization of the oldest one. The feature state, $\mathbf{x}_f$, is placed at the end, because: (i) SLAM features are marginalized frequently, (ii) this ordering allows better sparsity when computing $\mathbf{C}$ in Eq. (11) and (iii) it ensures $\mathbf{U}$ is still upper-triangular after initializing a new SLAM feature, thereby preserving the structure and improving efficiency.
- *Delayed QR and LLT operation:* As introduced in previous sections, QR decomposition is used during propagation, state cloning, marginalization, and anchor changes when an anchor feature is used. The efficiency can be further improved by skipping the QR step in propagation and cloning, with QR being performed only once during marginalization before update to maintain an upper triangular square root covariance to facilitate the subsequent update process. Meanwhile, measurements from different feature types (e.g., MSCKF, old SLAM, and new SLAM features) are processed individually and then stacked for a single, efficient LLT-based square-root update, as shown in Algorithm 1.
- *Avoiding repeated computation:* When performing SRF update, the computing of $\mathbf{C}$ is needed. As mentioned in Section IV-C1, the computation of $\mathbf{U}\mathbf{H}^\top$ during outlier rejection can be reused to enhance efficiency and avoid

Algorithm 1 $\sqrt{\text{VINS}}$

Propagation:
- Propagate the state to $\mathbf{x}(t + t_d)$ by Eq. (7)) (*skip QR*)

Clone and Marginalization:
- Cloning the latest IMU pose [Section IV-B] (*skip QR*)
- Anchor change for SLAM features by Eq. (7) (47) (*skip QR*)
- Marginalize oldest clone and lost tracked SLAM features [Eq. (31)] (**QR**)

Measurements Formulation:
Using the tracked features to formulate measurements and perform outlier rejection [Section IV-C1] to prepare for updates.
- MSCKF features via nullspace projection [Eq. (42)]
- SLAM feature initialization [Eq. (36),(37),(38)]
- SLAM features re-observation [Eq. (39)]

SRF update:
- Stack meas. [Eq. (43)] and do SRF update for both the state and the square-root covariance [Eq. (9),(10)] (**LLT**).

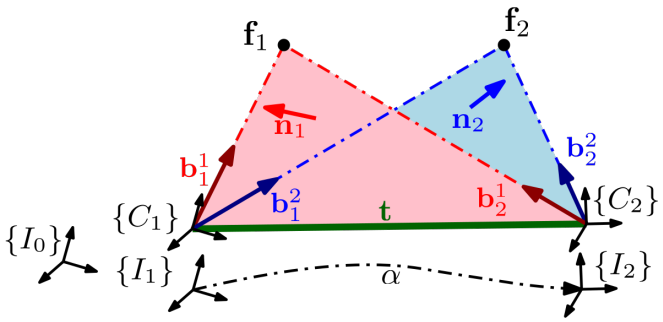


Fig. 3. An illustration of two-view geometry for featureless initialization. The pink and blue planes represent two epipolar planes formed by the features $\mathbf{f}_1$ and $\mathbf{f}_2$, and their corresponding camera frames $\{C_1\}$ and $\{C_2\}$. The bearing observations are denoted by $\mathbf{b}$, while $\mathbf{n}_1$ and $\mathbf{n}_2$ represent the normal directions of the two epipolar planes. $\{I_0\}$ indicates the initial reference frame and $\mathbf{t}$ is the direction of the relative pose.

redundant calculations. In certain cases where outlier rejection is not performed by χ^2 test, we first stack all the measurements Jacobians and then fully leverage the sparsity to perform measurement compression [6]. The triangular structure and symmetry are then exploited to efficiently compute $\mathbf{U}\mathbf{H}^\top$ and $\mathbf{C}$.

V. Robust and Fast Dynamic Initialization

Dynamic initialization is critical to ensure robust and smooth VINS operation without discontinuity. Successful initialization not only requires an initial solution but also ensures stable and continuous motion tracking afterwards. However, fast initialization with minimal data is challenging due to limited visual parallax and low signal-to-noise ratios. To address this challenge, we propose a novel dynamic initialization method, tightly integrated with the proposed SRF, providing robust and efficient initialization, even in minimal conditions.

Specifically, our initialization consists of two main steps: (i) *Feature-less initialization* efficiently recovers the initial velocity and gravity without recovering 3D feature positions to improve both robustness and efficiency; and (ii) *SRF refinement* optimizes the IMU state, features, and covariance through efficient iterative SRF updates, ensuring smooth and robust VINS operation. Note that in our initialization, as common practice, we assume the IMU biases are reasonably accurate (e.g., obtained from prior calibration) and the camera-IMU calibration and time offset are known.

A. Feature-less Initialization

To maximize efficiency, we first recover *only* the minimal states (i.e., without unobservable global position and yaw), including the local velocity (${}^{I_0}\mathbf{v}_{I_0}$) and gravity (${}^{I_0}\mathbf{g}$):

$$ {}^{I_0}\mathbf{x} = \begin{bmatrix} {}^{I_0}\mathbf{v}_{I_0}^\top & {}^{I_0}\mathbf{g}^\top \end{bmatrix}^\top \tag{48} $$

where $\{I_0\}$ is the initial IMU frame of reference. We parameterize the gravity with the minimal two parameters (α, β): ${}^{I_0}\mathbf{g}^\top = |\mathbf{g}| \begin{bmatrix} \cos\alpha \sin\beta & \sin\alpha \sin\beta & \cos\beta \end{bmatrix}^\top$. We now explain how to formulate the linear system to solve for ${}^{I_0}\mathbf{x}$.

1) IMU-induced local motion: Leveraging the inertial preintegration, we integrate the IMU measurements in the time interval $[t_0, t_k]$ to compute the relative motion in the local, instead of the global, frame of reference $\{I_0\}$ [see (24), (25) and (26)]:

$$ {}^{I_k}_{I_0}\mathbf{R} := {}^{I_k}_{I_0}\Delta\mathbf{R} \tag{49} $$

$$ {}^{I_0}\mathbf{p}_{I_k} := {}^{I_0}\mathbf{v}_{I_0}\Delta T_k + \frac{1}{2}{}^{I_0}\mathbf{g}\Delta T_k^2 + {}^{I_0}\boldsymbol{\alpha}_{I_k} \tag{50} $$

$$ {}^{I_0}\mathbf{v}_{I_k} := {}^{I_0}\mathbf{v}_{I_0} + {}^{I_0}\mathbf{g}\Delta T_k + {}^{I_0}\boldsymbol{\beta}_{I_k} \tag{51} $$

where $\Delta T_k = (t_k - t_0)$ is the time span for integration, ${}^{I_k}_{I_0}\Delta\mathbf{R}$, ${}^{I_0}\boldsymbol{\alpha}_{I_k}$ and ${}^{I_0}\boldsymbol{\beta}_{I_k}$ are obtained by IMU preintegration [81]. These can also be computed by rotating the orientation and velocity with ${}^{I_0}_{G}\mathbf{R}$ and computing the relative position change ${}^{I_0}\mathbf{p}_{I_k} = {}^{I_0}_{G}\mathbf{R}({}^{G}\mathbf{p}_{I_k} - {}^{G}\mathbf{p}_{I_0})$.

2) Camera-induced up-to-scale relative motion: With the visual measurements available in the time window $[t_o, t_k]$, we efficiently derive the relative motion direction, *without* relying on 3D features. The result is summarized as follows:

Proposition 1. *Given camera bearing measurements $\{\mathbf{b}_k^i\}$ of environmental features with known camera–IMU extrinsics, the up-to-scale relative translation direction $\mathbf{t}$ of the platform is obtained as the eigenvector corresponding to the smallest eigenvalue of*

$$ \mathbf{M} = \sum_{i=1}^{M} \mathbf{n}_i \mathbf{n}_i^\top \tag{52} $$

where $\mathbf{n}_i = \lfloor {}^{I_0}\mathbf{b}_1^i \rfloor {}^{I_0}\mathbf{b}_2^i$ denotes the normal vector of the epipolar plane formed by two camera observations represented in first IMU frame $\{I_0\}$.

Proof. As shown in Figure 3, consider an environmental feature $\mathbf{f}_i$ observed by two camera frames $\{C_1\}$ and $\{C_2\}$. Each observation $\mathbf{b}_k^i$ is a 2D bearing measurement from the k-th camera frame to the i-th feature. Based on the two-view geometry [89], an epipolar plane is thus formed with the two bearings and the relative pose $\mathbf{t}$ (e.g., see $\triangle C_1C_2\mathbf{f}_1$ in Figure 3). To solve for the relative motion, we exploit the fact that the normal of *any* epipolar plane is perpendicular to $\mathbf{t}$, which allows us to formulate an eigenvalue problem of $\mathbf{M}$ as defined in Eq. 55. Note that in this example, $\mathbf{n}_i$ is the normal direction of the epipolar plane $\triangle C_1C_2\mathbf{f}_i$. The direction $\mathbf{t}$ corresponds to the eigenvector associated with the smallest eigenvalue of $\mathbf{M}$. Specifically, the normal direction $\mathbf{n}_i$ of the epipolar plane $\triangle C_1C_2\mathbf{f}_i$ can be computed as (see Figure 3):

$$ \mathbf{n}_i = \lfloor {}^{I_0}\mathbf{b}_1^i \rfloor {}^{I_0}\mathbf{b}_2^i \tag{53} $$

where ${}^{I_0}\mathbf{b}_k^i = {}^{I_0}_{I_i}\mathbf{R}{}^{I}_{C}\mathbf{R}\mathbf{b}_k^i$ is the bearing measurement of the feature $\mathbf{f}_i$ rotated to the local IMU frame $\{I_0\}$. Note that the extrinsic calibration between the IMU and camera is assumed to be known. Geometrically, as the relative translation $\mathbf{t}$ is the intersection of all the epipolar planes with its corresponding camera frames, all the normals of these epipolar planes are perpendicular to $\mathbf{t}$, such that:

$$ \mathbf{n}_i^\top \mathbf{t} = 0 \;\Rightarrow\; \mathbf{n}_i \mathbf{n}_i^\top \mathbf{t} = \mathbf{0} \;\Rightarrow\; \underbrace{\sum_{i=1}^{M} \mathbf{n}_i \mathbf{n}_i^\top}_{\mathbf{M}} \mathbf{t} = \mathbf{0} \tag{54} $$

which implies that the relative motion $\mathbf{t}$ is an eigenvector corresponding to the zero eigenvalue (or null vector) of matrix $\mathbf{n}_i\mathbf{n}_i^\top$ and thus matrix $\mathbf{M}$, if noise free [73]. Although the 3×3 matrix $\mathbf{M}$ ideally is rank-2 with the null space spanned by $\mathbf{t}$, given noisy measurements in practice, $\mathbf{M}$ would become full-rank and thus $\mathbf{t}$ should be the eigenvector corresponding to the smallest eigenvalue. As such, determining the relative translation direction becomes an eigenvalue problem. Performing eigenvalue decomposition of $\mathbf{M}$ yields:

$$\mathbf{M} \overset{eig.}{=} \begin{bmatrix}\mathbf{t} & \mathbf{e}_1 & \mathbf{e}_2\end{bmatrix} \mathbf{Diag}(\lambda_t, \lambda_1, \lambda_2) \begin{bmatrix}\mathbf{t} & \mathbf{e}_1 & \mathbf{e}_2\end{bmatrix}^\top \tag{55}$$

where the diagonal matrix of eigenvalues arranged in an ascending order. We thus have computed the up-to-scale translation (relative motion direction) $\mathbf{t}$, as well as the eigenvectors $\mathbf{e}_1$ and $\mathbf{e}_2$ corresponding to the larger eigenvalues λ_1 and λ_2. □

For convenience, we define $\mathbf{e} := [\mathbf{e}_1, \mathbf{e}_2]$, which lies in the plane orthogonal to $\mathbf{t}$ (i.e., $\mathbf{t} \perp \mathbf{e}$) and will be used in the subsequent linear system formulation [see (62)].

3) Linear system: By combining the IMU-induced relative motion (with scale) and the camera-induced direction (without scale), we can then eliminate the unknown scale and recover the initial state through a linear system. The result is summarized below:

Proposition 2. *Given IMU preintegration over $[t_0, t_k]$, camera bearings with known camera–IMU extrinsics, and the relative translation direction $\mathbf{t}$ (with its orthogonal basis $\mathbf{e}$, such that $\mathbf{e}^\top\mathbf{t} = 0$), the initial state ${}^{I_0}\mathbf{x}$ can be obtained by solving the linear system*

$$\mathbf{e}^\top \mathbf{A}\, {}^{I_0}\mathbf{x} = \mathbf{e}^\top \mathbf{b},$$

where $\mathbf{A}$ and $\mathbf{b}$ are constructed from IMU integration and extrinsics.

Proof. Note that in the following, we use the two keyframes $\{C_1\}$ and $\{C_2\}$ in Figure 3 to illustrate our derivations:

$$s\mathbf{t} = {}^{I_0}\mathbf{p}_{C_2} - {}^{I_0}\mathbf{p}_{C_1} \tag{56}$$

$$= {}^{I_0}\mathbf{p}_{I_2} - {}^{I_0}\mathbf{p}_{I_1} + \left({}^{I_0}_{I_2}\mathbf{R} - {}^{I_0}_{I_1}\mathbf{R}\right) {}^{I}\mathbf{p}_C \tag{57}$$

where s is the unknown scale, and ${}^{I_0}\mathbf{p}_{I_2}$ and ${}^{I_0}\mathbf{p}_{I_1}$ are computed by integration with IMU measurements [see (25)]:

$${}^{I_0}\mathbf{p}_{I_1} = {}^{G}\mathbf{p}_{I_0} + {}^{I_0}\mathbf{v}_{I_0}\Delta T_1 + \frac{1}{2}{}^{I_0}\mathbf{g}\Delta T_1^2 + {}^{I_0}\boldsymbol{\alpha}_{I_1} \tag{58}$$

$${}^{I_0}\mathbf{p}_{I_2} = {}^{G}\mathbf{p}_{I_0} + {}^{I_0}\mathbf{v}_{I_0}\Delta T_2 + \frac{1}{2}{}^{I_0}\mathbf{g}\Delta T_2^2 + {}^{I_0}\boldsymbol{\alpha}_{I_2} \tag{59}$$

Substituting these into (57), we have:

$$s\mathbf{t} = \underbrace{\begin{bmatrix}\Delta T_2 - \Delta T_1 & \frac{1}{2}(\Delta T_2^2 - \Delta T_1^2)\end{bmatrix}}_{-\mathbf{A}} \begin{bmatrix}{}^{I_0}\mathbf{v}_{I_0} \\ {}^{I_0}\mathbf{g}\end{bmatrix} + \underbrace{{}^{I_0}\boldsymbol{\alpha}_{I_2} - {}^{I_0}\boldsymbol{\alpha}_{I_1} + \left({}^{I_0}_{I_2}\mathbf{R} - {}^{I_0}_{I_1}\mathbf{R}\right) {}^{I}\mathbf{p}_C}_{\mathbf{b}} \tag{60}$$

$$\Rightarrow \quad s\mathbf{t} + \mathbf{A}\, {}^{I_0}\mathbf{x} = \mathbf{b} \tag{61}$$

To further improve the efficiency in solving the above linear system (61), we remove the dependency of the unknown scale s by projecting (61) onto the nullspace of $\mathbf{t}$, such that:

$$\mathbf{e}^\top \mathbf{A}\, {}^{I_0}\mathbf{x} = \mathbf{e}^\top \mathbf{b} \tag{62}$$

where we have employed the fact that $\mathbf{e}^\top\mathbf{t} = \mathbf{0}$. □

Interestingly, as the eigenvectors $\mathbf{e}$ have already been computed during the eigenvalue decomposition of $\mathbf{M}$ (55), we effectively eliminate the need for redundant computations. It is worth mentioning that the proposed method is naturally applicable to static motion, as s will be close to zero, the ambiguity of translation direction does not negatively impact the system. Importantly, as compared to traditional initialization methods (e.g., [14], [23]), the proposed linear system (61) does *not* recover 3D feature positions, offering the following key advantages:

- *Robust to outliers:* Without estimating 3D features, the influence of outliers on the solution is minimized.
- *Robust to small parallax due to low excitation:* Since feature positions are not explicitly recovered, even if some features are near rank-deficient, it does not lead to degeneracy.
- *Better efficiency:* Without including 3D features, the problem reduces to a 6×6 linear system which is solved in constant time. The complexity with respect to the number of keyframes, k, is $\mathcal{O}(k^2)$, but since k is typically small (e.g., 3-5 frames), this is marginal. The overall complexity is dominated by the linear dependence on the number of features, $\mathcal{O}(M)$. In contrast, methods that recover 3D features have the complexity of $\mathcal{O}(M^3)$, making our approach significantly more efficient.

B. SRF Refinement

Due to measurement noise, the linear system solution (62) – though fast – inevitably would be inaccurate in practice especially given an extremely small initialization window. We thus perform iterative updates with SRF to refine the *full* (not minimal) state and the corresponding covariance, ensuring the successful operation of subsequent VINS. The full state vector of initialization is defined as:

$$\mathbf{x}_{\text{full}} = \begin{bmatrix}\mathbf{x}_K^\top & \mathbf{x}_F^\top\end{bmatrix}^\top \tag{63}$$

$$\mathbf{x}_K = \begin{bmatrix}\mathbf{x}_{I_0}^\top & \dots & \mathbf{x}_{I_N}^\top\end{bmatrix}^\top \tag{64}$$

$$\mathbf{x}_F = \begin{bmatrix}{}^{G}\mathbf{f}_1^\top & \dots & {}^{G}\mathbf{f}_M^\top\end{bmatrix}^\top \tag{65}$$

$$\mathbf{x}_{I_k} = \begin{bmatrix}{}^{I_k}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_k}^\top & {}^{G}\mathbf{v}_{I_k}^\top & \mathbf{b}_{g,k}^\top & \mathbf{b}_{a,k}^\top\end{bmatrix}^\top \tag{66}$$

where $\mathbf{x}_K$ denotes the keyframe states and $\mathbf{x}_F$ is the key features during initialization. In contrast to the commonly used dynamic initialization pipeline (e.g., [14]), which solves the VI-BA problem using an optimization solver in the information form, our approach leverages *iterative* SRF update and offers several advantages: (i) seamless integration with the SRF-based VINS, (ii) improved computational efficiency and reduced numerical issues while enabling iterative error refinement from relinearization, and (iii) direct covariance computation without the need for matrix inversion.

In particular, we first use the IMU readings to propagate the keyframe states and their corresponding square-root covariance (2), and use the obtained pose to triangulate the features, then use the feature-bearing measurements to iteratively update the states. For clarity, we use a single key feature, ${}^{G}\mathbf{f}_j$, as an example. Its measurements are stacked and linearized as [see (32) and (33)]:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}_K, {}^{G}\mathbf{f}_j) + \mathbf{n} \tag{67}$$

$$\Rightarrow \quad \mathbf{r}^{(l)} \simeq \mathbf{H}_K^{(l)} \tilde{\mathbf{x}}_K^{(l)} + \mathbf{H}_F^{(l)} {}^G\tilde{\mathbf{f}}_j^{(l)} + \mathbf{n} \tag{68}$$

where $\mathbf{H}_K^{(l)}$ and $\mathbf{H}_F^{(l)}$ are the Jacobians, and $\tilde{\mathbf{x}}_K^{(l)}$ and ${}^G\tilde{\mathbf{f}}_j^{(l)}$ are the error states for keyframes and key feature at the l-th iteration. We perform a QR decomposition on the feature Jacobian $\mathbf{H}_F$, followed by a left multiplication of the full system, which results in two decoupled systems. This step is analogous to the process shown in Eqs. (36) and (37), where we apply the same technique. For brevity, we omit the detailed derivation here and refer readers to that earlier discussion.

$$\begin{bmatrix} \mathbf{r}_1^{(l)} \\ \mathbf{r}_2^{(l)} \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{K,1}^{(l)} \\ \mathbf{H}_{K,2}^{(l)} \end{bmatrix} \tilde{\mathbf{x}}_K^{(l)} + \begin{bmatrix} \mathbf{H}_{F,1}^{(l)} \\ \mathbf{0} \end{bmatrix} {}^G\tilde{\mathbf{f}}_j^{(l)} + \mathbf{n} \tag{69}$$

For each key feature, we construct the linear systems as described above. Thus, with all the features, we stack its corresponding bottom system (69) and perform the proposed LLT-based SRF update for the key state (i.e., $\mathbf{x}_K^{(l+1)} = \mathbf{x}_K^{(l)} + d\mathbf{x}_K^{(l)}$ [see (11)]). Then we can update each key feature estimate with the top system of (69) as:

$${}^G\hat{\mathbf{f}}_j^{(l+1)} = {}^G\hat{\mathbf{f}}_j^{(l)} + \mathbf{H}_{F,1}^{(l)^{-1}} (\mathbf{r}_1^{(l)} - \mathbf{H}_{K,1}^{(l)} d\mathbf{x}_K^{(l)}) \tag{70}$$

This process of linearization and update of state estimate is repeated until convergence, while we update the square-root covariance only after being converged.

Note that this iterative SRF refinement allows us to directly obtain the initial covariance for use in the subsequent $\sqrt{\text{VINS}}$. We have chosen to initialize not only the IMU states but also retain key features $\mathbf{x}_F$ within the initialization window, along with their covariance. This ensures smooth, tightly-coupled integration into $\sqrt{\text{VINS}}$, enhancing its robustness, as discussed in the following.

We first stress that our initialization consists of two main steps, *without* and *with* 3D features involved:

1) In the feature-less initialization step, our primary focus is to estimate the initial velocity and gravity as quickly as possible, *without* recovering 3D feature positions.
2) During SRF refinement step, we seamlessly transition to $\sqrt{\text{VINS}}$, by initializing the inertial state *with* the SLAM features and their corresponding covariance. These features can be immediately tracked in the subsequent VINS process and used for instant updates, preserving all information within the initialization window.

Note that traditional methods take a completely opposite approach: they first recover the initial state using 3D feature positions and then refine only the IMU states in a second step. In contrast, our proposed feature-less linear system offers robustness to outliers and small parallax, while significantly improving efficiency. For the refinement step, our approach retains features from the initialization window, enabling immediate updates when new measurements of the same features become available. This ensures that all information is preserved, tightly coupled, and seamlessly transferred to the VINS module. Traditional methods, by comparison, solve only for the 15-DOF IMU state during refinement, which requires additional time and frames for VINS to initialize new features and operate effectively. Furthermore, SRF directly accesses the state covariance without needing to invert the information matrix—a process commonly required in BA-based initialization methods—thereby avoiding potential numerical instability, particularly in challenging initialization scenarios.

TABLE II
SIMULATION PARAMETERS AND PRIOR STANDARD DEVIATIONS FOR MEASUREMENT PERTURBATIONS.

Parameter	Value	Parameter	Value
Gyro. White Noise	2.0e-4	Gyro. Rand. Walk	2.0e-5
Accel. White Noise	5.0e-4	Accel. Rand. Walk	4.0e-4
Cam Freq. (Hz)	10	IMU Freq. (Hz)	400
Num. Clones	11	Tracked Feat.	100
Max. MSCKF Feat.	40	Max. SLAM Feat.	50

TABLE III
RMSE VALUES FOR ORIENTATION (DEG.) AND POSITION (M) BASED ON 200 RUNS ON UD-ARL WITH DIFFERENT ESTIMATORS.

Methods	EKF	SRF (QR)	SRF (LLT)	SRIF
double	0.957 / 0.146	0.957 / 0.146	0.956 / 0.146	0.957 / 0.146
float	0.960 / 0.146	0.959 / 0.146	0.958 / 0.146	1.045 / 0.174

As a result, by leveraging these advantages, our initialization method achieves high robustness and efficiency.

VI. NUMERICAL STUDIES OF LLT-BASED SRF

We now present the simulation results of the proposed LLT update method for SRF, showcasing its improved numerical stability and efficiency. To ensure a fair comparison across estimators, we use OpenVINS [36], which implements an EKF-based approach as the baseline (denoted EKF(d)), along with a float version (EKF(f)). Additionally, we evaluate double- and float-precision versions of the square-root information filter (SRIF(d) and SRIF(f)) and the proposed square-root covariance $\sqrt{\text{VINS}}$ (SRF(d) and SRF(f)).

A. Accuracy and Robustness of SRF

We first aim to demonstrate that $\sqrt{\text{VINS}}$ offers improved numerical stability compared to other forms of estimators. To evaluate this, we use a 30-minute, 2.4 km UD-ARL trajectory (see Figure 4) and generate realistic visual bearing and inertial measurements, as summarized in Table II.

Table III reports the average Root Mean Square Error (RMSE) of different estimators based on 200 Monte Carlo runs. In Figure 5, the top two plots show the orientation and position errors of various estimators with both double

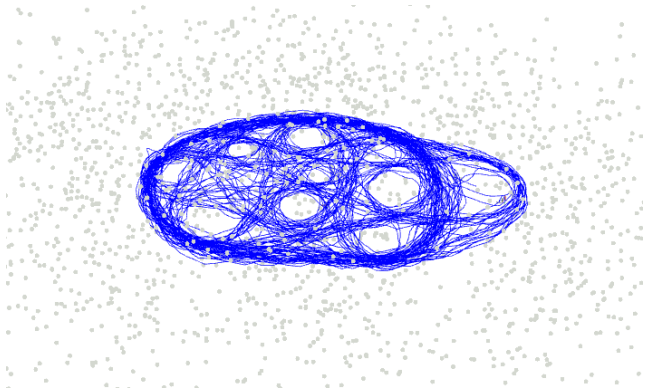


Fig. 4. Simulated 2.4km UD-ARL trajectory.

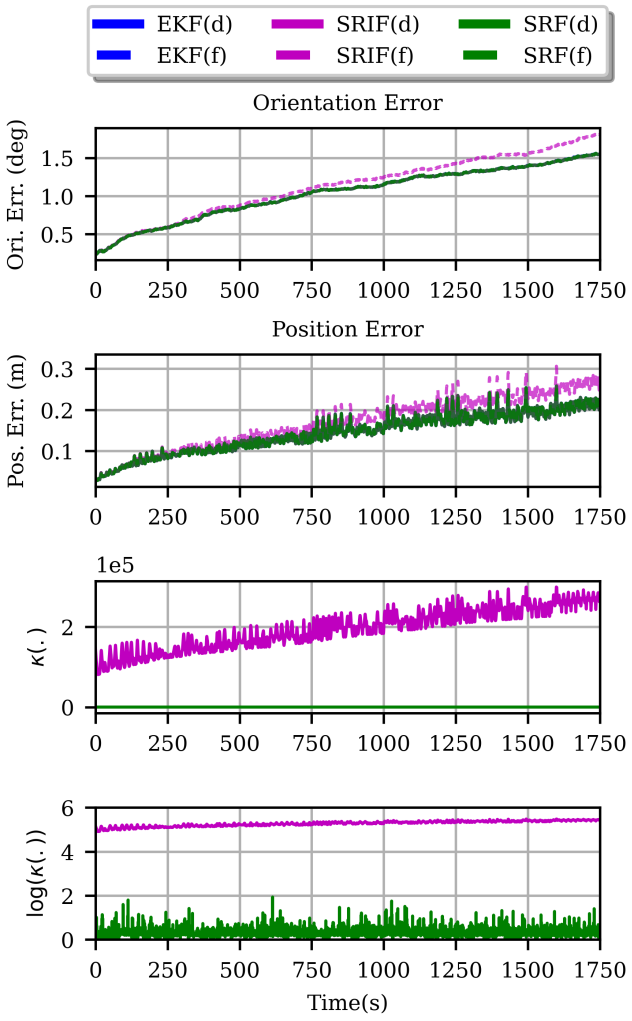


Fig. 5. **Top:** Orientation/position errors of different estimators performed on UD-ARL dataset. 'd' is for double; 'f' is for float. While most estimators perform similarly and are hard to distinguish from the plot, SRIF(f) shows a clear drop in accuracy over time. **Bottom:** Condition numbers of the square-root information matrix (purple line) and the Cholesky decomposed matrix $\mathbf{C}$ (green line, see Eq. (11)), presented in both standard (scientific) and logarithmic scales.

and float precision. The bottom two plots the standard and logarithmic condition numbers of the square-root information matrix (purple) and the $\mathbf{C}$ matrix [see Eq. (11)] for the LLT-based update over time (green). Given the covariance matrix $\mathbf{P}$, the square-root information matrix $\mathbf{R}$ is $\mathbf{R}^\top\mathbf{R} = \mathbf{P}^{-1}$. Note that this figure only presents the LLT-based SRF results, as the different update methods are expected to exhibit nearly identical accuracy performance, as shown in Table III.

Note that the authors have carefully reported the condition numbers of these two matrices to evaluate numerical stability. For SRIF, the most challenging step is inverting the square-root information matrix $\mathbf{R}$, so we report its condition number. For the LLT-based SRF, two numerical challenges arise: i) the Cholesky decomposition of $\mathbf{C}$, which requires a positive definite matrix, and ii) the inversion of $\mathbf{F}$. But $\mathbf{F}$ is the square root of $\mathbf{C}$, we plot the condition number of $\mathbf{C}$ as it also reflects that of $\mathbf{F}$. From Figure 5, it is clear that the condition number

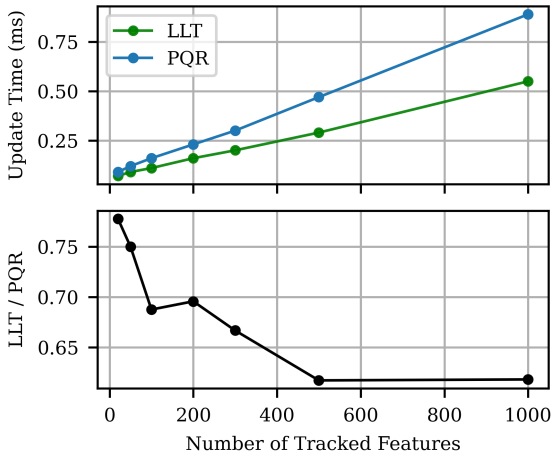


Fig. 6. Comparison of update efficiency between P-QR and LLT update methods. The top plot illustrates the update times for the two methods: the blue line represents the original P-QR-based method, and the green line represents the proposed LLT-based method. The bottom plot shows the time ratio of LLT to P-QR update times (LLT/P-QR), indicating that the LLT-based method becomes increasingly efficient as the number of tracked features increases.

of $\mathbf{C}$ (green line) remains stable, with a magnitude close to 1 and a maximum below 10^2, demonstrating the improved numerical stability of the proposed SRF. Intuitively, $\mathbf{F}^{-\top}$ serves as the transition matrix between $\mathbf{U}_{k|k-1}$ and $\mathbf{U}_{k|k}$ (the square-root covariance before and after propagation). As long as the measurement uncertainty (i.e., camera measurement noise) is not significantly smaller than the state uncertainty after propagation, we expect $\mathbf{F}^{-\top}$ to be close to the identity matrix and well-conditioned. Consequently, $\mathbf{C}$ will share the same benefit, which is almost always the case in practical VINS applications.

From the figure, we also observe as the condition number of $\mathbf{R}$ grows larger than $2e^5$, both orientation and position errors of SRIF(f) start showing a degraded performance compared to other estimators. This can also be seen in Table III, the float SRIF is inaccurate with large RMSE values. This is likely due to the numerical issue when performing inversion on ill-conditioned $\mathbf{R}$ to solve for state update under limited machine precision (see Chapter 3.5.1 in [49]). In contrast, the covariance-form estimators, both EKF and the proposed SRF, no matter which update method is used (i.e., P-QR or LLT) demonstrated consistent performance regardless of using double or float. This is evident from the comparable RMSE values in Table III, as well as the consistent error trends in Figure 5.

B. *Efficiency of LLT-based SRF Update*

In recent work [9], the efficiency improvement of the P-QR-based SRF compared to other estimators was reported. Here, we aim to thoroughly compare the novel LLT-based SRF with P-QR, demonstrating its efficiency improvements with theoretical guarantees, as shown in Table I. To make our experimental evaluation more comprehensive, we will also evaluate and compare all forms of estimators in the following sections.

In this numerical study, we fix the state size and vary the number of tracked features to evaluate performance with

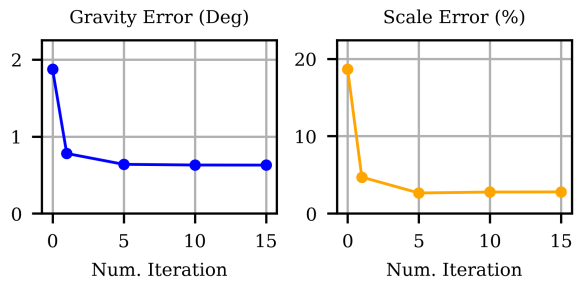


Fig. 7. The impact of the number of iterations for initialization accuracy in monocular setup with the 0.5-second window in TUM-VI Dataset.

different numbers of measurements. All features are treated as MSCKF features and the results are reported in Figure 6.

The top figure plots the update time (in ms) for both methods as the number of features changes, while the bottom figure shows the ratio of the update times (i.e., LLT/P-QR). The results clearly demonstrate that the LLT-based method is significantly more efficient, with the performance gap widening as the number of features increases. Moreover, the time ratio between the two methods (bottom figure) decreases progressively as the number of measurements becomes more dominating and, in the end, converges to roughly the theoretical ratio $\frac{2}{3}$, which proves the FLOPs analysis as shown in the Table I.

VII. Experimental Evaluation of System Initialization

To demonstrate the enhanced performance of the proposed novel dynamic initialization method, we validate it using two widely recognized and publicly available visual-inertial (VI) datasets: EuRoC MAV [90] and TUM-VI [91]. We compare our method (**Ours**) with the state-of-the-art dynamic initialization approach in OpenVINS, a reimplementation of Dongsi's method [92], referred to as **DS** in the following sections. We also evaluate the performance with both monocular and stereo cameras. In stereo setup, except for independent KLT tracking of both cameras, we also perform tracking between each stereo pair to formulate stereo constraint. The parallax gained from stereo allows easy feature triangulation even in not fully excited motion. To evaluate initialization performance, we divide each sequence into 10-second windows, run initialization at each entry point, and average the results across all runs. The keyframes are selected based on the average parallax, and all the features obtained from tracking are used to formulate the linear system constraints.

A. Initialization Accuracy

We begin by reporting the accuracy of the initialized scale and gravity under various setups, as shown in Figure 8. The figure presents the gravity error (top) and the scale error (bottom) for different initialization methods, including DS (red), our method without iterative SRF refinement (blue), and our method (green), evaluated with varying window sizes (in second) for both monocular and stereo setups. Specifically, a Sim(3) transformation is fitted between the estimated and ground truth trajectories. The scale error is computed as $100 \times (\max(s, 1/s) - 1)$, where s represents the scale obtained from the Sim(3) transformation.

From the figure, it is evident that the initialization errors decrease as the initialization time increases due to the availability of more measurements. However, the key advantage of the proposed initialization method lies in its ability to successfully initialize within a very short time frame (0.1 s), where DS method fails. The results also highlight the improved performance achieved with the iterative update in SRF especially with small window sizes. To further illustrate this, we compare the gravity and scale errors across different numbers of iterations, as shown in Figure 7. It is evident that the errors consistently decrease with an increasing number of iterations and eventually converge. Interestingly, we also observe that as the window size increases, the accuracy gain from refinement decreases. Because with the increasing motion for inertial measurements and increasing parallax for visual measurements, the signal-noise-ratio increases. Thanks to the robust novel constraint formulation, our method achieves desirable accuracy without further refinement. This indicates the possibility to reduce or eliminate iterations for further efficiency gains, as discussed in the following section.

B. Successful Initiation Rate

We now present the success rates based on our defined criteria for success under various system initialization setups. Traditionally, the success rate is defined by dividing each sequence into small windows, running initialization at each entry point, and calculating the percentage of successful dynamic initializations. It is important to note that we have applied a stricter criterion for success, given the strong performance of our proposed method. Our criterion ensures initialization is tested at various points within each sequence, assessing both the accuracy of the initialization window poses and the subsequent VIO performance using the initialization results. The criteria are as follows:

- We divide each sequence into 10-second windows, run the initialization with a different initialization window (from 0.1 to 1s) at each entry point. The initial states must be successfully solved for *all* runs in the sequences, including ensuring that the linear system can be solved without issues. Additionally, the initial states must converge after refinement, and their covariance must be successfully obtained.
- The position ATE of VIO within the first 10 seconds must remain below a predefined threshold.

Given these criteria, the results for the Euroc MAV dataset are shown in Table IV, while Table V reports the results for the TUM VI dataset. In these tests, we vary the initialization window (from 0.1s to 1.0s) and report the success rates for different thresholds (i.e., 0.1m, 0.3m, and 0.5m). In the tables, success rates higher than 95% are highlighted in green, while those below 95% are marked in red. Results are not reported if the defined success criteria are not met.

The results clearly demonstrate that the proposed SRF method consistently achieves a higher success rate compared to DS. For instance, with monocular setup initialization windows of 0.5s, 0.75s, and 1.0s, SRF significantly outperforms DS for both datasets. Our method also consistently outperforms DS, especially in more challenging scenarios. For

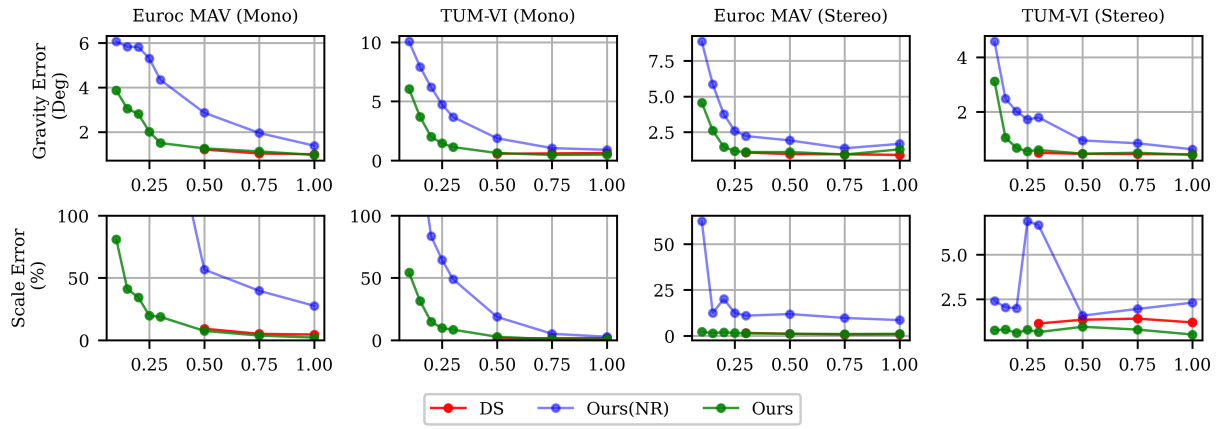


Fig. 8. Initialization gravity error and scale error across different window sizes (time) and datasets are evaluated for both monocular and stereo setups. The results are compared using the DS method, our proposed method, and our method without iterative refinement, denoted as Ours (NR).

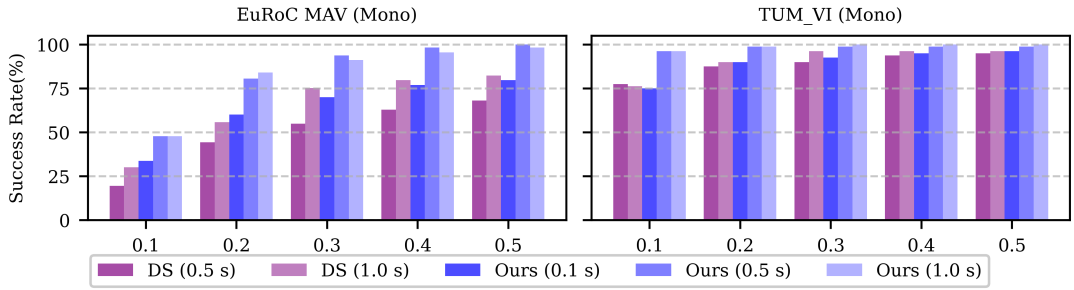


Fig. 9. Successful initialization rates for the Euroc MAV dataset (left) and TUM VI dataset (right) with a 0.5m VIO position error threshold. The results compare the DS method with 0.5s and 1.0s windows to our method with 0.1s, 0.5s, and 1.0s windows, represented in different colors.

TABLE IV
INITIALIZATION SUCCESS RATES ON THE EUROC MAV DATASET.

	Threshold (m)	DS							Ours						
		0.1 s	0.15 s	0.2 s	0.3 s	0.5 s	0.75 s	1.0 s	0.1 s	0.15 s	0.2 s	0.3 s	0.5 s	0.75 s	1.0 s
Mono	0.1	-	-	-	-	19.5	28.3	30.1	33.6	43.4	48.7	51.3	47.8	51.3	47.8
	0.3	-	-	-	-	54.9	63.7	75.2	69.9	83.2	82.3	86.7	93.8	86.7	91.2
	0.5	-	-	-	-	68.1	76.1	82.3	79.6	87.6	87.6	92.0	**100.0**	**96.5**	**98.2**
Stereo	0.1	-	-	-	45.1	44.2	47.8	43.4	63.7	65.5	68.1	69.0	64.6	62.8	61.1
	0.3	-	-	-	84.1	86.7	82.3	85.8	94.7	**96.5**	**96.5**	**96.5**	**96.5**	**95.6**	94.7
	0.5	-	-	-	90.3	92.0	90.3	93.8	**96.5**	**99.1**	**99.1**	**100.0**	**99.1**	**98.2**	**98.2**

Note: This table compares the DS method and our proposed method across different initialization window sizes for monocular and stereo setups on the TUM VI dataset. Rates $\geq 95\%$ are shown in **green and bold**, while rates $< 95\%$ are shown in red. Missing entries indicate failure to meet the defined success criteria.

TABLE V
INITIALIZATION SUCCESS RATES ON THE TUM VI DATASET.

	Threshold (m)	DS							Ours						
		0.1 s	0.15 s	0.2 s	0.3 s	0.5 s	0.75 s	1.0 s	0.1 s	0.15 s	0.2 s	0.3 s	0.5 s	0.75 s	1.0 s
Mono	0.1	-	-	-	-	77.5	77.5	76.2	75.0	82.5	91.2	93.8	**96.2**	**100.0**	**96.2**
	0.3	-	-	-	-	90.0	90.0	**96.2**	92.5	91.2	**96.2**	**97.5**	**98.8**	**100.0**	**100.0**
	0.5	-	-	-	-	95.0	93.8	**96.2**	**96.2**	93.8	**97.5**	**100.0**	**98.8**	**100.0**	**100.0**
Stereo	0.1	-	-	-	91.2	86.2	92.5	**98.8**	**98.8**	**100.0**	**100.0**	**100.0**	**100.0**	**98.8**	**100.0**
	0.2	-	-	-	**97.5**	**98.8**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**98.8**	**100.0**
	0.3	-	-	-	**97.5**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**98.8**	**100.0**

Note: This table compares the DS method and our proposed method across different initialization window sizes for monocular and stereo setups on the TUM VI dataset. Rates $\geq 95\%$ are shown in **green and bold**, while rates $< 95\%$ are shown in red. Missing entries indicate failure to meet the defined success criteria.

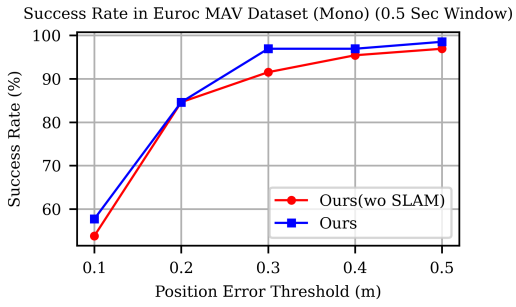


Fig. 10. Visual-inertial tracking successful rate over different position error threshold using dynamic initialization in EurocMAV Dataset with 0.5-second initialization window and monocular setup with and without initialization of SLAM features in the state.

example, for the Euroc MAV dataset (shown in Table IV), our method successfully initializes VIO with a 33% success rate for a 0.1m position error threshold and a 79.6% success rate for a 0.5m position error threshold using a 0.1s window. In contrast, the DS method fails to guarantee successful initialization in such a short time period. These results demonstrate that the proposed method is well-constrained, robust to changes in window size, and significantly more efficient and reliable compared to DS. To further illustrate this, Figure 9 shows the success rates for both the DS method and our proposed method under 0.5s and 1.0s initialization windows. Additionally, we include results for our method with a 0.1s window. From this plot, it is evident that, given the same initialization window, our method consistently outperforms DS (e.g., Ours (0.5s) achieves a higher success rate compared to DS (0.5s)). Impressively, Ours (0.1s) not only outperforms DS (0.5s) but also achieves comparable performance to DS (1.0s), demonstrating that our method is ultrafast and highly effective, even with minimal initialization windows.

We also investigate the inclusion of SLAM features and their impact on initialization, as shown in Figure 10. Incorporating SLAM features significantly increases the success rate. As discussed in Section V-B, keeping SLAM features allows more information to be preserved after initialization, effectively enhancing the robustness and improving the performance of VIO. Notably, despite the inclusion of additional states (i.e., SLAM features), the proposed method achieves remarkable efficiency, enabled by our streamlined linear system formulation and SRF update methods, as demonstrated in the following section.

C. Initialization Timing Analysis

We report the runtime of our initialization method across various window sizes on both a laptop and a Jetson Nano, compared with the DS method, as shown in Figure 11.[3] Results for the DS initialization method are not reported for initialization windows shorter than 0.5s because it fails under these conditions. We should note that our refinement time also includes the time used for feature triangulation. But in DS's method this is calculated in initial guess step. For covariance recovery, which is part of the refinement stage, DS only includes recovery of the 15-DoF IMU state, while we also allow for recovery of covariance of SLAM features. The results clearly show that the proposed method with a 0.1s initialization window is the most efficient. For a 0.5s initialization window, ours demonstrates significantly improved efficiency compared to DS. Notably, when running in Jetson Nano, DS's method can not be initialized before the next camera measurement comes (for a 20 Hz camera, dynamic initialization is required to finish within 50 ms to guarantee real-time), while our system can still achieve real-time performance.

This improvement stems from several key factors. First, the proposed method avoids the need to solve for feature positions in the linear system, significantly reducing the computational time required for generating the initial guess. Second, the efficient iterative update mechanism effectively minimizes errors while maintaining computational efficiency. Furthermore, the proposed method provides direct access to the covariance matrix without requiring the inversion of the information matrix—a process that is computationally expensive and numerically unstable, often necessitating inflation of the initial covariance to stabilize the system. These advantages make ours method not only accurate and robust but also highly efficient.

We further discuss the 0.1-second window (3 keyframes) and 0.5-second window (5 keyframes) reported in Figure 11 for our method to provide a deeper understanding. The key difference lies in the number of keyframes and the total number of measurements. As discussed in Section V. at the initial guess stage, the computational complexity of our method is quadratic with respect to the number of keyframes and linear with respect to the number of tracked features per frame. Therefore, using only three keyframes requires approximately half the computation time compared to five keyframes. In the refinement stage, the measurement size has a greater impact than the state size, as the latter remains relatively small. Since our refinement method has linear complexity with respect to measurements, the computation time for 5 keyframes increases linearly compared to 3 keyframes. As mentioned in Section VII-A, our method provides a good initial guess for large initialization windows. When computational resources are limited, it is possible to skip iterative refinement and use a conservative initial covariance, allowing the system to initialize in 0.6 ms on a laptop and 2.8 ms on a low-end embedded system.

VIII. Experimental Evaluation of $\sqrt{\text{VINS}}$

In this section, we demonstrate the capabilities of $\sqrt{\text{VINS}}$ in comparison with state-of-the-art VINS. The system is tested with both double-precision ($\sqrt{\text{VINS}}$ (d)) and float-precision ($\sqrt{\text{VINS}}$ (f)) versions. For comparison, we use OpenVINS [36] in its original double-precision (OpenVINS(d)) and implement the float-precision version (OpenVINS(f)), which required tuning to avoid divergence due to negative covariance diagonals. Additionally, we compare with RVIO2 [53], a square-root inverse filter VIO with a robocentric state formulation, and VINS-Mono [14], an optimization-based sliding-window VIO.

Since RVIO2 uses only MSCKF features by default, resulting in a smaller state size, we also test in a similar configu-

[3]Computational results were performed in a single thread on Laptop with an Intel(R) Core(TM) i7-11800H @ 2.30GHz and Jetson Nano with ARM Cortex-A57 4 Core @ 1.5GHz.

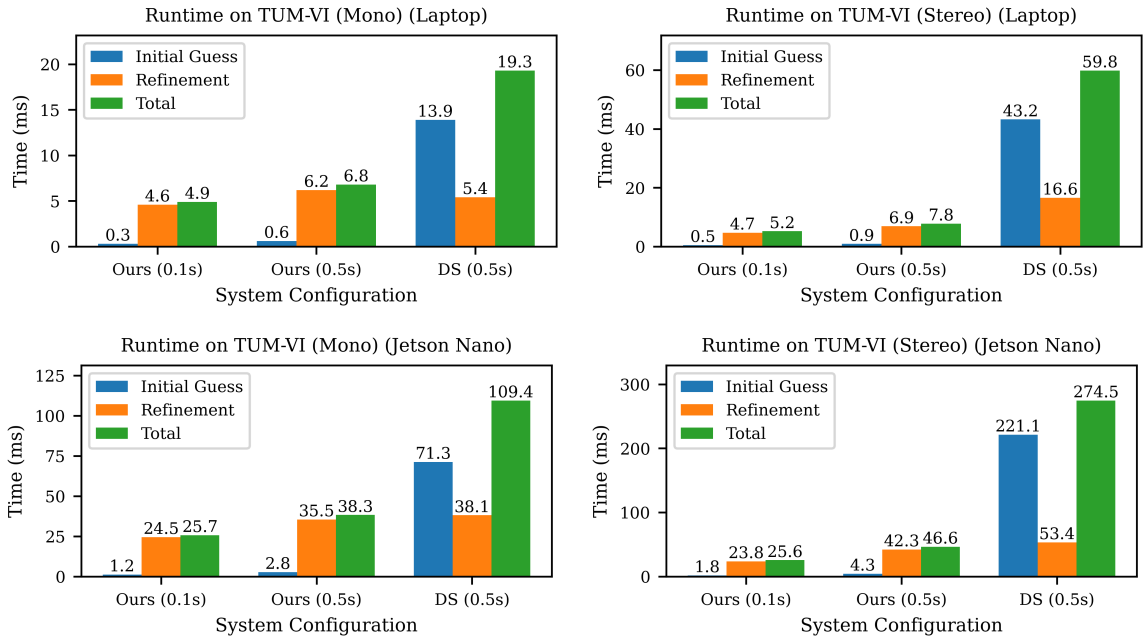


Fig. 11. Initialization runtime on the TUM VI dataset evaluated on a laptop (top) and Jetson Nano (bottom) for different system configurations, comparing our method with the DS method.

TABLE VI
AVERAGE ABSOLUTE TRAJECTORY ERROR (ATE) IN DEGREES/METERS.

Algo.	V101	V102	V103	V201	V202	V203	MH01	MH02	MH03	MH04	MH05
OpenVINS(d)	0.70 / 0.06	1.67 / 0.06	2.88 / 0.07	0.95 / 0.10	1.38 / 0.06	1.28 / 0.14	1.74 / 0.10	0.91 / 0.17	1.14 / 0.12	0.95 / 0.25	1.03 / 0.41
OpenVINS(f)	0.71 / 0.06	1.66 / 0.06	2.87 / 0.06	0.94 / 0.10	1.40 / 0.06	1.25 / 0.14	1.76 / 0.10	0.91 / 0.17	1.18 / 0.13	0.94 / 0.25	1.04 / 0.41
$\sqrt{\text{VINS}}$(d)	0.54 / 0.06	1.60 / 0.06	2.94 / 0.05	1.09 / 0.10	1.41 / 0.06	1.36 / 0.12	1.90 / 0.11	0.77 / 0.14	1.06 / 0.12	1.02 / 0.23	1.14 / 0.35
$\sqrt{\text{VINS}}$(f)	0.54 / 0.06	1.65 / 0.05	2.68 / 0.06	1.09 / 0.10	1.48 / 0.06	1.17 / 0.11	1.97 / 0.10	0.74 / 0.14	0.88 / 0.11	0.99 / 0.25	1.13 / 0.35
$\sqrt{\text{VINS}}$(M)	0.63 / 0.07	1.73 / 0.08	1.76 / 0.07	0.80 / 0.07	1.39 / 0.09	1.48 / 0.14	2.18 / 0.16	0.57 / 0.15	1.59 / 0.23	0.59 / 0.14	0.49 / 0.30
$\sqrt{\text{VINS}}$(L)	0.73 / 0.05	1.83 / 0.10	2.75 / 0.06	0.71 / 0.06	1.22 / 0.07	1.53 / 0.15	1.32 / 0.13	0.73 / 0.15	1.57 / 0.23	0.86 / 0.21	0.74 / 0.40
RVIO2	0.88 / 0.09	2.27 / 0.10	2.02 / 0.10	2.19 / 0.13	1.90 / 0.11	1.50 / 0.15	2.60 / 0.17	1.00 / 0.15	1.08 / 0.19	1.10 / 0.24	0.95 / 0.32
EqVIO	0.66 / 0.05	2.64 / 0.14	3.37 / 0.19	1.31 / 0.10	1.67 / 0.18	1.71 / 0.20	2.14 / 0.14	0.89 / 0.15	1.11 / 0.09	2.09 / 0.35	1.29 / 0.23
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	1.82 / 0.23	2.51 / 0.41	0.94 / 0.29

Note: 'd' and 'f' indicate the use of double and float precision, respectively. $\sqrt{VINS}$(M)uses float precision and MSCKF features for comparison with RVIO2. $\sqrt{VINS}$(L)is a lightweight configuration optimized for ultra-efficiency.

TABLE VII
ESTIMATOR RUNTIME (MS) EXCLUDING FEATURE TRACKING ON EUROCMAV (LAPTOP).

	OpenVINS	$\sqrt{\text{VINS}}$	$\sqrt{\text{VINS}}$(M)	$\sqrt{\text{VINS}}$(L)	RVIO2	EqVIO	VINS-Mono
Double	4.2	2.4	-	-	-	0.9	22.4
Float	3.0	1.7	0.7	0.4	1.8	-	-

Note: Estimator runtime (ms) excluding feature tracking on EuRocMAV (Laptop). The efficiency of EqVIO primarily stems from its relatively compact state size compared to other methods. However, further speed improvements may be achieved by reformulating the system in SRF form.

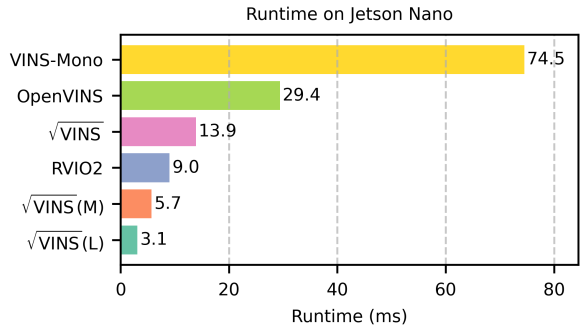


Fig. 12. Estimator runtime (ms) excluding feature tracking on EuRocMAV (Jetson Nano).

ration (15 clones, 200 tracked features, all MSCKF), denoted as $\sqrt{\text{VINS}}$(M), for a fair comparison. Finally, $\sqrt{\text{VINS}}$(L) is a lightweight setup that features ultra-efficiency, which tracks 150 features, keeps a maximum of 4 clones, and 15 SLAM features, and uses a maximum of 20 MSCKF features in the update. We also compared another state-of-the-art EKF-based system EqVIO [38], [39]that features equivariant formulation and impressive computational efficiency. In the following, we report the performance of the above mentioned systems on the EurRoC MAV dataset [90] and the Aria Everyday Activities Dataset [93].

1) EurRoC MAV Dataset: To evaluate this dataset, we use the default configuration of OpenVINS. This setup extracts 200 sparse features, keep 11 clones, and tracks up to 50

TABLE VIII
RESULTS ON THE ARIA EVERYDAY ACTIVITIES DATASET.

Algo.	Location 1	Location 2	Location 3	Location 4	Location 5	Avg.	Runtime
OpenVINS*	1.02 / 0.05	1.25 / 0.05	1.22 / 0.04	1.30 / 0.04	1.38 / 0.07	1.23 / 0.05	2.1
$\sqrt{\text{VINS}}$	1.13 / 0.04	0.95 / 0.04	1.13 / 0.04	1.42 / 0.05	1.15 / 0.06	1.16 / 0.05	0.9

Note: Accuracy is reported as Average Absolute Trajectory Error (ATE) in degrees/meters; estimator runtime is in microseconds.
*Some OpenVINS runs diverged (position error $>$ 1m); only successful runs are reported, which may unfairly favor $\sqrt{\text{VINS}}$.

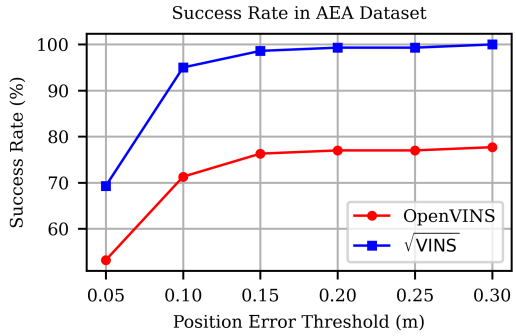


Fig. 13. Successful rate over different position error thresholds with dynamic initialization in Aria Everyday Activities (AEA) Dataset.

Fig. 14. Example images from the Aria Everyday Activities (AEA) Dataset.

SLAM features and 40 MSCKF features. The system performs online calibration for camera-IMU extrinsics, time offsets, and camera intrinsics. Evaluation is conducted using only the left camera, with initialization performed from a static state.

The averaged ATE values are reported in Table VI. It is clear that the estimation accuracy of $\sqrt{\text{VINS}}$(d), $\sqrt{\text{VINS}}$(f), OpenVINS(d), and OpenVINS(f) are very similar as expected. They are not exactly the same in the real world due to two reasons. First, χ^2 test is adopted to reject outliers and robustify the estimator and might introduce randomness. For example, in certain cases, $\sqrt{\text{VINS}}$(d) might reject measurements that pass χ^2 test in $\sqrt{\text{VINS}}$(f) because of slight numerical differences, this will cause different versions to use different measurements and have different performance. Second, OpenVINS performs a "sequential" update, which first processes MSCKF features and then SLAM features for the consideration of efficiency, while $\sqrt{\text{VINS}}$ performs the update all at once. This also introduces differences in the state linearization points. Compared with RVIO2, EqVIO and VINS-mono, $\sqrt{\text{VINS}}$ also achieves superior performance in almost all the sequences. Surprisingly, even $\sqrt{\text{VINS}}$(M) and $\sqrt{\text{VINS}}$(L) achieves similar or even better performance than the other systems.

We then look into the efficiency of the estimators as reported in Table VII. Feature tracking is also a crucial components in VINS, however, as it is not the focus of this work, we exclude its runtime in the efficiency analysis. Clearly, $\sqrt{\text{VINS}}$ is much faster than OpenVINS, reducing the runtime by half. Regardless of being in double or float format, $\sqrt{\text{VINS}}$ consistently prevails over OpenVINS. Remarkably, the double precision $\sqrt{\text{VINS}}$ even outperforms the float OpenVINS. VINS-Mono runs the slowest as it performs iterative optimization. RVIO2 is also developed in float and shows excellent efficiency, but with a similar setup and a fair comparison, $\sqrt{\text{VINS}}$(M) still prevails. EqVIO achieves strong efficiency, outperforming vanilla $\sqrt{\text{VINS}}$ but remaining less efficient than $\sqrt{\text{VINS}}(L)$ and $\sqrt{\text{VINS}}(M)$. Its efficiency stems primarily from the SLAM-based state design: unlike MSCKF-based VINS, which maintains multiple clones, EqVIO keeps only a single clone while tracking a small number of SLAM features, thereby reducing both state and measurement dimensions, while compromising accuracy. Importantly, however, EqVIO's key novelty lies in its use of Lie-group symmetry to improve estimator consistency. This contribution is orthogonal to our work, implying that a VIO system can, in principle, combine Lie-group symmetry with square-root filtering. In EqVIO, the Riccati matrix plays a role analogous to the covariance in the EKF. Tracking it in square-root form allows the use of SRF-based propagation and update methods, enabling robust operation under lower floating-point precision, exploiting triangular and symmetric structures for additional speedups, while simultaneously benefiting from the improved consistency offered by the equivariant formulation. Finally, $\sqrt{\text{VINS}}$(L) achieves the best efficiency with 0.4 ms in estimator runs, which means it can run over 2.5kHz, especially suitable for running on a computation-constrained platform. The efficiency gain mainly comes from the proposed LLT-based SRF update method, which fully explored the problem structure (state order, upper-triangular covariance, Jacobian structure, avoid redundent computation) as discussed in Section IV.

We also report the estimator runtime on the Jetson Nano, as shown in Figure 12. VINS-Mono fails to meet the real-time requirement (50 ms), $\sqrt{\text{VINS}}$ demonstrates the highest efficiency among systems with features in the state (e.g., OpenVINS and VINS-Mono) and achieves comparable efficiency to RVIO2, which uses only MSCKF features. $\sqrt{\text{VINS}}$(L) requires only 3.1 ms in an estimator run, which is almost 10 times faster than the default baseline OpenVINS, achieving ultra-efficiency.

2) Aria Everyday Activities Dataset: Aria Everyday Activities (AEA) dataset [93] provides an ego-centric perspective view recorded from Aria AR glasses, as shown in Figure 14. It contains 143 daily activity sequences recorded by multiple

users in 5 indoor locations. The Aria glasses have two 10 Hz cameras and two IMUs. In the evaluation, both cameras are used, and only the right 1000 Hz IMU is used in our evaluation. The challenge for this dataset is that the two tracking cameras have limited view overlap so that does not provide good stereo depth. Also, the dataset is recorded mostly within motion at the beginning and requires dynamic initialization. In certain sequences, the cameras face a white textureless wall or door or have a large portion of dynamic objects (e.g. the user carries an object in front of the glass) further challenges feature tracking and the overall system robustness.

In our eveluation, OpenVINS and $\sqrt{\text{VINS}}$ use the same setup, which tracks 200 features, uses a maximum of 6 clones, 15 SLAM features, and 40 MSCKF features in the update. Factory calibrations parameters from the dataset are used. For dynamic initialization, both systems use a 0.5-second window with 5 keyframes.

We first report the success rate for this challange dataset, reported in Figure 13. In this test, after initialization, the subsequent VINS is run for the full trajectory, and the position error is evaluated at the end. The figure illustrates the percentage of runs that result in VIO achieving a position error below a certain threshold. It is easy to observe that $\sqrt{\text{VINS}}$ significantly outperforms in this challenging AR/VR scenarios, achieving over 95% success with VIO errors under 0.1m at the end, compared to just 70% for OpenVINS.

We also report the VIO accuracy and estimator runtime on the AEA dataset in Table VIII. For OpenVINS, only successful runs with position errors under 1 meter are included, whereas full results are presented for $\sqrt{\text{VINS}}$, as all runs were successful. Despite this seemingly 'unfair' comparison favoring OpenVINS, $\sqrt{\text{VINS}}$ demonstrates superior accuracy. Runtime performance is also highlighted, with $\sqrt{\text{VINS}}$ taking 0.9 ms compared to OpenVINS's 2.1 ms. Together with previous results, this evaluation further confirms the superior efficiency of $\sqrt{\text{VINS}}$.

IX. Conclusions and Future Work

In this paper, we have developed a complete, robust and fast square-root VINS ($\sqrt{\text{VINS}}$) with a faster-than-ever dynamic initialization which is able to robustly initialize the system even in just 100 ms with 3 keyframes. Our system sets a new VINS benchmark, offering over twice the speed, improved numerical stability, and enhanced robustness compared to SOTA algorithms of 3D motion tracking. The square-root covariance filter (SRF) offers substantial benefits for VINS, especially in embedded systems, thanks to its improved numerical stability and efficiency. However, exploiting these advantages has been hindered by inefficiencies in the update process, particularly with large measurements. Building on a recent work [9], this work proposed a novel LLT-based SRF update method, which leverages the structure of the VINS problem to achieve high efficiency and operational effectiveness in $\sqrt{\text{VINS}}$. Additionally, the proposed dynamic initialization in $\sqrt{\text{VINS}}$ ensures rapid and reliable initialization under minimal conditions, marking a *first* in the literature. This capability is critical for practical deployments, where quick reinitialization is often required following system resets or failures. Extensive numerical studies and real-world experiments have validated the robustness and efficiency of $\sqrt{\text{VINS}}$. Notably, our system achieves twice the speed of SOTA methods while maintaining high accuracy, even under challenging conditions such as 32-bit single-precision float operations, and the real-world results further demonstrate its superior performance across diverse scenarios for edge computing platforms. Future work will focus on extending $\sqrt{\text{VINS}}$ to support multi-sensor fusion and operate in dynamic environments, further improving scalability and robustness. We hope this work—and its open-source release—opens new possibilities for fast, reliable, and resource-efficient state estimation across a wide range of applications.

References

[1] K. J. Wu, C. X. Guo, G. Georgiou, and S. I. Roumeliotis, "VINS on wheels," in *Proc. of the IEEE International Conference on Robotics and Automation*, May 2017, pp. 5155–5162.

[2] 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.

[3] 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.

[4] Y. Peng, C. Chen, and G. Huang, "Quantized visual-inertial odometry," in *Proc. International Conference on Robotics and Automation*, Yokohama, Japan, May 2024.

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

[6] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Rome, Italy, Apr. 10–14, 2007, pp. 3565–3572.

[7] 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.

[8] G. J. Bierman, *Factorization methods for discrete sequential estimation*. Courier Corporation, 2006.

[9] 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.

[10] A. Martinelli, "Closed-form solution of visual-inertial structure from motion," *International Journal of Computer Vision*, vol. 106, no. 2, pp. 138–152, Jan 2014.

[11] L. Kneip, S. Weiss, and R. Siegwart, "Deterministic initialization of metric state estimation filters for loosely-coupled monocular vision-inertial systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2011, pp. 2235–2241.

[12] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular slam with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.

[13] T. Qin and S. Shen, "Robust initialization of monocular visual-inertial estimation on aerial robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 4225–4232.

[14] 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.

[15] L. Von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2510–2517.

[16] C. Campos, J. M. Montiel, and J. D. Tardós, "Inertial-only optimization for visual-inertial initialization," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 51–57.

[17] D. Zuñiga-Noël, F.-A. Moreno, and J. Gonzalez-Jimenez, "An analytical solution to the imu initialization problem for visual-inertial systems," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 6116–6122, 2021.

[18] H. Wei, T. Zhang, and L. Zhang, "A fast analytical two-stage initial-parameters estimation method for monocular-inertial navigation," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[19] Y. He, B. Xu, Z. Ouyang, and H. Li, "A rotation-translation-decoupled solution for robust and efficient visual-inertial initialization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 739–748.

[20] J. He, M. Li, Y. Wang, and H. Wang, "Ple-slam: A visual-inertial slam based on point-line features and efficient imu initialization," *arXiv preprint arXiv:2401.01081*, 2024.

[21] W. Wang, C. Chou, G. Sevagamoorthy, K. Chen, Z. Chen, Z. Feng, Y. Xia, F. Cai, Y. Xu, and P. Mordohai, "Stereo-nec: Enhancing stereo visual-inertial slam initialization with normal epipolar constraints," *arXiv preprint arXiv:2403.07225*, 2024.

[22] A. Martinelli, "Closed-form solutions for attitude, speed, absolute scale and bias determination by fusing vision and inertial measurements," Ph.D. dissertation, INRIA, 2011.

[23] T.-C. Dong-Si and A. I. Mourikis, "Estimator initialization in vision-aided inertial navigation with unknown camera-imu calibration," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 1064–1071.

[24] M. Li and A. I. Mourikis, "A convex formulation for motion estimation using visual and inertial sensors," in *Proceedings of the Workshop on Multi-View Geometry, held in conjunction with RSS. Berkeley, CA*, 2014.

[25] J. Kaiser, A. Martinelli, F. Fontana, and D. Scaramuzza, "Simultaneous state initialization and gyroscope bias calibration in visual inertial aided navigation," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 18–25, 2016.

[26] J. Domínguez-Conti, J. Yin, Y. Alami, and J. Civera, "Visual-inertial slam initialization: A general linear formulation and a gravity-observing non-linear optimization," in *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, 2018, pp. 37–45.

[27] C. Campos, J. M. Montiel, and J. D. Tardós, "Fast and robust initialization for visual-inertial slam," in *International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1288–1294.

[28] G. Evangelidis and B. Micusik, "Revisiting visual-inertial structure-from-motion for odometry and slam initialization," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1415–1422, 2021.

[29] Y. Zhou, A. Kar, E. Turner, A. Kowdle, C. X. Guo, R. C. DuToit, and K. Tsotsos, "Learned monocular depth priors in visual-inertial initialization," in *European conference on computer vision*. Springer, 2022, pp. 552–570.

[30] N. W. Merrill, P. Geneva, S. Katragadda, C. Chen, and G. Huang, "Fast monocular visual-inertial initialization leveraging learned single-view depth." in *Robotics: Science and Systems*, 2023.

[31] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[32] V. Usenko, N. Demmel, D. Schubert, J. Stückler, and D. Cremers, "Visual-inertial mapping with non-linear factor recovery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 422–429, 2019.

[33] C. Chen, P. Geneva, Y. Peng, W. Lee, and G. Huang, "Optimization-based vins: Consistency, marginalization, and fej," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Detroit, MI., 2023.

[34] 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.

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

[36] P. Geneva, K. Eckenhoff, and G. Huang, "A linear-complexity EKF for visual-inertial navigation with loop closures," in *Proc. International Conference on Robotics and Automation*, Montreal, Canada, May 2019.

[37] Z. Huai and G. Huang, "Markov parallel tracking and mapping for probabilistic slam," in *Proc. of the IEEE International Conference on Robotics and Automation*, Xi'an, China, 2021.

[38] P. van Goor and R. Mahony, "An equivariant filter for visual inertial odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 14 432–14 438.

[39] ——, "Eqvio: An equivariant filter for visual-inertial odometry," *IEEE Transactions on Robotics*, 2023.

[40] 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, (submitted).

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

[42] P. Geneva, J. Maley, and G. Huang, "An efficient schmidt-ekf for 3D visual-inertial SLAM," in *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, Jun. 2019.

[43] Y. Yang, C. Chen, W. Lee, and G. Huang, "Decoupled right invariant error states for consistent visual-inertial navigation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1627–1634, 2022.

[44] 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.

[45] 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, Sep. 2011, pp. 65–72.

[46] 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.

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

[48] K. J. Wu, A. M. Ahmed, G. A. Georgiou, and S. I. Roumeliotis, "A square root inverse filter for efficient vision-aided inertial navigation on mobile devices," in *Robotics: Science and Systems Conference (RSS)*, 2015.

[49] G. H. Golub and C. F. Van Loan, *Matrix computations*. JHU press, 2013.

[50] K. J. Wu and S. I. Roumeliotis, "Inverse schmidt estimators," *Multiple Autonomous Robotic System Laboratory, Department of Computer Science & Engineering, University of Minnesota, Tech. Rep. Number-2016-003*, 2016.

[51] D. Caruso, A. Eudes, M. Sanfourche, D. Vissiere, and G. Le Besnerais, "An inverse square root filter for robust indoor/outdoor magneto-visual-inertial odometry," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2017, pp. 1–8.

[52] T. Ke, K. J. Wu, and S. I. Roumeliotis, "Rise-slam: A resource-aware inverse schmidt estimator for slam," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 354–361.

[53] Z. Huai and G. Huang, "Square-root robocentric visual-inertial odometry with online spatiotemporal calibration," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9961–9968, 2022.

[54] F. Dellaert and M. Kaess, "Square root SAM: Simultaneous localization and mapping via square root information smoothing," *International Journal of Robotics Research*, vol. 25, no. 12, pp. 1181–1203, Dec. 2006.

[55] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Fast incremental smoothing and mapping with efficient data association," in *Proceedings IEEE international conference on robotics and automation*, 2007, pp. 1670–1677.

[56] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *International Journal of Robotics Research*, vol. 31, pp. 217–236, Feb. 2012.

[57] N. Demmel, D. Schubert, C. Sommer, D. Cremers, and V. Usenko, "Square root marginalization for sliding-window bundle adjustment," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 260–13 268.

[58] M. W. Givens and J. W. McMahon, "Square-root extended information filter for visual-inertial odometry for planetary landing," *Journal of Guidance, Control, and Dynamics*, vol. 46, no. 2, pp. 231–245, 2023.

[59] 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

[60] P. Chauchat, A. Barrau, and S. Bonnabel, "Factor graph-based smoothing without matrix inversion for highly precise localization," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 3, pp. 1219–1232, 2020.

[61] P. Chauchat, S. Bonnabel, and A. Barrau, "Invariant smoothing with low process noise," in *IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 4758–4763.

[62] T. Ke, P. Agrawal, Y. Zhang, W. Zhen, C. X. Guo, T. Sharp, and R. C. Dutoit, "Pc-srif: Preconditioned cholesky-based square root information filter for vision-aided inertial navigation," *arXiv preprint arXiv:2409.11372*, 2024.

[63] J. Potter and R. Stern, "Statistical filtering of space navigation measurements," in *Guidance and Control Conference*, 1963, p. 333.

[64] R. H. Battin, "Astronautical guidance," 1964.

[65] P. Dyer and S. McReynolds, "Extension of square-root filtering to include process noise," *Journal of Optimization Theory and Applications*, vol. 3, pp. 444–458, 1969.

[66] J. Bellantoni and K. Dodge, "A square root formulation of the kalman-schmidt filter." *AIAA journal*, vol. 5, no. 7, pp. 1309–1314, 1967.

[67] A. Andrews, "A square root formulation of the kalman covariance equations." *Aiaa Journal*, vol. 6, no. 6, pp. 1165–1166, 1968.

[68] P. Kaminski, A. Bryson, and S. Schmidt, "Discrete square root filtering: A survey of current techniques," *IEEE Transactions on automatic control*, vol. 16, no. 6, pp. 727–736, 1971.

[69] W. S. Agee and R. H. Turner, "Triangular decomposition of a positive definite matrix plus a symmetric dyad with application to kalman filtering," *White Sands Missile Range Tech. Rep*, vol. 38, 1972.

[70] N. A. Carlson, "Fast triangular formulation of the square root filter." *AIAA journal*, vol. 11, no. 9, pp. 1259–1265, 1973.

[71] J. J. Dongarra, J. Du Croz, S. Hammarling, and I. S. Duff, "A set of level 3 basic linear algebra subprograms," *ACM Transactions on Mathematical Software (TOMS)*, vol. 16, no. 1, pp. 1–17, 1990.

[72] A. Martinelli, "State estimation based on the concept of continuous symmetry and observability analysis: The case of calibration," *IEEE Transactions on Robotics*, vol. 27, no. 2, pp. 239–255, 2011.

[73] L. Kneip and S. Lynen, "Direct optimization of frame-to-frame rotation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2352–2359.

[74] 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.

[75] Q. Cai, L. Zhang, Y. Wu, W. Yu, and D. Hu, "A pose-only solution to visual reconstruction and navigation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 73–86, 2021.

[76] P. S. Maybeck, *Stochastic models, estimation, and control*. Academic press, 1982.

[77] J. Sola, "Quaternion kinematics for the error-state kalman filter," *arXiv preprint arXiv:1711.02508*, 2017.

[78] A. B. Chatfield, *Fundamentals of High Accuracy Inertial Navigation*. AIAA, 1997.

[79] 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, Feb. 2012.

[80] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Proc. of the Robotics: Science and Systems Conference*, Rome, Italy, Jul. 13–17, 2015.

[81] 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.

[82] S. I. Roumeliotis and J. W. Burdick, "Stochastic cloning: A generalized framework for processing relative state measurements," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Washington, DC, May 11-15, 2002, pp. 1788–1795.

[83] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "Openvins: A research platform for visual-inertial estimation," in *Proc. of the IEEE International Conference on Robotics and Automation*, Paris, France, 2020. [Online]. Available: https://github.com/rpng/open_vins

[84] Y. Peng, C. Chen, and G. Huang, "Technical report: Ultrafast square-root filter-based vins," University of Delaware, Tech. Rep., 2024. [Online]. Available: https://udel.edu/~ghuang/papers/tr_srf.pdf

[85] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[86] M. Li, "Visual-inertial odometry on resource-constrained systems," Ph.D. dissertation, UC Riverside, 2014.

[87] ——, *Visual-inertial odometry on resource-constrained systems*. University of California, Riverside, 2014.

[88] M. Li and A. I. Mourikis, "Online temporal calibration for camera–imu systems: Theory and algorithms," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 947–964, 2014.

[89] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[90] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016.

[91] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1680–1687.

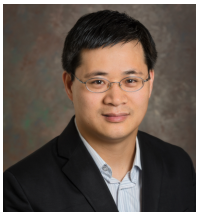
[92] T. Dong-Si and A. I. Mourikis, "Initialization in vision-aided inertial navigation with unknown camera-imu calibration," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Portugal, Oct. 2012, pp. 1064–1071.

[93] Z. Lv, N. Charron, P. Moulon, A. Gamino, C. Peng, C. Sweeney, E. Miller, H. Tang, J. Meissner, J. Dong *et al.*, "Aria everyday activities dataset," *arXiv preprint arXiv:2402.13349*, 2024.

Yuxiang Peng (Student Member, IEEE) received the Bachelor of Engineering degree in Mechatronics Engineering from Zhejiang University, Hangzhou, China, in 2019. Now he is a Ph.D. candidate in Mechanical Engineering at the University of Delaware, Newark, DE, USA. He is a member of the Robot Perception and Navigation Group (RPNG) at the University of Delaware. His research interests focus on efficient state estimation and spatial computing for embedded and resource-constrained robotic and XR platforms. He was a Finalist for the Best Paper Award on Robot Vision at the ICRA 2024.

Chuchu Chen (Member, IEEE) received the B.Eng. degree in Automation from Harbin Engineering University (HEU), China, in 2017, and the Ph.D. degree in Mechanical Engineering from the University of Delaware (UD), Newark, DE, USA, in 2025, where she worked in the Robot Perception and Navigation Group (RPNG). She is currently an Assistant Professor in the Department of Mechanical and Aerospace Engineering at The George Washington University (GWU), Washington, DC, USA, where she directs the Estimation, Perception, and Intelligent Computing (EPIC) Lab. Her research interests lie at the intersection of state estimation, spatial perception, and embodied intelligence. She has received multiple recognitions, including the Doctoral Fellowship for Excellence at the University of Delaware, the Best Student Paper Award Finalist at RSS 2023, and the Best Paper Award Finalist at ICRA 2024 (Robot Vision).

Kejian Wu received a Bachelor of Engineering in Information and Communication Technology from Zhejiang University, Hangzhou, China, in 2010, and a Ph.D. in Electrical and Computer Engineering from University of Minnesota - Twin Cities, Minneapolis, MN, USA, in 2020. His research interests focus on simultaneous localization and mapping (SLAM), visual-inertial odometry (VIO), computer vision, estimation theory, sensor fusion and calibration, and artificial intelligence. From 2013 to 2018, he was a member of UMN MARS Lab's partnership with Project Tango, Google, and an intern visual-inertial system researcher, Daydream, Google. He is currently the co-founder and head of perception at XREAL Inc, where he leads the perception and algorithm team at the company, building technical foundations of AR glasses.



Guoquan (Paul) Huang (Senior Member, IEEE) received his BS in automation (electrical engineering) from the University of Science and Technology Beijing, China, in 2002, and MS and PhD in computer science from the University of Minnesota–Twin Cities, in 2009 and 2012, respectively. He currently is a Professor of Mechanical Engineering (ME) and Computer and Information Sciences (CIS) at the University of Delaware (UD), where he is leading the Robot Perception and Navigation Group (RPNG). From 2012 to 2014, he was a Postdoctoral Associate with the MIT CSAIL (Marine Robotics). His research interests focus on state estimation and spatial computing for autonomous robots and mobile devices, including sensing, calibration, localization, mapping, perception, and navigation of ground, aerial and underwater vehicles. He serves as an Associate Editor for the IEEE Transactions on Robotics (T-RO) and IET Cyber-Systems and Robotics (CSR). Dr. Huang has received multiple honors and awards including the ICRA 2022 Best Paper Award (Navigation), 2022 GNC Journal Best Paper Award, and the Finalists of the ICRA 2024 Best Paper Award (Robot Vision), RSS 2023 Best Student Paper Award, ICRA 2021 Best Paper Award (Robot Vision), RSS 2009 Best Paper Award, and Mid-Career Faculty Excellence in Scholarship Award, as well as several faculty research awards from Google and Meta Reality Labs, and the NSF and NASA Research Initiation Awards.