Decoupled Right Invariant Error States for Consistent Visual-Inertial Navigation

Yulin Yang, Chuchu Chen, Woosik Lee and Guoquan Huang

***Abstract*—The invariant extended Kalman filter (IEKF) is proven to preserve the observability property of visual-inertial navigation systems (VINS) and suitable for consistent estimator design. However, if features are maintained in the state vector, the propagation of IEKF will become more computationally expensive because these features are involved in the covariance propagation. To address this issue, we propose two novel algorithms which preserve the system consistency by leveraging the invariant state representation and ensure efficiency by decoupling features from covariance propagation. The first algorithm combines right invariant error states with first-estimates Jacobian (FEJ) technique, by decoupling the features from the Lie group representation and utilizing FEJ for consistent estimation. The second algorithm is designed specifically for sliding-window filter-based VINS as it associates the features to an active cloned pose, instead of the current IMU state, for Lie group representation. A new pseudo-anchor change algorithm is also proposed to maintain the features in the state vector longer than the window span. Both decoupled right- and left-invariant error based VINS methods are implemented for a complete comparison. Extensive Monte-Carlo simulations on three simulated trajectories and real world evaluations on the TUM-VI datasets are provided to verify our analysis and demonstrate that the proposed algorithms can achieve improved accuracy than a state-of-art filter-based VINS algorithm using FEJ.**

***Index Terms*—Visual-Inertial SLAM; Localization; Mapping; Invariant Extended Kalman Filter**

I. Introduction and Related Work

DURING the past decades, visual-inertial navigation systems (VINS) have been widely applied to self-driving cars, AR/VR and autonomous robots especially in GPS-denied scenarios [1]. By leveraging a low-cost sensor rig containing both an IMU and camera, a lightweight, robust and accurate VINS algorithm [2]–[8] can be deployed to resource-constrained devices for challenging applications. The majority of VINS algorithms are either based on filter or batch optimization. In spite of many great works on batch optimization-based VINS [2], [3], [7], [9], the filter-based algorithms [6], [10]–[14] are still extensively used mainly due to their simplicity and efficiency. It has been investigated that VINS is not fully observable [15] and has four unobservable directions related to global yaw and global translation. The standard extended Kalman filter (std-EKF) based VINS algorithms suffer from inconsistency issues [5], [15], [16] caused by overconfidence of state estimates. This is mainly because certain unobservable directions will erroneously become observable due to the linearization point change when computing state transition matrix and measurement Jacobians. Spurious information will be gained along with the unobservable directions. In this case, the system uncertainty will be underestimated and the estimation accuracy will also be downgraded.

Efforts [5], [6], [11], [15]–[25] have been devoted to tackling the inconsistency problems of VINS through different methodologies. Based on system observability analysis, Huang et al. [26] proposed a first-estimates Jacobian (FEJ) algorithm, which avoids the inconsistent problem by intentionally fixing the linearization points with initial estimates. FEJ was successfully applied to multi-state-constraint Kalman filter (MSCKF) based VINS systems [5], [14], [27] and achieved state-of-art performances. However, the estimation performance of FEJ can be easily hurt due to bad state initialization. Huang et al. [16] also proposed observability-constrained (OC)-EKF for consistent estimator design, which enforces the unobservable subspace by manually changing the Jacobians. Hesch et al. [15] and Sun et al. [12] applied observability constraints to build robust and accurate VINS algorithms. However, the Jacobians used in these algorithms do not strictly follow first-order Taylor series expansion and are not theoretically optimal.

Instead of directly manipulating the linearization points or Jacobians, Bloesch et al. [6] and Huai et al. [11] proposed to use robocentric state representations, which keep the IMU states and the landmarks all in the local IMU frame to mitigate the VINS inconsistency. In this formulation, the system is consistent as it automatically preserves the unobservable subspaces without any Jacobian manipulations. However, extra states are required to be kept in the state vector, such as gravity [11]), which might cause extra unobservable directions.

In recent years, the manifold theory, especially Lie group [28] representation for robot pose, has been applied to robot navigation [18], [19], [23], [25], [29], [30], and shows to improve both estimation consistency and accuracy. Bonnabel and Barrau et al. [29], [31] first justified and applied an appropriate group $\mathbf{SE}_3(3)$ to enforce system observability for filter-based simultaneous localization and mapping (SLAM). Brossard et al. [21] showed that the IMU propagation with invariant error states can naturally describe the banana-shaped uncertainty distribution for extended pose represented in $\mathbf{SE}_2(3)$. Huai et al. [25] leveraged extended poses to design a consistent fixed-lag smoother which jointly estimates IMU poses and features. Wu et al. [19] proposed right invariant (RI)-VINS which uses right invariant errors for VINS states

Manuscript received: Sept., 9th, 2021; Revised Nov., 24th, 2021; Accepted Dec., 12th, 2021. This paper was recommended for publication by Editor Behnke, Sven upon evaluation of the Associate Editor and Reviewers' comments. This work was partially supported by the University of Delaware (UD) College of Engineering, the NSF (IIS-1924897), and the ARL (W911NF-19-2-0226, W911NF-20-2-0098, JWS 10-051-003).

The authors are with Robot Perception and Navigation Group, University of Delaware, Newark, DE 19716, USA. {yuyang,ccchu,woosik,ghuang}@udel.edu

Thanks Patrick Geneva for OpenVINS support and proofreading. Thanks Teng Zhang from CVTE and Kejian Wu from Nreal for helpful discussions.

Digital Object Identifier (DOI): see top of this page.

except for IMU biases. They theoretically showed that RI-VINS can directly satisfy the unobservable properties of VINS without Jacobian modifications. They further combined RI and MSCKF as RI-MSCKF, which demonstrates improved performance compared to std-EKF based MSCKF. Heo et al. [23] and Tsao et al. [24] also did observability analysis to show the nice properties of using invariant error states for algorithm design. They improved MSCKF based VINS with IMU state perturbed by RI and left invariant (LI) state errors, respectively. The above works have applied invariant error states for the VINS algorithms and shown to achieve better convergence and accuracy. However, the above filter-based algorithms do not keep any features in the state vector, while keeping features in the state, even temporally with limited qualities will substantially improve the system's accuracy and robustness for long-term navigation [5]. Also, their works were compared only with std-EKF based algorithms even though there are state-of-art and open-sourced VINS algorithms based on FEJ-EKF [14]. In addition, we find that the features will be involved in the state covariance propagation in RI-VINS if the features are associated with the current IMU state for the Lie group representation. This will lead to computation increases if more variables (i.e., more features) are included in the state vectors.

In order to leverage the nice consistent properties of the invariant VINS but also achieve comparable efficiency when keeping features in the state for better performance, we propose two novel algorithms that can decouple the features from the state covariance propagation while maintaining system consistency. One algorithm represents IMU state in Lie group and the features in $\mathbb{R}^3$. "FEJ" technique is applied to feature states to keep system consistent. The other algorithm is designed specifically for sliding-window based VINS, as it decouples the feature state from the current IMU state and associates it to an active cloned pose from the sliding window for the Lie group representation. Specifically, the main contributions of the this work can be listed as:

- We analytically derived right invariant error based VINS and found that the feature state is involved in the state covariance propagation if it is associated with the current IMU state for the Lie group representation.
- We propose decoupled right invariant (DRI)-VINS algorithms that decouple the feature from state covariance propagation while still keeping the system consistent.
- We perform extensive Monte-Carlo simulations with three simulated trajectories and different measurement noises to verify both consistency and accuracy of the proposed algorithms. Real world evaluations based on TUM-VI datasets also show improved performances of the proposed algorithms to state-of-art filter-based VINS.

II. VINS With Right Invariant Error States

In this section, we introduce the VINS algorithms with invariant state errors.

A. Definition of Left- And Right-Invariant Errors

If $\mathbf{y}$ and $\hat{\mathbf{y}}$ are both $n \times n$ matrices within the same matrix Lie group [28], the right- and left-invariant errors can be defined as [32]:

$$\delta\mathbf{y}^r = \hat{\mathbf{y}}\mathbf{y}^{-1} = (\hat{\mathbf{y}}\mathbf{L})(\mathbf{y}\mathbf{L})^{-1}, \quad \delta\mathbf{y}^l = \mathbf{y}^{-1}\hat{\mathbf{y}} = (\mathbf{L}\mathbf{y})^{-1}(\mathbf{L}\hat{\mathbf{y}})$$

where $\delta\mathbf{y}^l$ and $\delta\mathbf{y}^r$ denote left- and right-invariant errors, respectively; $\mathbf{L}$ is an arbitrary element of the same Lie group as $\mathbf{y}$.

B. State Vector

For simplicity, we assume the state vector contains the IMU navigation state $\mathbf{x}_n$, the bias state $\mathbf{x}_b$ and a single feature ${}^G\mathbf{p}_f$:

$$\mathbf{x} = \left(\mathbf{x}_n, \mathbf{x}_b, {}^G\mathbf{p}_f\right) \tag{1}$$

where $\mathbf{x}_n = \left({}^G_I\mathbf{R}, {}^G\mathbf{p}_I, {}^G\mathbf{v}_I\right)$ and $\mathbf{x}_b = (\mathbf{b}_g, \mathbf{b}_a)$; ${}^G\mathbf{p}_I$, ${}^G\mathbf{v}_I$ and ${}^G\mathbf{p}_f$ denote the IMU position, IMU velocity and feature position in the global frame $\{G\}$, respectively; ${}^G_I\mathbf{R} \in \mathbf{SO}(3)$ represents the rotation from IMU frame $\{I\}$ to $\{G\}$; $\mathbf{b}_g$ and $\mathbf{b}_a$ are both in $\mathbb{R}^3$ and represent the random walk for gyroscope and accelerometer, respectively. Following [19], [21], we define $\mathbf{x}_{nf}$ to include $\mathbf{x}_n$ and ${}^G\mathbf{p}_f$ in the Lie group as $\mathbf{SE}_3(3)$ [31] as:

$$\mathbf{x}_{nf} \triangleq (\mathbf{x}_n, {}^G\mathbf{p}_f) = \left[\begin{array}{c|cc} {}^G_I\mathbf{R} & {}^G\mathbf{p}_I & {}^G\mathbf{v}_I \; {}^G\mathbf{p}_f \\ \hline \mathbf{0}_3 & \multicolumn{2}{c}{\mathbf{I}_3} \end{array}\right] \tag{2}$$

Note $\mathbf{x}_{nf}$ can be perturbed with either left or right invariant error states. For clarity, we only present right invariant error based algorithms in this paper and our supplementary materials [33] cover both cases for readers. By defining $\hat{\mathbf{x}}$ as the estimate of $\mathbf{x}$ and $\delta\mathbf{x} = \begin{bmatrix}\delta\mathbf{x}_n^\top & \delta\mathbf{x}_b^\top & \delta\mathbf{p}_f^\top\end{bmatrix}^\top \in \mathbb{R}^{18}$ as the error of the estimation, we can define $\boxplus$ on manifold as:

$$\mathbf{x} = \hat{\mathbf{x}} \boxplus \delta\mathbf{x} = \left(\exp\left(\begin{bmatrix}\delta\mathbf{x}_n \\ \delta\mathbf{p}_f\end{bmatrix}\right) \cdot \hat{\mathbf{x}}_{nf}, \mathbf{x}_b + \delta\mathbf{x}_b\right) \tag{3}$$

$$= \begin{bmatrix} \exp(\delta\boldsymbol{\theta}_I){}^G_I\mathbf{R} \\ \exp(\delta\boldsymbol{\theta}_I){}^G\hat{\mathbf{p}}_I + \mathbf{J}_l(\delta\boldsymbol{\theta}_I)\delta\mathbf{p} \\ \exp(\delta\boldsymbol{\theta}_I){}^G\hat{\mathbf{v}}_I + \mathbf{J}_l(\delta\boldsymbol{\theta}_I)\delta\mathbf{v} \\ \hat{\mathbf{b}}_g + \delta\mathbf{b}_g \\ \hat{\mathbf{b}}_a + \delta\mathbf{b}_a \\ \exp(\delta\boldsymbol{\theta}_I){}^G\hat{\mathbf{p}}_f + \mathbf{J}_l(\delta\boldsymbol{\theta}_I)\delta\mathbf{p}_f \end{bmatrix} \tag{4}$$

where $\delta\mathbf{x}_n = \begin{bmatrix}\delta\boldsymbol{\theta}_I^\top & \delta\mathbf{p}^\top & \delta\mathbf{v}^\top\end{bmatrix}^\top \in \mathbb{R}^9$ denotes the right invariant error states for IMU navigation state $\mathbf{x}_n$ which includes the errors of orientation $\delta\boldsymbol{\theta}_I$, position $\delta\mathbf{p}$, and velocity $\delta\mathbf{v}$. $\delta\mathbf{x}_b = \begin{bmatrix}\delta\mathbf{b}_g^\top & \delta\mathbf{b}_a^\top\end{bmatrix}^\top \in \mathbb{R}^6$ denotes the IMU bias error states including $\delta\mathbf{b}_g$ and $\delta\mathbf{b}_a$. $\delta\mathbf{p}_f \in \mathbb{R}^3$ represents the feature error state; $\exp(\cdot)$ and $\mathbf{J}_l(\cdot)$ are matrix exponential and left Jacobians defined for Lie group [34], respectively. Note that $\begin{bmatrix}\delta\mathbf{x}_n^\top & \delta\mathbf{p}_f^\top\end{bmatrix}^\top$ represent the right invariant errors of $\mathbf{x}_{nf}$.

C. State Propagation

The linear acceleration $\mathbf{a}_m$ and angular velocity $\boldsymbol{\omega}_m$ readings from an IMU are:

$$\mathbf{a}_m = \mathbf{a} + \mathbf{b}_a + \mathbf{n}_a \tag{5}$$

$$\boldsymbol{\omega}_m = \boldsymbol{\omega} + \mathbf{b}_g + \mathbf{n}_g \tag{6}$$

where $\mathbf{n}_a$ and $\mathbf{n}_g$ are zero-mean Gaussian random noises. The system dynamic model is thus defined as [35]:

$$\begin{array}{lll} {}^G_I\dot{\mathbf{R}} = {}^G_I\mathbf{R}\lfloor{}^I\boldsymbol{\omega}\rfloor, & {}^G\dot{\mathbf{p}}_I = {}^G\mathbf{v}_I, & {}^G\dot{\mathbf{v}}_I = {}^G_I\mathbf{R}{}^I\mathbf{a} + {}^G\mathbf{g} \\ \dot{\mathbf{b}}_g = \mathbf{n}_{wg}, & \dot{\mathbf{b}}_a = \mathbf{n}_{wa}, & {}^G\dot{\mathbf{p}}_f = \mathbf{0}_{3\times1} \end{array} \tag{7}$$

where $\mathbf{n}_{wg}$ and $\mathbf{n}_{wa}$ are Gaussian noises driving the biases $\mathbf{b}_g$ and $\mathbf{b}_a$, respectively; ${}^G\mathbf{g} = [0 \; 0 \; -9.8]^\top$ is the gravity;

$\lfloor\cdot\rfloor$ denotes a skew-symmetric matrix [35]. By integrating the IMU measurements between time t_k and t_{k+1}, the consecutive system state $\mathbf{x}_{k+1}$ propagated from $\mathbf{x}_k$ is derived as:

$$
\begin{aligned}
{}^{G}_{I_{k+1}}\mathbf{R} &= {}^{G}_{I_k}\mathbf{R}\Delta\mathbf{R}_k && (8)\\
{}^{G}\mathbf{p}_{I_{k+1}} &= {}^{G}\mathbf{p}_{I_k} + {}^{G}\mathbf{v}_{I_k}\delta t + {}^{G}_{I_k}\mathbf{R}\Delta\mathbf{p}_k + \frac{1}{2}{}^{G}\mathbf{g}\delta t^2 && (9)\\
{}^{G}\mathbf{v}_{I_{k+1}} &= {}^{G}\mathbf{v}_{I_k} + {}^{G}_{I_k}\mathbf{R}\Delta\mathbf{v}_k + {}^{G}\mathbf{g}\delta t && (10)\\
\mathbf{b}_{g_{k+1}} &= \mathbf{b}_{g_k} + \int_{t_k}^{t_{k+1}} \mathbf{n}_{wg}dt && (11)\\
\mathbf{b}_{a_{k+1}} &= \mathbf{b}_{a_k} + \int_{t_k}^{t_{k+1}} \mathbf{n}_{wa}dt && (12)
\end{aligned}
$$

where $\delta t = t_{k+1} - t_k$; $\Delta\mathbf{R}_k = \exp\left(\int_{t_k}^{t_{k+1}} {}^{I_\tau}\boldsymbol{\omega}d\tau\right)$; $\Delta\mathbf{v}_k = \int_{t_k}^{t_{k+1}} {}^{I_k}_{I_\tau}\mathbf{R}\,{}^{I_\tau}\mathbf{a}d\tau$; $\Delta\mathbf{p}_k = \int_{t_k}^{t_{k+1}}\int_{t_k}^{s} {}^{I_k}_{I_\tau}\mathbf{R}\,{}^{I_\tau}\mathbf{a}d\tau ds$. Following [21], the state propagation is reorganized as:

$$
\begin{aligned}
\mathbf{x}_{nf_{k+1}} &= \boldsymbol{\Gamma}_k\boldsymbol{\Psi}(\mathbf{x}_{nf_k})\boldsymbol{\Upsilon}_k && (13)\\
\boldsymbol{\Gamma}_k &= \left[\begin{array}{c|ccc} \mathbf{I}_3 & \frac{1}{2}{}^{G}\mathbf{g}\delta t^2 & \mathbf{g}\delta t & \mathbf{0}_{3\times1} \\ \hline \mathbf{0}_3 & \multicolumn{3}{c}{\mathbf{I}_3} \end{array}\right] && (14)\\
\boldsymbol{\Psi}(\mathbf{x}_{nf_k}) &= \left[\begin{array}{c|cccc} {}^{G}_{I_k}\mathbf{R} & {}^{G}\mathbf{p}_{I_k} + {}^{G}\mathbf{v}_{I_k}\delta t & {}^{G}\mathbf{v}_{I_k} & {}^{G}\mathbf{p}_f \\ \hline \mathbf{0}_3 & \multicolumn{3}{c}{\mathbf{I}_3} \end{array}\right] && (15)\\
\boldsymbol{\Upsilon}_k &= \left[\begin{array}{c|ccc} \Delta\mathbf{R}_k & \Delta\mathbf{p}_k & \Delta\mathbf{v}_k & \mathbf{0}_{3\times1} \\ \hline \mathbf{0}_3 & \multicolumn{3}{c}{\mathbf{I}_3} \end{array}\right] && (16)
\end{aligned}
$$

Hence, the linearized system (detailed derivations can be found in [33]) from t_k to t_{k+1} is written as:

$$
\begin{aligned}
\delta\mathbf{x}_{k+1} &\simeq \boldsymbol{\Phi}_{k+1,k}\delta\mathbf{x}_k + \mathbf{G}_k\mathbf{n}_{dk} && (17)\\
\boldsymbol{\Phi}_{k+1,k} &= \begin{bmatrix} \boldsymbol{\Phi}_{nn} & \boldsymbol{\Phi}_{nb} & \mathbf{0}_{9\times3} \\ \mathbf{0}_{6\times9} & \mathbf{I}_6 & \mathbf{0}_{6\times3} \\ \mathbf{0}_{3\times9} & \boldsymbol{\Phi}_{fb} & \mathbf{I}_3 \end{bmatrix},\ \mathbf{G}_k = \begin{bmatrix} \mathbf{G}_{nn} & \mathbf{0}_{9\times6} \\ \mathbf{0}_6 & \mathbf{I}_6\delta t \\ \mathbf{G}_{fn} & \mathbf{0}_{3\times6} \end{bmatrix} && (18)\\
\boldsymbol{\Phi}_{nn} &= \begin{bmatrix} \mathbf{I}_3 & 0 & 0 \\ 0 & \mathbf{I}_3 & \delta t\mathbf{I}_3 \\ 0 & 0 & \mathbf{I}_3 \end{bmatrix}, \boldsymbol{\Phi}_{fb} = \begin{bmatrix} -\lfloor{}^{G}\hat{\mathbf{p}}_f\rfloor{}^{G}_{I_{k+1}}\hat{\mathbf{R}}\mathbf{J}_r(\Delta\hat{\boldsymbol{\theta}})\delta t & \mathbf{0}_3 \end{bmatrix}\\
\boldsymbol{\Phi}_{nb} &= \begin{bmatrix} -{}^{G}_{I_{k+1}}\hat{\mathbf{R}}\mathbf{J}_r(\Delta\hat{\boldsymbol{\theta}})\delta t & \mathbf{0}_3 \\ -\lfloor{}^{G}\hat{\mathbf{p}}_{I_{k+1}}\rfloor{}^{G}_{I_{k+1}}\hat{\mathbf{R}}\mathbf{J}_r(\Delta\hat{\boldsymbol{\theta}})\delta t + {}^{G}_{I_k}\hat{\mathbf{R}}\boldsymbol{\Xi}_4 & -{}^{G}_{I_k}\hat{\mathbf{R}}\boldsymbol{\Xi}_2 \\ -\lfloor{}^{G}\hat{\mathbf{v}}_{I_{k+1}}\rfloor{}^{G}_{I_{k+1}}\hat{\mathbf{R}}\mathbf{J}_r(\Delta\hat{\boldsymbol{\theta}})\delta t + {}^{G}_{I_k}\hat{\mathbf{R}}\boldsymbol{\Xi}_3 & -{}^{G}_{I_k}\hat{\mathbf{R}}\boldsymbol{\Xi}_1 \end{bmatrix}
\end{aligned}
$$

where $\boldsymbol{\Phi}_{k+1,k}$ denotes the state transition matrix; $\mathbf{G}_k$ is the measurement noise Jacobian; $\mathbf{n}_{dk}$ is discretized IMU noise; Note that $\boldsymbol{\Phi}_{fb} = \mathbf{G}_{fn}$ and $\boldsymbol{\Phi}_{nb} = \mathbf{G}_{nn}$; $\Delta\hat{\boldsymbol{\theta}} = \log\left(\Delta\hat{\mathbf{R}}_k\right)$; $\log(\cdot)$ and $\mathbf{J}_r(\cdot)$ denote the log matrix operation and right Jacobians for $\mathbf{SO}(3)$ [34]; $\boldsymbol{\Xi}_i, i = 1\ldots4$, are integration components defined in our previous work ACI^2 [35]. With (13) and (17), we have both mean and covariance propagation for Kalman filter.

D. Visual Measurements Update

When the camera is exploring the environment, we can track the point feature and get the corresponding normalized image pixel measurement as:

$$
\begin{aligned}
\mathbf{z} &= \begin{bmatrix} \frac{x}{z} & \frac{y}{z} \end{bmatrix}^\top, {}^{C}\mathbf{p}_f = \begin{bmatrix} x & y & z \end{bmatrix}^\top && (19)\\
{}^{C}\mathbf{p}_f &= {}^{C}_{I}\mathbf{R}^{I}_{G}\mathbf{R}({}^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_I) + {}^{C}\mathbf{p}_I && (20)
\end{aligned}
$$

With the definition of error states in (3), we can linearize the image measurement and compute the Jacobians $\mathbf{H} \triangleq \frac{\partial\delta\mathbf{z}}{\partial\delta\mathbf{x}}$ as:

$$
\begin{aligned}
\mathbf{H} &= \frac{\partial\delta\mathbf{z}}{\partial\delta{}^{C}\mathbf{p}_f}\begin{bmatrix} \frac{\partial\delta{}^{C}\mathbf{p}_f}{\partial\delta\boldsymbol{\theta}_I} & \frac{\partial\delta{}^{C}\mathbf{p}_f}{\partial\delta\mathbf{p}} & \frac{\partial\delta{}^{C}\mathbf{p}_f}{\partial\delta\mathbf{v}} & \frac{\partial\delta{}^{C}\mathbf{p}_f}{\partial\delta\mathbf{b}_g} & \frac{\partial\delta{}^{C}\mathbf{p}_f}{\partial\delta\mathbf{b}_a} & \frac{\partial\delta{}^{C}\mathbf{p}_f}{\partial\delta\mathbf{p}_f} \end{bmatrix}\\
&= \mathbf{H}_C\begin{bmatrix} \mathbf{0}_3 & -{}^{C}_{I}\hat{\mathbf{R}}{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & {}^{C}_{I}\hat{\mathbf{R}}{}^{I}_{G}\hat{\mathbf{R}} \end{bmatrix} && (21)
\end{aligned}
$$

where $\mathbf{H}_C$ represents the projection Jacobian. We can then perform EKF update to get refined state estimates and covariances. Now we have presented the algorithm for RI-VINS.

E. Observability Analysis

Observability indicates whether the system can recover its initial states with all the measurements. It can also be used for system consistency analysis [15], [16] and degenerate motion identification [36]–[41]. Following [15], the observability matrix for the linearized system is defined as:

$$
\mathcal{O} = \begin{bmatrix} \mathcal{O}_0 \\ \mathcal{O}_1 \\ \vdots \\ \mathcal{O}_k \end{bmatrix} = \begin{bmatrix} \mathbf{H}_0 \\ \mathbf{H}_1\boldsymbol{\Phi}_{1,0} \\ \vdots \\ \mathbf{H}_k\boldsymbol{\Phi}_{k,0} \end{bmatrix} \quad (22)
$$

Dropping the subscripts k for simplicity, each block row $\mathcal{O}_k$ is computed as:

$$
\mathcal{O}_k = \mathbf{H}_C{}^{C}_{I}\hat{\mathbf{R}}{}^{I}_{G}\hat{\mathbf{R}}\begin{bmatrix} \mathbf{M}_1 & -\mathbf{I}_3 & -\mathbf{I}_3\delta t & \mathbf{M}_2 & \mathbf{M}_3 & \mathbf{I}_3 \end{bmatrix} \quad (23)
$$

where:

$$
\begin{aligned}
\mathbf{M}_1 &= \frac{1}{2}\lfloor{}^{G}\mathbf{g}\rfloor\delta t^2, \quad \mathbf{M}_3 = {}^{G}_{I_0}\hat{\mathbf{R}}\boldsymbol{\Xi}_2 && (24)\\
\mathbf{M}_2 &= \lfloor{}^{G}\hat{\mathbf{p}}_{I_k}\rfloor{}^{G}_{I_k}\hat{\mathbf{R}}\mathbf{J}_r(\Delta\hat{\boldsymbol{\theta}})\delta t - {}^{G}_{I_0}\hat{\mathbf{R}}\boldsymbol{\Xi}_4 && (25)
\end{aligned}
$$

It can be easily found that there are still 4 unobservable directions for RI-VINS, s.t., $\mathcal{O}\mathbf{N} = \mathbf{0}$ with unobservable subspace $\mathbf{N}$:

$$
\mathbf{N} = \begin{bmatrix} {}^{G}\mathbf{g}^\top & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times9} & \mathbf{0}_{1\times3} \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_{3\times9} & \mathbf{I}_3 \end{bmatrix}^\top \quad (26)
$$

Different from std-EKF based VINS [5], [15], [27], the null space of the RI-VINS is unrelated to the state vector. Hence, the system unobservable subspaces will not be affected by system linearization change. This property makes RI-EKF automatically avoid inconsistent issues caused by spurious information inflation to system unobservable directions [15], [16].

However, the drawback for the RI-VINS is that the feature is associated with current IMU state $\mathbf{x}_n$ for the Lie Group representation and involved in the state covariance propagation (as $\boldsymbol{\Phi}_{fb}$ and $\mathbf{G}_{fn}$ shown in (18), which would normally be $\mathbf{0}$ in std-EKF based VINS [5], [15], [27]). If there are l features in the state vector, the computation cost of covariance propagation for std-EKF based VINS is only around $\mathbf{O}(l)$ but can rise to $\mathbf{O}(l^2)$ for RI-VINS (see complementary materials [33]). This slows down the system propagation dramatically if more variables are involved in the state vector and will limit the application of RI-VINS.

In order to keep the nice property of the right invariant error states but also consistently incorporate more state variables (especially features), the most straight-forward solution is to decouple the feature from the state covariance propagation. In the following sections, we propose two new algorithms to achieve this goal, named as decoupled right invariant (DRI)-VINS.

III. DRI-FEJ For VINS

If the features are decoupled from the Lie group representation with the current IMU state and parameterized in $\mathbb{R}^3$ space naively, named DRI-NAIVE, the system becomes inconsistent because the system unobservable subspaces will be affected by the feature states. We hence propose DRI-FEJ to combine DRI and FEJ, which leverages right invariant errors and performs "FEJ" to the feature state ${}^G\mathbf{p}_f$ to achieve consistent performance.

A. DRI-NAIVE

If the feature is not associated with the Lie group of the current IMU state but represented in $\mathbb{R}^3$ naively, the $\boxplus$ in manifold is re-defined as:

$$\mathbf{x} = \hat{\mathbf{x}} \boxplus \delta\mathbf{x} = \left(\exp\left(\delta\mathbf{x}_n\right)\hat{\mathbf{x}}_n, \mathbf{x}_b + \delta\mathbf{x}_b, {}^G\mathbf{p}_f + \delta\mathbf{p}_f\right) \quad (27)$$

$$= \begin{bmatrix} \exp(\delta\boldsymbol{\theta}_I){}^G_I\mathbf{R} \\ \exp(\delta\boldsymbol{\theta}_I){}^G\hat{\mathbf{p}}_I + \mathbf{J}_l(\delta\boldsymbol{\theta}_I)\delta\mathbf{p} \\ \exp(\delta\boldsymbol{\theta}_I){}^G\hat{\mathbf{v}}_I + \mathbf{J}_l(\delta\boldsymbol{\theta}_I)\delta\mathbf{v} \\ \hat{\mathbf{b}}_g + \delta\mathbf{b}_g \\ \hat{\mathbf{b}}_a + \delta\mathbf{b}_a \\ {}^G\hat{\mathbf{p}}_f + \delta\mathbf{p}_f \end{bmatrix} \quad (28)$$

Following similar procedures, the state transition $\boldsymbol{\Phi}_{k+1,k}$ and measurement Jacobians are computed as:

$$\boldsymbol{\Phi}_{k+1,k} = \begin{bmatrix} \boldsymbol{\Phi}_{nn} & \boldsymbol{\Phi}_{nb} & \mathbf{0}_{9\times3} \\ \mathbf{0}_{6\times9} & \mathbf{I}_6 & \mathbf{0}_{6\times3} \\ \mathbf{0}_{3\times9} & \mathbf{0}_{3\times6} & \mathbf{I}_3 \end{bmatrix} \quad (29)$$

$$\begin{aligned} \mathbf{H} &= \mathbf{H}_C \begin{bmatrix} \frac{\partial\delta^C\mathbf{p}_f}{\partial\delta\boldsymbol{\theta}_I} & \frac{\partial\delta^C\mathbf{p}_f}{\partial\delta\mathbf{p}} & \frac{\partial\delta^C\mathbf{p}_f}{\partial\delta\mathbf{v}} & \frac{\partial\delta^C\mathbf{p}_f}{\partial\delta\mathbf{b}_g} & \frac{\partial\delta^C\mathbf{p}_f}{\partial\delta\mathbf{b}_a} & \frac{\partial\delta^C\mathbf{p}_f}{\partial\delta\mathbf{p}_f} \end{bmatrix} \\ &= \mathbf{H}_C {}^C_I\hat{\mathbf{R}}{}^I_G\hat{\mathbf{R}} \begin{bmatrix} \lfloor {}^G\hat{\mathbf{p}}_f \rfloor & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \end{aligned} \quad (30)$$

It can be observed that the feature is decoupled from the state covariance propagation as expected, as $\boldsymbol{\Phi}_{fb} = \mathbf{0}_{3\times6}$. We then compute the observability matrix $\mathcal{O}$ and get the new unobservable subspace for DRI-NAIVE as:

$$\mathbf{N} = \begin{bmatrix} {}^G\mathbf{g}^\top & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times9} & -(\lfloor {}^G\hat{\mathbf{p}}_f \rfloor {}^G\mathbf{g})^\top \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_{3\times9} & \mathbf{I}_3 \end{bmatrix}^\top \quad (31)$$

It shows that, different from RI-VINS of which the unobservable subspace is invariant of state estimates, the unobservable subspace of DRI-NAIVE VINS can be affected by feature estimate ${}^G\hat{\mathbf{p}}_f$. That means if we naively decouple the features, the system will suffer from inconsistency caused by the feature linearization change. This claim is further verified by Monte-Carlo simulations in Section V.

B. DRI-FEJ

For tackling this inconsistency issue for DRI-NAIVE, the most straight forward solution is to perform "FEJ" technique when computing the feature state Jacobians, which avoids the linearization point change. This proposed DRI-FEJ algorithm will keep the four unobservable directions of VINS by simply fixing the linearization points of feature states. Unlike the "FEJ" based VINS algorithms in [5], [14], which need to perform "FEJ" to all the IMU states and feature states, the proposed DRI-FEJ only uses FEJ for the feature states. Therefore, in the proposed algorithm, the system can always utilize the current best IMU state estimates for system linearization. The side effects of bad state initialization can be mitigated and DRI-FEJ can achieve better accuracy than FEJ (see Section V). In addition, the invariant error states can also better capture the uncertainty of the extended pose [21], and benefit the system performance.

IV. DRI with Sliding Window For VINS

DRI-FEJ is a general algorithm for EKF, and we also propose a DRI specifically for sliding-window based estimators which are widely used for VINS algorithms. In this section, we find that the feature can be decoupled from the state covariance propagation by associating features with an active cloned pose within the sliding window for Lie group representation. We further extend RI-MSCKF [19] to keep the features in the state longer than the sliding window span for better accuracy and robustness [5]. We also, for the first time, propose a pseudo-anchor change algorithm, which represents the features in global frame but changes their associating Lie group pose which might be marginalized from the sliding window, thus the features can be kept in the state vector as long as they can still be tracked actively in current frame.

A. DRI-Sliding Window (SW)

The state vector contains the IMU navigation state $\mathbf{x}_n$, the bias state $\mathbf{x}_b$, one feature point ${}^G\mathbf{p}_f$ and sliding window states $\mathbf{x}_c$:

$$\mathbf{x} = \left(\mathbf{x}_n, \mathbf{x}_b, \mathbf{x}_c, {}^G\mathbf{p}_f\right) \quad (32)$$

$$\mathbf{x}_c = \left({}^G_{I_{c0}}\mathbf{R}, {}^G\mathbf{p}_{I_{c0}}, {}^G_{I_{c1}}\mathbf{R}, {}^G\mathbf{p}_{I_{c1}}\right) \quad (33)$$

Note that we only consider two poses in the sliding window for simplicity. Different from DRI-FEJ, we associate the feature ${}^G\mathbf{p}_f$ with the first clone pose $\{I_{c0}\}$ for the Lie group representation. Thus, the new error states $\delta\mathbf{x} \in \mathbb{R}^{30}$ can be defined as:

$$\delta\mathbf{x} = \begin{bmatrix} \delta\mathbf{x}_n^\top & \delta\mathbf{x}_b^\top & \delta\mathbf{x}_c^\top & \delta\mathbf{p}_f^\top \end{bmatrix}^\top \quad (34)$$

$$\delta\mathbf{x}_c = \begin{bmatrix} \delta\boldsymbol{\theta}_{I_{c0}}^\top & \delta\mathbf{p}_{I_{c0}}^\top & \delta\boldsymbol{\theta}_{I_{c1}}^\top & \delta\mathbf{p}_{I_{c1}}^\top \end{bmatrix}^\top \quad (35)$$

where $\delta\mathbf{x}_c$ represents the errors states for sliding window states $\mathbf{x}_c$. Therefore, the new $\boxplus$ for DRI-SW is defined as:

$$\mathbf{x} = \hat{\mathbf{x}} \boxplus \delta\mathbf{x} = \begin{bmatrix} \exp(\delta\boldsymbol{\theta}_I){}^G_I\mathbf{R} \\ \exp(\delta\boldsymbol{\theta}_I){}^G\hat{\mathbf{p}}_I + \mathbf{J}_l(\delta\boldsymbol{\theta}_I)\delta\mathbf{p} \\ \exp(\delta\boldsymbol{\theta}_I){}^G\hat{\mathbf{v}}_I + \mathbf{J}_l(\delta\boldsymbol{\theta}_I)\delta\mathbf{v} \\ \hat{\mathbf{b}}_g + \delta\mathbf{b}_g \\ \hat{\mathbf{b}}_a + \delta\mathbf{b}_a \\ \exp(\delta\boldsymbol{\theta}_{I_{c0}}){}^G_{I_{c0}}\mathbf{R} \\ \exp(\delta\boldsymbol{\theta}_{I_{c0}}){}^G\hat{\mathbf{p}}_{I_{c0}} + \mathbf{J}_l(\delta\boldsymbol{\theta}_{I_{c0}})\delta\mathbf{p}_{I_{c0}} \\ \exp(\delta\boldsymbol{\theta}_{I_{c1}}){}^G_{I_{c1}}\mathbf{R} \\ \exp(\delta\boldsymbol{\theta}_{I_{c1}}){}^G\hat{\mathbf{p}}_{I_{c1}} + \mathbf{J}_l(\delta\boldsymbol{\theta}_{I_{c1}})\delta\mathbf{p}_{I_{c1}} \\ \exp(\delta\boldsymbol{\theta}_{I_{c0}}){}^G\hat{\mathbf{p}}_f + \mathbf{J}_l(\delta\boldsymbol{\theta}_{I_{c0}})\delta\mathbf{p}_f \end{bmatrix} \quad (36)$$

B. State Transition Matrix And Measurement Jacobians

The state transition matrix for DRI-SW is derived as:

$$\boldsymbol{\Phi}_{k+1,k} = \begin{bmatrix} \boldsymbol{\Phi}_{nn} & \boldsymbol{\Phi}_{nb} & \mathbf{0}_{9\times12} & \mathbf{0}_{9\times3} \\ \mathbf{0}_{6\times9} & \mathbf{I}_6 & \mathbf{0}_{6\times12} & \mathbf{0}_{6\times3} \\ \mathbf{0}_{12\times9} & \mathbf{0}_{12\times6} & \mathbf{I}_{12} & \mathbf{0}_{12\times3} \\ \mathbf{0}_{3\times9} & \mathbf{0}_{3\times6} & \mathbf{0}_{3\times12} & \mathbf{I}_3 \end{bmatrix} \quad (37)$$

Clearly, the feature is decoupled from the IMU state covariance propagation with $\mathbf{\Phi}_{fb} = \mathbf{0}_{3\times 6}$ (see Eq. (18)). If feature is observed at a cloned pose $\{I_{ci}\}, i = \{0, 1\}$, the frame transformation for the feature is:

$$^{C}\mathbf{p}_f = {}^{C}_{I}\mathbf{R}\,{}^{I_{ci}}_{G}\mathbf{R}(^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_{I_{ci}}) + {}^{C}\mathbf{p}_I \tag{38}$$

Hence, we can recompute the Jacobians with feature associated with $\{I_{c0}\}$ for the Lie group representation as:

$$\frac{\partial \delta^{C}\mathbf{p}_f}{\partial \delta\boldsymbol{\theta}_{I_{c0}}} = -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I_{ci}}_{G}\hat{\mathbf{R}}\lfloor {}^{G}\hat{\mathbf{p}}_f \rfloor, \quad \frac{\partial \delta^{C}\mathbf{p}_f}{\partial \delta\boldsymbol{\theta}_{I_{c1}}} = -\frac{\partial \delta^{C}\mathbf{p}_f}{\partial \delta\boldsymbol{\theta}_{I_{c0}}}$$
$$\frac{\partial \delta^{C}\mathbf{p}_f}{\partial \delta\mathbf{p}_{I_{ci}}} = -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I_{ci}}_{G}\hat{\mathbf{R}}, \quad \frac{\partial \delta^{C}\mathbf{p}_f}{\partial \delta\mathbf{p}_f} = {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I_{ci}}_{G}\hat{\mathbf{R}}$$

C. Pseudo-Anchor Change

Different from frame anchor change in [14] which represents the features in the local sensor frame $\{I\}$ instead of the global frame $\{\mathrm{G}\}$, the proposed pseudo-anchor change refers to changing the associated Lie group representation for the features in global frame $\{G\}$. In order to keep the long-tracked features in the state vector even when the associated cloned pose is marginalized from the sliding window, we can perform pseudo-anchor change, that is to "anchor" the feature to another active cloned pose in the sliding window for the Lie group representation.

Since the feature state estimate ${}^{G}\hat{\mathbf{p}}_f$ remains the same, we only need to modify the covariance of the feature after pseudo-anchor change. If the associated pose I_{c0} of the feature is about to be marginalized from the sliding window, we then need to associate the feature with the pose I_{c1} for the Lie group representation. If we define $\delta\mathbf{p}_{fi}$ as the error states for ${}^{G}\mathbf{p}_f$ when it is associated with cloned pose $\{I_{ci}\}$, we have the following perturbations:

$$^{G}\mathbf{p}_f = \exp(\delta\boldsymbol{\theta}_{I_{c0}})\,{}^{G}\hat{\mathbf{p}}_f + \mathbf{J}_l(\delta\boldsymbol{\theta}_{I_{c0}})\delta\mathbf{p}_{f0} \tag{39}$$
$$^{G}\mathbf{p}_f = \exp(\delta\boldsymbol{\theta}_{I_{c1}})\,{}^{G}\hat{\mathbf{p}}_f + \mathbf{J}_l(\delta\boldsymbol{\theta}_{I_{c1}})\delta\mathbf{p}_{f1} \tag{40}$$

Then the feature's new error state $\delta\mathbf{p}_{f1}$ is written as:

$$\delta\mathbf{p}_{f1} = -\lfloor {}^{G}\hat{\mathbf{p}}_f \rfloor \delta\boldsymbol{\theta}_{I_{c0}} + \lfloor {}^{G}\hat{\mathbf{p}}_f \rfloor \delta\boldsymbol{\theta}_{I_{c1}} + \delta\mathbf{p}_{f0} \tag{41}$$

We then leverage EKF update to get the new state covariance with (41). The proposed pseudo-anchor change algorithm will not affect the system's unobservable property. More detailed discussion can be found in supplementary materials [33].

V. Monte-Carlo Simulations

In order to verify our proposed algorithms, we leverage the simulator in [14], [42] to simulate inertial readings and camera pixel measurements with 3 different trajectories (sim_rpng, sine_3d, and udel_gore, shown in Fig. 1). These 3 trajectories can cover most motion profiles that might appear in real world environments. The basic configuration for the simulation is listed in Table I.

TABLE I: Monte-Carlo Simulation Parameters

Parameter	Value	Parameter	Value
IMU Freq. (hz)	400	Max Cam Pts/Frame	100
Cam Freq. (hz)	10	Max Feats	40
Gyro White Noise	1.6968e-04	Gyro Random Walk	1.9393e-04
Accel. White Noise	2.0000e-03	Accel. Random Walk	3.0000e-03
Max. Clone Size	11	Pt Feat. Rep.	GLOBAL

TABLE II: Average absolute trajectory error (ATE) for 50 Monte-Carlo runs on 3 simulated trajectories (sim_rpng, sine_3d and udel_gore) with 1- and 3-pixel camera noises.

Noise	Algorithms	sim_rpng (degree/m)	sine_3d (degree/m)	udel_gore (degree/m)	Average (degree/m)
1 pixel	**dli_fej** [33]	0.268 / **0.155**	**0.538 / 0.127**	**0.351** / 0.160	**0.386 / 0.148**
	dri_fej	0.268 / 0.156	**0.538 / 0.127**	0.358 / 0.160	0.388 / **0.148**
	dri_sw	**0.267** / 0.157	0.539 / **0.127**	0.382 / **0.159**	0.396 / **0.148**
	fej [14]	0.368 / 0.207	0.575 / 0.130	0.389 / 0.163	0.444 / 0.167
	dli_naive	0.517 / 0.291	1.855 / 0.238	0.752 / 0.192	1.042 / 0.240
	dri_naive	0.553 / 0.305	1.985 / 0.238	0.769 / 0.190	1.102 / 0.244
	std	0.514 / 0.288	1.863 / 0.236	0.759 / 0.189	1.045 / 0.237
3 pixels	**dli_fej** [33]	0.567 / 0.349	**1.369** / 0.303	1.039 / 0.421	0.991 / 0.357
	dri_fej	0.574 / 0.353	1.409 / 0.305	0.995 / 0.418	0.993 / 0.359
	dri_sw	**0.507 / 0.328**	**1.369 / 0.302**	**0.860 / 0.392**	**0.912 / 0.341**
	fej [14]	0.639 / 0.384	1.417 / 0.305	1.064 / 0.420	1.040 / 0.370
	dli_naive	0.859 / 0.506	4.271 / 0.541	1.724 / 0.484	2.285 / 0.510
	dri_naive	0.855 / 0.496	4.255 / 0.521	1.654 / 0.468	2.255 / 0.495
	std	0.875 / 0.509	4.269 / 0.525	1.691 / 0.472	2.278 / 0.502

TABLE III: Average normalized estimation error-squared (NEES) for 50 Monte-Carlo runs of sim_rpng trajectory with 1- and 3-pixel camera noises.

Algorithm	NEES with 1-Pixel Noise (Orientation / Position)	NEES with 3-Pixel Noise (Orientation / Position)
dli_fej [33]	2.317 / 2.200	2.736 / 3.092
dri_fej	2.317 / 2.956	2.750 / 3.671
dri_sw	2.302 / 2.994	2.559 / 3.545
fej [14]	2.774 / 2.587	3.122 / 3.363
dli_naive	32.693 / 25.467	33.053 / 22.901
dri_naive	37.046 / 3.596	32.720 / 5.253
std	32.209 / 24.674	34.155 / 22.546

We leverage OpenVINS [14] to build our decoupled invariant error states based VINS system. In the system, the visual features are classified into two categories: 1) MSCKF features which are tracked within the sliding window and then marginalized; 2) SLAM features which are kept in the state vector until we lose tracks of them. We implement DRI-FEJ (dri_fej) and DRI-SW (dri_sw) introduced on Section III and IV. For completeness, we also implement decoupled left invariant(DLI)-FEJ (dli_fej) algorithm and detailed derivations can be found in supplementary materials [33]. In addition, the DRI-NAIVE (dri_naive) and DLI-NAIVE (dli_naive) algorithms, which both decouple features from the Lie group of the current IMU state and represent features in $\mathbb{R}^3$ naively, are also implemented and evaluated. For Monte-Carlo simulations, we perform online spatial-temporal and camera intrinsic calibration. We also compare our results with VINS based on std-EKF (std) and state-of-art FEJ-EKF (fej) both implemented in [14]. In order to test the robustness of the proposed algorithm to noises, we run 50 Monte-Carlo runs with all the above mentioned algorithms using 1-pixel and 3-pixel camera measurement noises. Root mean square error (RMSE), absolute trajectory error (ATE) and relative pose error (RPE) are used for trajectory accuracy evaluation while normalized estimation error squared (NEES) for system consistency evaluation.

The overall results are shown in Table II and III. From the results, we can see the dri_naive and dli_naive (similar to std) are inconsistent as their NEES values grow unbounded (see Table III and Fig. 3). Their performances are much worse

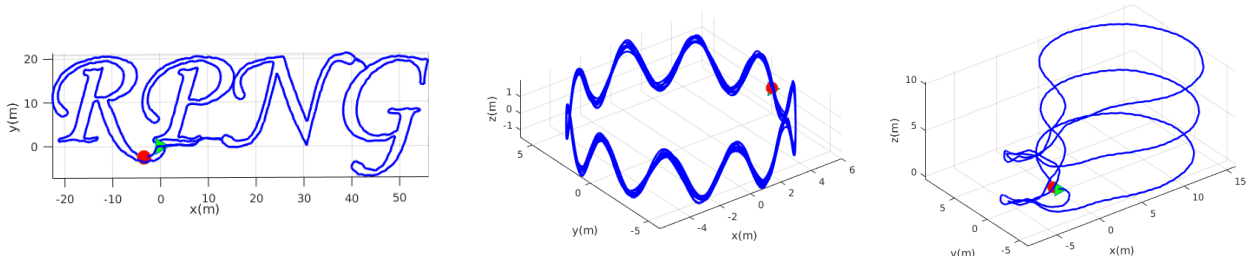

Fig. 1: Simulated trajectories. Left: sim_rpng (around 659m), middle: sine_3d (around 697m) and right: udel_gore (around 225m). Note that the green triangular denotes the start of the trajectory while the red circle indicates the end.

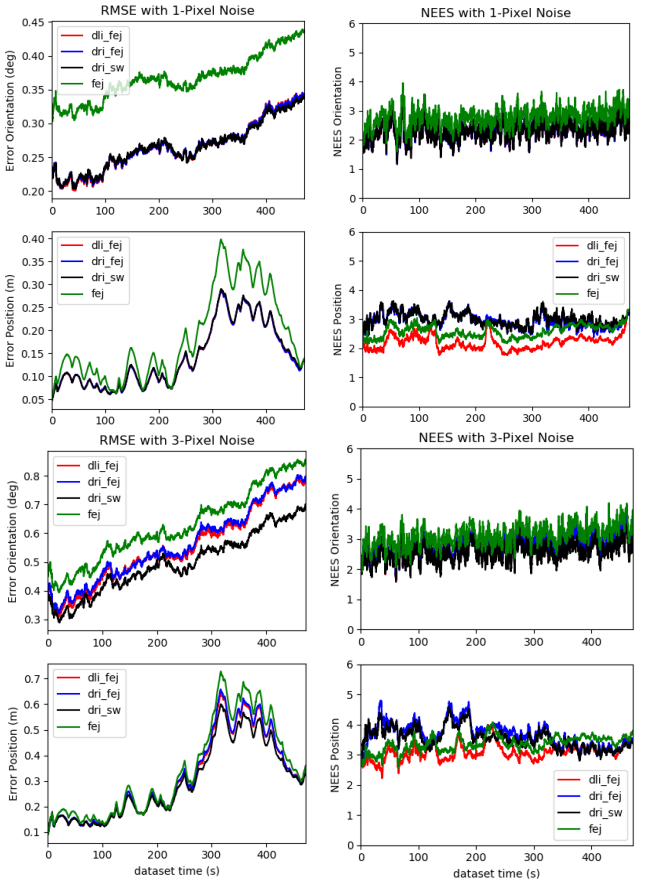

Fig. 2: Average RMSE and NEES for 50 Monte-Carlo runs of algorithms (dli_fej, dri_fej, dri_sw, and fej) with sim_rpng trajectory with 1-pixel and 3-pixel noises, receptively. The proposed dli_fej, dri_fej, and dri_sw achieve smaller RMSE than fej. As noise increases, all the algorithms remain consistent with NEES near 3.

than those consistent estimators we proposed, like dri_fej, dri_sw and dli_fej, of which the NEES values are around or smaller than 3, for both orientation and position estimates. No matter using 1-pixel or 3-pixel noises, all the proposed invariant error based algorithms (dri_fej, dri_sw and dli_fej) can output smaller ATE and RMSE than fej (see Fig. 2 for sim_rpng and Table II for all three trajectories). Note that when the noise is increased from 1 pixel to 3 pixels, the performance of dri_fei and dli_fej is approaching fej while dri_sw is still performing the best. This is because dri_sw always uses the best linearization points for the system Jacobians, while

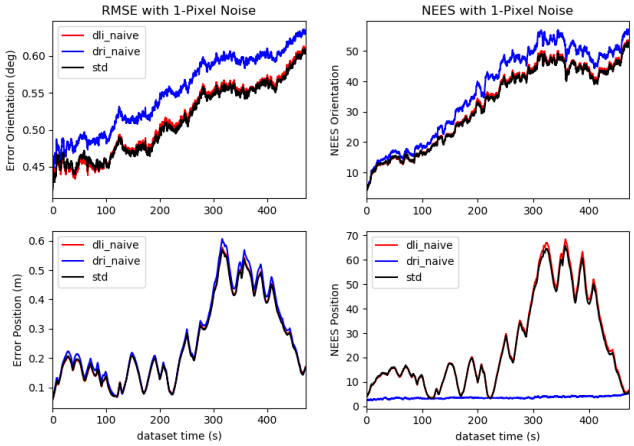

Fig. 3: Average RMSE and NEES for 50 Monte-Carlo runs of algorithms (dri_naive, dli_naive, and std) with sim_rpng trajectory at 1-pixel noise. dri_naive and dli_naive are inconsistent because the unobservable subspace can be affected by state estimate changes.

dri_fej, dli_fej and fej, which all need to perform FEJ to some states, will eventually affected by the bad state initialization caused by larger noises.

VI. Real-World Experiments

We further evaluate our proposed algorithms with real-world TUM-VI benchmark [43], which provides gray-scale stereo images at 20Hz, a time-synchronized 200Hz IMU and accurate pose ground-truth from a motion capture system. We run each of the above mentioned algorithms (dli_fej, dri_fej, dri_sw, fej, dli_naive, dri_naive and std) 5 times on TUM-VI Room sequences (6 datasets in total). For each run, we keep 11 clones and track at most 50 SLAM features in the state vector. Some visual features are treated as MSCKF features and marginalized when we lose tracking of them. We also perform spatial-temporal and camera intrinsic calibration during evaluations. The results for all these algorithms can be seen in Fig. 4 for average RPE and Table IV for average ATE.

It can be seen that the naive DRI algorithms (dri_naive and dli_naive) perform similar to std but much worse than those consistent algorithms (dli_fej, dri_fej, dri_sw and fej). This verifies that naively decoupling the features from the Lie group causes system to become inconsistent and degrades VINS performances, while our proposed algorithms (dli_fej, dri_fej and dri_sw) can achieve better or comparable accuracy to the state-of-art fej-based algorithm [14], especially for orientation

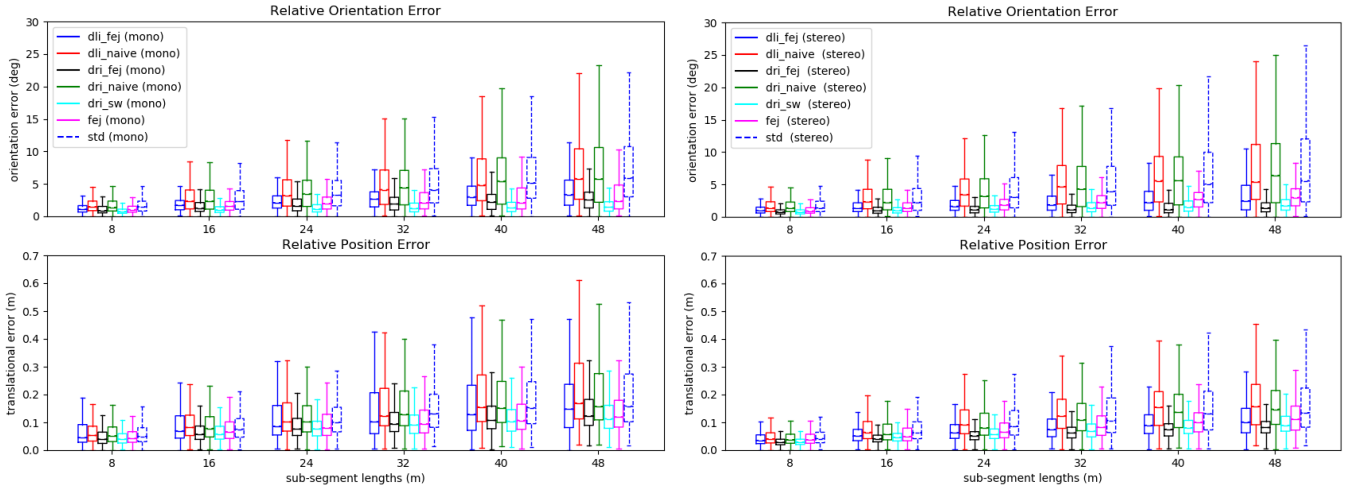

Fig. 4: Average RPE (orientation and position) plots for 5 runs with different algorithms on TUV-VI datasets Room sequences. In both monocular and stereo cases, the proposed dri_fej and dri_sw are better than fej.

TABLE IV: Average trajectory (orientation/position) error (ATE) for 5 runs with different algorithms on TUM-VI datasets Room sequences. The proposed dri_fej and dri_sw demonstrate improved accuracy than fej while dli_fej is similar to fej.

Camera	Algorithm	dataset-room1 (degree/m)	dataset-room2 (degree/m)	dataset-room3 (degree/m)	dataset-room4 (degree/m)	dataset-room5 (degree/m)	dataset-room6 (degree/m)	Average (degree/m)
Mono	**dli_fej** [33]	2.017 / 0.127	5.331 / 0.235	3.667 / 0.142	1.857 / 0.440	4.207 / 0.283	1.868 / 0.137	3.158 / 0.227
	dri_fej	1.860 / 0.147	2.654 / 0.199	1.998 / **0.114**	2.330 / 0.194	2.328 / **0.201**	3.084 / 0.262	2.376 / 0.186
	dri_sw	**1.557** / 0.143	**1.619** / **0.198**	**1.750** / 0.121	1.576 / 0.197	**1.306** / 0.211	**1.377** / 0.230	**1.531** / 0.183
	fej [14]	2.496 / **0.122**	4.435 / 0.217	2.242 / 0.117	**1.513** / **0.162**	1.549 / 0.236	3.939 / **0.126**	2.696 / **0.163**
	dli_naive	4.070 / 0.153	13.933 / 0.314	7.084 / 0.185	2.770 / 0.226	2.716 / 0.193	6.228 / 0.272	6.134 / 0.224
	dri_naive	3.391 / 0.155	14.471 / 0.319	7.055 / 0.169	3.114 / 0.213	1.835 / 0.227	5.763 / 0.210	5.938 / 0.215
	std	3.428 / 0.156	14.893 / 0.323	6.493 / 0.152	2.556 / 0.138	3.265 / 0.177	6.204 / 0.232	6.140 / 0.197
Stereo	**dli_fej** [33]	1.480 / 0.063	9.903 / 0.229	1.775 / 0.083	1.133 / 0.059	2.162 / 0.093	**0.732** / **0.057**	2.864 / 0.097
	dri_fej	**1.273** / **0.058**	1.397 / **0.094**	1.273 / 0.077	**0.861** / **0.039**	**1.411** / **0.089**	1.460 / 0.081	**1.279** / **0.073**
	dri_sw	1.596 / 0.061	**1.279** / 0.103	**1.260** / 0.082	0.916 / 0.046	1.683 / 0.092	2.465 / 0.078	1.533 / 0.077
	fej [14]	2.519 / 0.071	9.378 / 0.194	1.749 / **0.076**	1.103 / 0.065	1.812 / 0.098	1.578 / 0.075	3.023 / 0.097
	dli_naive	2.361 / 0.066	14.554 / 0.267	6.447 / 0.139	3.828 / 0.083	2.547 / 0.119	7.855 / 0.139	6.265 / 0.135
	dri_naive	3.479 / 0.079	14.482 / 0.267	7.268 / 0.146	3.188 / 0.063	1.622 / 0.108	6.130 / 0.102	6.028 / 0.127
	std	2.402 / 0.069	15.425 / 0.277	7.559 / 0.147	3.117 / 0.068	2.168 / 0.090	7.186 / 0.121	6.310 / 0.129

estimation. From Table IV, it can be seen that dri_fej and dri_sw generate smaller orientation ATE and comparable position ATE than fej for monocular case, and they perform better than fej both in orientation and position ATE for stereo cases. The advantages of dri_fej and dri_sw can be observed more clearly from the RPE plots (Fig. 4), from which it can be seen that both orientation and position errors are improved.

For both monocular and stereo cases, the proposed dri_fej and dri_sw have similar performances. But they achieve better accuracy than dli_fej and fej, this might be due to the fact that we only need to perform FEJ to the feature states in dri_fej or no FEJ is needed for dri_sw. In the meanwhile, dli_fej and fej both need to perform FEJ to all the sliding window states and feature states (see supplementary materials [33]), which makes them more vulnerable to bad state initialization caused by larger noises or bad feature triangulation.

VII. Conclusions and Future Work

In this paper we investigate the observability of VINS built on right invariant error states and find that invariant EKF based VINS can automatically satisfy the system unobservable properties. However, compared with the standard VINS algorithms, if the features are maintained in the state vector, the system suffers from increased computation costs because these feature states are involved in the covariance propagation. Instead of naively decoupling the feature state from the Lie group, we proposed two algorithms to decouple the feature from state propagation consistently. The first algorithm represents the feature in $\mathbb{R}^3$ and performs FEJ only to feature states for consistent performance. The second algorithm is designed specifically for sliding-window based VINS, which associates the feature states with a cloned pose in the sliding window, instead of the current IMU state, for the Lie group representation. We propose a pseudo-anchor change algorithm to maintain the feature in the state vector longer than window span. We implement both right- and left-invariant error based algorithms for a complete comparison. In Monte-Carlo simulations, we evaluate three trajectories with different measurement noises and show our proposed algorithms remain consistent and perform better than a state-of-art filter-based VINS algorithm using FEJ. Real world evaluations on the TUM-VI datasets with both monocular and stereo cameras are also provided to further verify our proposed algorithms. In the future, we will apply observability constrained (OC)-EKF to design DRI-OC algorithm. We will also investigate the performances of DRI-VINS algorithms under degenerate motions [40], [41].

REFERENCES

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

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

[3] 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.

[4] E. S. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *International Journal of Robotics Research*, vol. 30, no. 4, pp. 407–430, Apr. 2011.

[5] 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.

[6] 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.

[7] 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, 2020.

[8] K. Eckenhoff, P. Geneva, and G. Huang, "Mimc-vins: A versatile and resilient multi-imu multi-camera visual-inertial navigation system," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1360–1380, 2021.

[9] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, pp. 1–17, 2021.

[10] 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, pp. 1–8.

[11] Z. Huai and G. Huang, "Robocentric visual-inertial odometry," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Madrid, Spain, Oct. 1-5, 2018.

[12] K. Sun, K. Mohta, B. Pfrommer, M. Watterson, S. Liu, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, "Robust stereo visual inertial odometry for fast autonomous flight," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 965–972, April 2018.

[13] X. Fei and S. Soatto, "Xivo: An open-source software for visual-inertial odometry," https://github.com/ucla-vision/xivo, 2019.

[14] 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

[15] J. Hesch, D. Kottas, S. Bowman, and S. Roumeliotis, "Consistency analysis and improvement of vision-aided inertial navigation," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 158–176, 2013.

[16] G. Huang, A. I. Mourikis, and S. I. Roumeliotis, "Observability-based rules for designing consistent EKF SLAM estimators," *International Journal of Robotics Research*, vol. 29, no. 5, pp. 502–528, Apr. 2010.

[17] ——, "A first-estimates Jacobian EKF for improving SLAM consistency," in *Proc. of the 11th International Symposium on Experimental Robotics*, Athens, Greece, July 14–17, 2008.

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

[19] K. Wu, T. Zhang, D. Su, S. Huang, and G. Dissanayake, "An invariant-ekf vins algorithm for improving consistency," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2017, pp. 1578–1585.

[20] M. Brossard, S. Bonnabel, and A. Barrau, "Unscented kalman filter on lie groups for visual inertial odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 649–655.

[21] M. Brossard, A. Barrau, P. Chauchat, and S. Bonnabel, "Associating uncertainty to extended poses for on lie group imu preintegration with rotating earth," *IEEE Transactions on Robotics*, pp. 1–18, 2021.

[22] A. Barrau and S. Bonnabel, "A mathematical framework for imu error propagation with applications to preintegration," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 5732–5738.

[23] S. Heo and C. G. Park, "Consistent ekf-based visual-inertial odometry on matrix lie group," *IEEE Sensors Journal*, vol. 18, no. 9, pp. 3780–3788, 2018.

[24] S.-H. Tsao and S.-S. Jan, "Observability analysis and consistency improvements for visual-inertial odometry on the matrix lie group of extended poses," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 8341–8353, 2021.

[25] J. Huai, Y. Lin, Y. Zhuang, and M. Shi, "Consistent right-invariant fixed-lag smoother with application to visual inertial slam," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 7, May 2021, pp. 6084–6092. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16758

[26] G. Huang, A. I. Mourikis, and S. I. Roumeliotis, "A first-estimates jacobian EKF for improving SLAM consistency," in *Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, O. Khatib, V. Kumar, and G. J. Pappas, Eds. Springer Berlin Heidelberg, 2009, vol. 54, pp. 373–382.

[27] 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.

[28] G. Chirikjian, *Stochastic models, information theory, and Lie groups*, ser. Applied and numerical harmonic analysis. Birkhäuser, no. v. 1. [Online]. Available: https://books.google.com/books?id=lfOMoAEACAAJ

[29] S. Bonnabel, "Symmetries in observer design: review of some recent results and applications to ekf-based slam," *CoRR*, vol. abs/1105.2254, 2011. [Online]. Available: http://arxiv.org/abs/1105.2254

[30] A. Barrau and S. Bonnabel, "An EKF-SLAM algorithm with consistency properties," *CoRR*, vol. abs/1510.06263, 2015. [Online]. Available: http://arxiv.org/abs/1510.06263

[31] A. Barrau, "Non-linear state error based extended kalman filters with applications to navigation," Ph.D. dissertation, Center for Robotics, PSL Research University, Mines ParisTech, 2015.

[32] A. Barrau and S. Bonnabel, "Invariant kalman filtering," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 237–257, 2018.

[33] Y. Yang, C. Chen, W. Lee, and G. Huang, "Supplementary materials: Decoupled right invariant error states for consistent visual-inertial navigation," University of Delaware, Tech. Rep. RPNG-2021-DRI, 2021. [Online]. Available: https://yangyulin.net/papers/2021_supp_DRI.pdf

[34] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2017.

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

[36] 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.

[37] Y. Yang and G. Huang, "Aided inertial navigation with geometric features: Observability analysis," in *Proc. of the IEEE International Conference on Robotics and Automation*, Brisbane, Australia, May 21–25, 2018.

[38] ——, "Aided inertial navigation: Unified feature representations and observability analysis," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3528–3534.

[39] ——, "Observability analysis of aided ins with heterogeneous features of points, lines and planes," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 399–1418, Dec. 2019.

[40] Y. Yang, P. Geneva, K. Eckenhoff, and G. Huang, "Degenerate motion analysis for aided INS with online spatial and temporal calibration," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 2, pp. 2070–2077, 2019.

[41] Y. Yang, P. Geneva, X. Zuo, and G. Huang, "Online imu intrinsic calibration: Is it necessary?" in *Proc. of Robotics: Science and Systems (RSS)*, Corvallis, Or, 2020, pp. 1–8.

[42] Y. Yang, W. Lee, P. Osteen, P. Geneva, X. Zuo, and G. Huang, "icalib: Inertial aided multi-sensor calibration," in *VINS Workshop 2021 at ICRA, Xi'an, China*, 2021, pp. 1–8.

[43] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The TUM VI benchmark for evaluating visual-inertial odometry," *CoRR*, vol. abs/1804.06120, 2018. [Online]. Available: http://arxiv.org/abs/1804.06120