Optimization-based VINS: Consistency, Marginalization, and FEJ

Chuchu Chen - ccchu@udel.edu
Patrick Geneva - pgeneva@udel.edu
Yuxiang Peng - yxpeng@udel.edu
Woosik Lee - woosik@udel.edu
Guoquan Huang - ghuang@udel.edu

Department of Mechanical Engineering
University of Delaware, Delaware, USA

Robot Perception and Navigation Group (RPNG)
Tech Report - RPNG-2023-GRAPH
Last Updated - September 9, 2023

Contents

1 Optimization-Based VINS

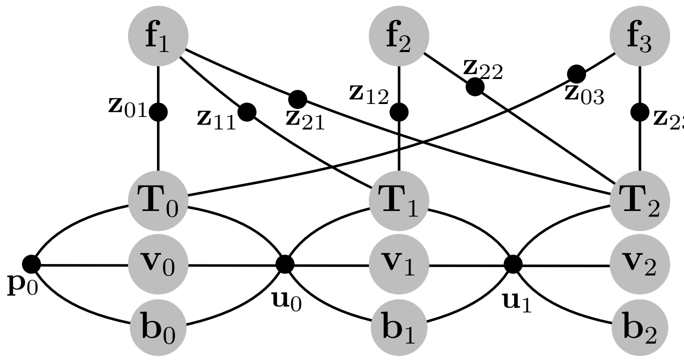


Figure 1: Example visual-inertial factor graph of bearing observations $\mathbf{z}$, inertial readings $\mathbf{u}$, and prior $\mathbf{p}_0$. The nodes (state variables) are represented as grey circles and edges (measurements) connect their related states. $\mathbf{f}_j$ denotes the jth feature, $\mathbf{T}_i$ is robot pose at t_i, $\mathbf{v}_i$ and $\mathbf{b}_i$ are the robot velocity and biases.

We formulate the nonlinear least squares (NLS) problem over the entire trajectory up to the current time t_k. The system state consists of the current navigation states, $\mathbf{x}_k$, and 3D features, $\mathbf{x}_f$:

$$\mathbf{x}_{0:k} = \begin{bmatrix}\mathbf{x}_0^\top & \dots & \mathbf{x}_k^\top & \mathbf{x}_f^\top\end{bmatrix}^\top \tag{1}$$

$$\mathbf{x}_k = \begin{bmatrix}{}^{I_k}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_k}^\top & {}^{G}\mathbf{v}_{I_k}^\top & \mathbf{b}_{g,k}^\top & \mathbf{b}_{a,k}^\top\end{bmatrix}^\top \tag{2}$$

$$\mathbf{x}_f = \begin{bmatrix}{}^{G}\mathbf{f}_1^\top & \dots & {}^{G}\mathbf{f}_g^\top\end{bmatrix}^\top \tag{3}$$

where ${}^{I}_{G}\bar{q}$ is the unit quaternionthat represents the rotation ${}^{I}_{G}\mathbf{R}$ from global frame $\{G\}$ to the IMU frame $\{I\}$. Note that throughout the paper, $\hat{\mathbf{x}}$ is used to denote the *current* best estimate of a random variable $\mathbf{x}$ with $\delta\mathbf{x} = \mathbf{x} \boxminus \hat{\mathbf{x}}$ denotes the error state. For the quaternion error state, we employ JPL multiplicative error [1] and use $\delta\boldsymbol{\theta} \in \mathbb{R}^3$ defined by the error quaternion i.e., $\delta\bar{q} = \bar{q} \otimes \hat{\bar{q}}^{-1} \simeq [\frac{1}{2}\delta\boldsymbol{\theta}^\top \; 1]^\top$. The "$\boxplus$" and "$\boxminus$" operations map elements to and from a given manifold and equate to simple "+" and "-" for vector variables [2]. ${}^{G}\mathbf{p}_I$ and ${}^{G}\mathbf{v}_I$ are the IMU position and velocity in $\{G\}$, respectively; $\mathbf{b}_g$ and $\mathbf{b}_a$ are the gyroscope and accelerometer biases; and the feature state $\mathbf{x}_f$ comprises the global position of g landmarks. A common visualization technique in the optimization-based method is the "factor graph" [see Figure 1] for which states are represented as graph nodes and measurements are represented as edges which connect to their involve states.

1.1 IMU Kinematic Constraints

The inertial state is integrated forward using a series of IMU measurements $\mathbf{u}$ over a time interval $[t_{i-1}, t_i]$, consisting of linear accelerations ${}^{I}\mathbf{a}_m$ and angular velocities ${}^{I}\boldsymbol{\omega}_m$. The motion model can be described based on the following generic nonlinear IMU function [3]:

$$\mathbf{g}(\mathbf{x}_{0:k}, \mathbf{u}_i, \mathbf{w}_i) = \mathbf{x}_{i+1} - \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i + \mathbf{w}_i) = \mathbf{0} \tag{4}$$

where the measurements $\mathbf{u}_i$ are contaminated by zero-mean white Gaussian noises $\mathbf{w}_i$ with covariance $\mathbf{Q}_i$. Linearize the above equation we can derive the Jacobians with respect to state vector, Eq. (1), and noise $\mathbf{w}_i$, respectively:

$$\mathbf{0} \simeq \tilde{\mathbf{x}}_{i+1} - \mathbf{\Phi}_{I_i}\tilde{\mathbf{x}}_i - \mathbf{G}_i\mathbf{w}_i \tag{5}$$

$$\mathbf{\Phi}_i := \frac{\partial \mathbf{g}}{\partial \mathbf{x}_{0:k}} = \begin{bmatrix}\mathbf{0} & \cdots & -\mathbf{\Phi}_{I_i} & \mathbf{I} & \cdots & \mathbf{0}\end{bmatrix} \tag{6}$$

$$\mathbf{G}_i := \frac{\partial \mathbf{g}}{\partial \mathbf{w}_i} \tag{7}$$

After linearization, the state translation matrix can be derived as [4]:

$$\mathbf{\Phi}_{I_i} = \begin{bmatrix} \mathbf{\Phi}_{11} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{\Phi}_{14} & \mathbf{0}_3 \\ \mathbf{\Phi}_{21} & \mathbf{I}_3 & \mathbf{I}_3\Delta t & \mathbf{\Phi}_{24} & \mathbf{\Phi}_{25} \\ \mathbf{\Phi}_{31} & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{\Phi}_{34} & \mathbf{\Phi}_{35} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{8}$$

with:

$$\mathbf{\Phi}_{11} = {}^{I_i}_{I_{i+1}}\hat{\mathbf{R}}^\top \qquad \mathbf{\Phi}_{14} = -\mathbf{J}_r\left({}^{I_i}\hat{\boldsymbol{\theta}}_{I_{i+1}}\right)\Delta t \qquad \mathbf{\Phi}_{21} = -{}^{I_i}_{G}\hat{\mathbf{R}}^\top\lfloor\Xi_2\hat{\mathbf{a}}_i\rfloor \qquad \mathbf{\Phi}_{24} = {}^{I_i}_{G}\hat{\mathbf{R}}^\top\Xi_4 \tag{9}$$

$$\mathbf{\Phi}_{25} = -{}^{I_i}_{G}\hat{\mathbf{R}}^\top\Xi_2 \qquad \mathbf{\Phi}_{31} = -{}^{I_i}_{G}\hat{\mathbf{R}}^\top\lfloor\Xi_1\hat{\mathbf{a}}_i\rfloor \qquad \mathbf{\Phi}_{34} = {}^{I_i}_{G}\hat{\mathbf{R}}^\top\Xi_3 \qquad \mathbf{\Phi}_{35} = -{}^{I_i}_{G}\hat{\mathbf{R}}^\top\Xi_1 \tag{10}$$

and:

$$\mathbf{\Xi}_1 \triangleq \int_{t_i}^{t_{i+1}} \exp\left({}^{I_i}\hat{\boldsymbol{\omega}}\delta\tau\right)d\tau \qquad \mathbf{\Xi}_2 \triangleq \int_{t_i}^{t_{i+1}}\int_{t_i}^{s} \exp\left({}^{I_i}\hat{\boldsymbol{\omega}}\delta\tau\right)d\tau ds \tag{11}$$

$$\mathbf{\Xi}_3 \triangleq \int_{t_i}^{t_{i+1}} {}^{I_i}_{I_\tau}\mathbf{R}\lfloor{}^{I_\tau}\hat{\mathbf{a}}\rfloor\mathbf{J}_r\left({}^{I_i}\hat{\boldsymbol{\omega}}\delta\tau\right)\delta\tau d\tau \qquad \mathbf{\Xi}_4 \triangleq \int_{t_i}^{t_{i+1}}\int_{t_i}^{s} {}^{I_i}_{I_\tau}\mathbf{R}\lfloor{}^{I_\tau}\hat{\mathbf{a}}\rfloor\mathbf{J}_r\left({}^{I_k}\hat{\boldsymbol{\omega}}\delta\tau\right)\delta\tau d\tau ds \tag{12}$$

A comprehensive derivation of IMU preintegration can be found in CPI [5] or ACI^2 [4], which we have omitted in this technical report.

1.2 Feature Observation Constraints

The camera provides bearing observations of environmental 3D points. These observations can be related to our states using the following measurement function (note that we here assume the global 3D feature model):

$$\mathbf{z}_{ij} = \mathbf{h}(\mathbf{x}_{0:k}) + \mathbf{n}_{ij} \tag{13}$$

$$=: \mathbf{\Lambda}({}^{C_i}\mathbf{f}_j) + \mathbf{n}_{ij} \tag{14}$$

$$\mathbf{\Lambda}([x\ y\ z]^\top) =: [x/z\ y/z]^\top \tag{15}$$

$${}^{C_i}\mathbf{f}_j = [x\ y\ z]^\top = {}^{C}_{I}\mathbf{R}{}^{I_i}_{G}\mathbf{R}\left({}^{G}\mathbf{f}_j - {}^{G}\mathbf{p}_{I_i}\right) + {}^{C}\mathbf{p}_I \tag{16}$$

where $\mathbf{n}_{ij} \sim \mathcal{N}(0, \mathbf{R}_{ij})$ is the white Gaussian bearing measurement noise and $\{{}^{C}_{I}\mathbf{R}, {}^{C}\mathbf{p}_I\}$ is the camera-IMU rigid transformation. Linearizing the measurement function with respect to the current state $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{f}}_j$ we get the following:

$$\mathbf{z}_{ij} \simeq \mathbf{h}(\hat{\mathbf{x}}_i, \hat{\mathbf{f}}_j) + \mathbf{H}_{I_i}(\mathbf{x}_i \boxminus \hat{\mathbf{x}}_i) + \mathbf{H}_{f_j}(\mathbf{f}_j - \hat{\mathbf{f}}_j) + \mathbf{n}_{ij} \tag{17}$$

$$= \mathbf{h}(\hat{\mathbf{x}}_{0:k}) + \mathbf{H}_{ij}\tilde{\mathbf{x}}_{0:k} + \mathbf{n}_{ij} \tag{18}$$

where $\mathbf{H}_{ij}$ denotes the Jacobian evaluated at $\hat{\mathbf{x}}_{0:k}$, it only contains non-zero blocks for the ith robot state, $\mathbf{x}_i$, and the jth feature, ${}^G\mathbf{f}_j$, thus is computed as:

$$\mathbf{H}_{ij} := \frac{\partial \mathbf{h}}{\partial \mathbf{x}_{0:k}} = \left[\begin{array}{ccccc|ccccc} \mathbf{0} & \cdots & \mathbf{H}_{x_i} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{H}_{f_j} & \cdots & \mathbf{0} \end{array}\right] \tag{19}$$

with

$$\mathbf{H}_{x_i} = \mathbf{H}_{proj,i} \begin{bmatrix} \mathbf{H}_{\theta_i} & \mathbf{H}_{p_i} & \mathbf{0}_{3\times 9} \end{bmatrix} \tag{20}$$

$$\mathbf{H}_{f_j} = -\mathbf{H}_{proj,i}\mathbf{H}_{p_j} \tag{21}$$

$$\mathbf{H}_{\theta_i} = \lfloor {}^{I_i}_{G}\hat{\mathbf{R}} \left({}^G\hat{\mathbf{f}}_j - {}^G\hat{\mathbf{p}}_{I_i} \right) \rfloor \tag{22}$$

$$\mathbf{H}_{p_i} = -{}^{I_i}_{G}\hat{\mathbf{R}} \tag{23}$$

$$\mathbf{H}_{proj,i} = \frac{1}{\hat{z}_i{}^2} \begin{bmatrix} \hat{z}_i & 0 & -\hat{x}_i \\ 0 & \hat{z}_i & -\hat{y}_i \end{bmatrix} {}^C_I\mathbf{R} \tag{24}$$

A code implementation can be seen in Appendix C.

1.3 Batch MAP Formulation

At timestep t_k, the batch maximum a posteriori (MAP) seeks to solve for the history of the state estimate $\hat{\mathbf{x}}_{0:k}$ by maximizing the posterior pdf leveraging: 1) prior information $\mathcal{N}(\hat{\mathbf{x}}_0, \mathbf{P}_0)$, 2) IMU motion constraints $\mathbf{u}$ (see Section 1.1), and 3) camera observation measurements $\mathbf{z}$ (see Section 1.2):

$$p(\mathbf{x}_{0:k}|\mathcal{Z}_{0:k}) \propto p(\mathbf{x}_0) \prod_{i=0}^{k-1} p(\mathbf{x}_{i+1}|\mathbf{x}_i, \mathbf{u}_i) \prod_{\mathbf{z}_{ij}\in\mathcal{Z}_{0:k}} p(\mathbf{z}_{ij}|\mathbf{x}_i, \mathbf{f}_j) \tag{25}$$

where the set $\mathcal{Z}_{0:k}$ denotes all measurements between $[t_0, t_k]$. Under Gaussian distribution assumption, this pdf can be written as:

$$p(\mathbf{x}_{0:k}|\mathcal{Z}_{0:k}) \propto \frac{1}{\sqrt{(2\pi)^n|\mathbf{P}_0|}} \exp\left(-\mathcal{C}_{p_0}\right) \prod_{i=0}^{k-1} \frac{1}{\sqrt{(2\pi)^d|\mathbf{Q}_i|}} \exp(-\mathcal{C}_{I_i}) \prod_{\mathbf{z}_{ij}\in\mathcal{Z}_{0:k}} \frac{1}{\sqrt{(2\pi)^m|\mathbf{R}_{ij}|}} \exp(-\mathcal{C}_{f_{ij}})$$

where n is the dimension of prior state $\mathbf{x}_0$, d denote the size of robot state $\mathbf{x}_k$ and m is the dimension of measurement $\mathbf{z}_{ij}$. Maximizing the above pdf is equivalent to minimizing:

$$\mathcal{C}(\mathbf{x}_{0:k}) = \mathcal{C}_{p_0} + \sum_{i=0}^{k-1} \mathcal{C}_{I_i} + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_{0:k}} \mathcal{C}_{f_{ij}} \tag{26}$$

where we define the following costs:

$$\text{Prior:} \quad \mathcal{C}_{p_0} = \frac{1}{2}||\mathbf{x}_0 \boxminus \hat{\mathbf{x}}_0||^2_{\mathbf{P}_0} \tag{27}$$

$$\text{Inertial: [5]} \quad \mathcal{C}_{I_i} = \frac{1}{2}||\mathbf{x}_{i+1} \boxminus \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i)||^2_{\mathbf{Q}_i} \tag{28}$$

$$\text{Camera: [6]} \quad \mathcal{C}_{f_{ij}} = \frac{1}{2}||\mathbf{z}_{ij} \boxminus \mathbf{h}(\mathbf{x}_i, \mathbf{f}_j)||^2_{\mathbf{R}_{ij}} \tag{29}$$

where $||\mathbf{a}||^2_{\mathbf{W}} := \mathbf{a}^\top \mathbf{W}^{-1}\mathbf{a}$ and can be solved iteratively given an initial linearization point. The second-order Taylor series of the l-th iteration with linearization point $\hat{\mathbf{x}}^l_{0:k}$ is:

$$\mathcal{C}(\hat{\mathbf{x}}^l_{0:k} \boxplus \delta\mathbf{x}^l_{0:k}) \simeq \mathcal{C}(\hat{\mathbf{x}}^l_{0:k}) + \mathbf{b}^{l^\top}\delta\mathbf{x}^l_{0:k} + \frac{1}{2}\delta\mathbf{x}^{l^\top}_{0:k}\mathbf{A}^l\delta\mathbf{x}^l_{0:k} \tag{30}$$

where $\mathbf{b}^l$ and $\mathbf{A}^l$ are the linearized gradient and Hessian of the cost function, respectively.

$$\mathbf{b}^l = \mathbf{\Gamma}_0^\top \mathbf{P}_0^{-1}(\hat{\mathbf{x}}^l_0 \boxminus \hat{\mathbf{x}}_0) + \sum_{i=0}^{k-1} \mathbf{\Phi}_i^{l^\top}\mathbf{Q}_i^{-1}\left(\hat{\mathbf{x}}^l_{i+1} \boxminus \mathbf{f}(\hat{\mathbf{x}}^l_i, \mathbf{u}_i)\right) + \sum_{\mathbf{z}_{i,j}\in\mathcal{Z}_{0:k}} \mathbf{H}_{ij}^{l^\top}\mathbf{R}_{ij}^{-1}\left(\mathbf{z}_{ij} \boxminus \mathbf{h}(\hat{\mathbf{x}}^l_{0:k})\right) \tag{31}$$

$$\mathbf{A}^l = \mathbf{\Gamma}_0^\top \mathbf{P}_0^{-1}\mathbf{\Gamma}_0 + \sum_{i=0}^{k-1} \mathbf{\Phi}_i^{l^\top}\mathbf{Q}_i^{-1}\mathbf{\Phi}_i^l + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_{0:k}} \mathbf{H}_{ij}^{l^\top}\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}^l \tag{32}$$

where $\mathbf{\Gamma}_0 = \begin{bmatrix}\mathbf{I}_n & \mathbf{0} & \dots & \mathbf{0}\end{bmatrix}$ with $n = \mathtt{dim}(\mathbf{x}_0)$ is the size of initial state $\mathbf{x}_0$. This indicates that we add a prior factor to the initial state variables up to the time t_k. The correction term and the updated state $\hat{\mathbf{x}}^{l+1}_{0:k}$ can be solved by:

$$\mathbf{A}^l\delta\mathbf{x}^l_{0:k} = -\mathbf{b}^l \quad \Rightarrow \quad \hat{\mathbf{x}}^{l+1}_{0:k} = \hat{\mathbf{x}}^l_{0:k} \boxplus \delta\mathbf{x}^l_{0:k} \tag{33}$$

Given initial state $\hat{\mathbf{x}}_0$, this iterative algorithm will compute the global minimal estimates (MAP) for the entire state $\mathbf{x}_{0:k}$ given all available measurements within time period $[t_0, t_k]$.

2 Marginalization and Consistency

Ideally, as the robot moves throughout the environment and observes new features, one could reformulate the batch-MAP problem and solve the objective function, Eq. (26), using all available measurements. This quickly becomes intractable as the size of the state becomes larger and becomes prohibitively expensive (complexity is $O(n^3)$ in size of the state) for real-time estimation, and thus requires marginalization of states to bound computational complexity.

However, this marginalization can cause issues because several states are permanently approximated and fixed, which will result in different observability properties as compared to the batch-MAP estimator. Particularly, the estimator erroneously believes it has gained information along directions that it cannot measure and results in overconfident estimates. This inconsistency can be a severe problem since it will also degrade the accuracy and make the estimator unreliable. In the following sections, we explain the marginalization-related inconsistency in more detail.

2.1 State Marginalization

The graph is partitioned into the to-marginalize states $\mathbf{x}_M$, remaining states connected to $\mathbf{x}_M$ are $\mathbf{x}_R$ (the Markov blanket), and new states not involved $\mathbf{x}_N$ (See Figure 2). Before marginalization, the MAP at time t_k can be formulated as:

$$\mathcal{C}(\mathbf{x}_{0:k}) = \mathcal{C}(\mathbf{x}_M, \mathbf{x}_R, \mathbf{x}_N) = \mathcal{C}_{mr}(\mathbf{x}_M, \mathbf{x}_R) + \mathcal{C}_{rn}(\mathbf{x}_R, \mathbf{x}_N)$$

where the cost term $\mathcal{C}_{mr}$ contains all terms involve states $\mathbf{x}_M$ and $\mathbf{x}_R$ only and $\mathcal{C}_{rn}$ contains all terms involve states $\mathbf{x}_R$ and $\mathbf{x}_N$ only. Note that once the states are marginalized they will not be

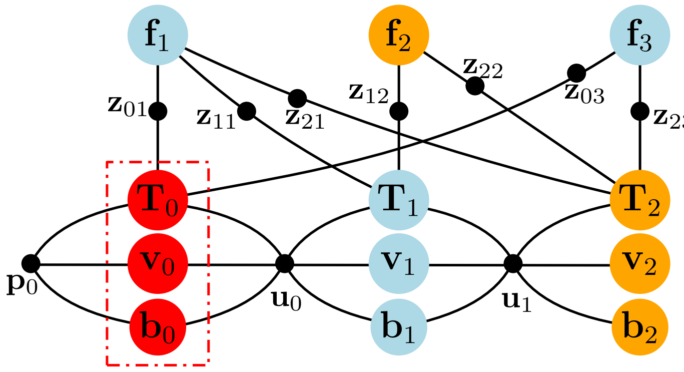


Figure 2: Example factor graph with different state definitions and Markov blanket. Red nodes denote to-marginalize states $\mathbf{x}_M = \begin{bmatrix} \mathbf{T}_0^\top & \mathbf{v}_0^\top & \mathbf{b}_0^\top \end{bmatrix}^\top$, blue ones represent remaining states $\mathbf{x}_R = \begin{bmatrix} \mathbf{T}_1^\top & \mathbf{v}_1^\top & \mathbf{b}_1^\top & \mathbf{f}_1^\top & \mathbf{f}_3^\top \end{bmatrix}^\top$, and the orange nodes are new states not involved $\mathbf{x}_N = \begin{bmatrix} \mathbf{T}_2^\top & \mathbf{v}_2^\top & \mathbf{b}_2^\top & \mathbf{f}_2^\top \end{bmatrix}^\top$.

involved in any measurement functions after that timestamp. As such, there are no joint terms (cost functions) containing both $\mathbf{x}_M$ and $\mathbf{x}_N$. We can then derive the following minimization problem:

$$\min_{\mathbf{x}_M,\mathbf{x}_R,\mathbf{x}_N} \mathcal{C}(\mathbf{x}_M,\mathbf{x}_R,\mathbf{x}_N) = \min_{\mathbf{x}_R,\mathbf{x}_N}\left(\min_{\mathbf{x}_M} \mathcal{C}\left(\mathbf{x}_M,\mathbf{x}_R,\mathbf{x}_N\right)\right) = \min_{\mathbf{x}_R,\mathbf{x}_N}\left(\min_{\mathbf{x}_M} \mathcal{C}_{mr}(\mathbf{x}_M,\mathbf{x}_R) + \mathcal{C}_{rn}\left(\mathbf{x}_R,\mathbf{x}_N\right)\right) \tag{34}$$

To solve the above problem, we start with minimize $\mathcal{C}_{mr}$ at time t_k with respect to $\mathbf{x}_M$ as follows:

$$\mathcal{C}_{mr}(\mathbf{x}_M,\mathbf{x}_R) = \mathcal{C}_{p_0} + \sum_{i=0}^{m-1}\mathcal{C}_{I_i} + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_{0:m}} \mathcal{C}_{f_{ij}} \tag{35}$$

$$\simeq \mathcal{C}(\hat{\mathbf{x}}_M(k),\hat{\mathbf{x}}_R(k)) + \begin{bmatrix}\mathbf{b}_{mm}(k)\\ \mathbf{b}_{rm}(k)\end{bmatrix}^\top \begin{bmatrix}\mathbf{x}_M-\hat{\mathbf{x}}_M(k)\\ \mathbf{x}_R-\hat{\mathbf{x}}_R(k)\end{bmatrix}$$
$$+\frac{1}{2}\begin{bmatrix}\mathbf{x}_M-\hat{\mathbf{x}}_M(k)\\ \mathbf{x}_R-\hat{\mathbf{x}}_R(k)\end{bmatrix}^\top \begin{bmatrix}\mathbf{A}_{mm}(k) & \mathbf{A}_{mr}(k)\\ \mathbf{A}_{rm}(k) & \mathbf{A}_{rr}(k)\end{bmatrix} \begin{bmatrix}\mathbf{x}_M-\hat{\mathbf{x}}_M(k)\\ \mathbf{x}_R-\hat{\mathbf{x}}_R(k)\end{bmatrix} \tag{36}$$

where $\hat{\mathbf{x}}(k)$ denotes the best state estimate at time t_k, $\mathbf{b}(k)$ and $\mathbf{A}(k)$ are the gradient and Hessian matrix computed using $\hat{\mathbf{x}}(k)$, respectively. The optimal value for $\mathbf{x}_M$ can be derived by solving the quadratic cost function as:

$$\mathbf{x}_M = \hat{\mathbf{x}}_M(k) - \mathbf{A}_{mm}^{-1}(k)\left(\mathbf{b}_{mm}(k) + \mathbf{A}_{mr}(k)(\mathbf{x}_R-\hat{\mathbf{x}}_R(k))\right) \tag{37}$$

Substituting in $\mathcal{C}_{mr}$ we can get its minimal value:

$$\min_{\mathbf{x}_M} = \mathcal{C}_{mr}(\mathbf{x}_M,\mathbf{x}_R) \simeq \alpha + \mathbf{b}_p(k)^\top\left(\mathbf{x}_R-\hat{\mathbf{x}}_R(k)\right) + \frac{1}{2}\left(\mathbf{x}_R-\hat{\mathbf{x}}_R(k)\right)^\top \mathbf{A}_p(k)\left(\mathbf{x}_R-\hat{\mathbf{x}}_R(k)\right) \tag{38}$$

where α is independent from state $\mathbf{x}_R$ and $\mathbf{x}_M$, and

$$\mathbf{b}_p(k) = \mathbf{b}_{mr}(k) - \mathbf{A}_{rm}(k)\mathbf{A}_{mm}^{-1}(k)\mathbf{b}_{mm}(k) \tag{39}$$
$$\mathbf{A}_p(k) = \mathbf{A}_{rr}(k) - \mathbf{A}_{mm}^{-1}(k)\mathbf{A}_{mm}(k) \tag{40}$$

The cost function with *all* states, Eq. (34), can then be approximated as:

$$\begin{aligned}\mathcal{C}'(\mathbf{x}_R,\mathbf{x}_N) &= \mathbf{b}_p(k)^\top(\mathbf{x}_R \boxminus \hat{\mathbf{x}}_R(k)) + \frac{1}{2}(\mathbf{x}_R \boxminus \hat{\mathbf{x}}_R(k))^\top \mathbf{A}_p(k)(\mathbf{x}_R \boxminus \hat{\mathbf{x}}_R(k)) \\ &\quad + \sum_{i=m}^{k'-1}\frac{1}{2}||\mathbf{x}_{i+1} \boxminus \mathbf{f}(\mathbf{x}_i,\mathbf{u}_i)||^2_{\mathbf{Q}_i} + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_A}\frac{1}{2}||\mathbf{z}_{ij} \boxminus \mathbf{h}(\mathbf{x}_{0:k'})||^2_{\mathbf{R}_{ij}} \end{aligned} \tag{41}$$

where $\mathcal{Z}_A$ is the active measurement set (all measurements involving state $\mathbf{x}_R$ and $\mathbf{x}_N$). This cost function is *independent* from the marginalized state $\mathbf{x}_M$. This is because $\mathcal{C}_{mr}$ has been approximated by its sceond-order Taylor expansion (see Eq.(36) and Eq.(37)) and the information has been "absorbed" into the optimization problem with the remaining state $\hat{\mathbf{x}}_R(k)$, the marginalized gradient $\mathbf{b}_p(k)$ and the marginalized information matrix $\mathbf{A}_p(k)$ at timestamp t_k. Note that this approximation will introduce small errors to the MAP problem but is able to bound the state size to balance accuracy and efficiency. The minimization of cost function $\mathcal{C}'(\mathbf{x}_R,\mathbf{x}_N)$ at timestamp t'_k can then be solved iteratively, where the gradient and Hessian at the l-th iteration can be found as:

$$\begin{aligned}\mathbf{b}^l &= \mathbf{\Gamma}_r^\top \mathbf{b}_p(k) + \mathbf{\Gamma}_r^\top \mathbf{A}_p(k)(\mathbf{x}_R^l \boxminus \hat{\mathbf{x}}_R(k)) \\ &\quad + \sum_{k=m}^{k'-1}\mathbf{\Phi}_i^{l\top}\mathbf{Q}_i^{-1}\left(\mathbf{x}_{i+1}^l \boxminus \mathbf{f}(\mathbf{x}_i^l,\mathbf{u}_i)\right) + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_A}\mathbf{H}_{ij}^{l\top}\mathbf{R}_{ij}^{-1}(\mathbf{z}_{ij} \boxminus \mathbf{h}_{ij}(\mathbf{x}_{0:k'}))\end{aligned} \tag{42}$$

$$\mathbf{A}^l = \mathbf{\Gamma}_r^\top \mathbf{A}_p(k)\mathbf{\Gamma}_r + \sum_{k=m}^{k'-1}\mathbf{\Phi}_i^{l\top}\mathbf{Q}_i^{-1}\mathbf{\Phi}_i^l + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_A}\mathbf{H}_{ij}^{l\top}\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}^l \tag{43}$$

where $\mathbf{\Gamma}_r = \begin{bmatrix}\mathbf{I}_r & \mathbf{0} & \dots & \mathbf{0}\end{bmatrix}$ and $r = \mathtt{dim}(\mathbf{x}_R)$ is the size of remaining state. It is important to note that in the above expressions, all the involved Jacoboians are evaluated using all the available measurements up to timestamp $t_{k'}$.

2.2 System Observability

We now investigate the system observability properties before, t_k and after marginalization, $t_{k'}$ as introduced in the previous section. This analysis follows the work of [7, 8] and [9] with the further extension to include inertial biases.

To show the differences in observability properties between the full-batch MAP estimator and the sub-optimal system with state marginalization, we will begin by examining the information matrix in the batch-MAP estimator. Specifically, we can derive the information matrix at time $t_{k'}$ as follows:

$$\begin{aligned}\mathbf{A}_{k'}^{\text{full}} &= \sum_{i=0}^{k'-1}\mathbf{\Phi}_i(k')^\top\mathbf{Q}_i^{-1}\mathbf{\Phi}_i(k') \\ &\quad + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_M}\mathbf{H}_{ij}(k')^\top\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}(k') + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_A}\mathbf{H}_{ij}(k')^\top\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}(k') \end{aligned} \tag{44}$$

$$= \begin{bmatrix}\mathbf{A}_{mm}(k') & \mathbf{A}_{mr}(k') & \mathbf{0} \\ \mathbf{A}_{rm}(k') & \mathbf{A}_{rr}(k') & \mathbf{A}_{rn}(k') \\ \mathbf{0} & \mathbf{A}_{nr}(k') & \mathbf{A}_{nn}(k')\end{bmatrix} \tag{45}$$

We note that *all* the Hessian matrix corresponding to *all* the states are evaluated at the latest (most recent) timestamp $t_{k'}$, we use $\mathbf{A}(k')$ to denote the linearization points for states that are involved in this matrix. Similarly, we can derive the information matrix at time $t_{k'}$ for the case that state marginalization is performed to $\mathbf{x}_M$:

$$\begin{aligned}\mathbf{A}_{k'}^{\text{marg}} &= \sum_{k=0}^{m-1} \mathbf{\Phi}_i(k)^\top \mathbf{Q}_i^{-1} \mathbf{\Phi}_i(k) + \sum_{i=m}^{k'-1} \mathbf{\Phi}_i(k')^\top \mathbf{Q}_i^{-1} \mathbf{\Phi}_i(k') \\ &\quad + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_M} \mathbf{H}_{ij}(k)^\top \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k) + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_A} \mathbf{H}_{ij}(k')^\top \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k') \qquad (46) \\ &= \begin{bmatrix} \mathbf{A}_{mm}(k) & \mathbf{A}_{mr}(k) & \mathbf{0} \\ \mathbf{A}_{rm}(k) & \mathbf{A}_{rr}(k) + \mathbf{A}_{rr}(k') & \mathbf{A}_{rn}(k') \\ \mathbf{0} & \mathbf{A}_{nr}(k') & \mathbf{A}_{nn}(k') \end{bmatrix} \qquad (47)\end{aligned}$$

It can be proved that [see Appendix A and B]:

$$\begin{aligned}\texttt{rank}(\mathbf{A}_{k'}^{\text{full}}) &< \texttt{rank}(\mathbf{A}_{k'}^{\text{marg}}) \\ \texttt{dim}(\mathbb{N}(\mathbf{A}_{k'}^{\text{full}})) &> \texttt{dim}(\mathbb{N}(\mathbf{A}_{k'}^{\text{marg}}))\end{aligned} \qquad (48)$$

where $\mathbb{N}(\mathbf{A})$ is the nullspace of $\mathbf{A}$. This implies that by performing marginalization, which fixes the information related to the marginal and remaining states at time t_k, spurious information has been gained to cause a loss of nullspace dimension and leading to *inconsistencies*. We can inspect the nullspace of $\mathbf{A}_{k'}^{\text{full}}$ [see Appendix A]:

$$\mathbb{N}(\mathbf{A}_{k'}^{\text{full}}) = \begin{bmatrix} \mathbb{N}_{x_0}^\top & \cdots & \mathbb{N}_{x_{k'}}^\top & \mathbb{N}_{f_0}^\top & \cdots & \mathbb{N}_{f_g}^\top \end{bmatrix}^\top \qquad (49)$$

$$\mathbb{N}_{x_i} = \begin{bmatrix} {}^{i}_{G}\mathbf{R}\mathbf{g} & \mathbf{0}_3 \\ \lfloor {}^G\mathbf{p}_i \rfloor \mathbf{g} & \mathbf{I}_3 \\ \lfloor {}^G\mathbf{v}_i \rfloor \mathbf{g} & \mathbf{0}_3 \\ \mathbf{0}_{3\times1} & \mathbf{0}_3 \\ \mathbf{0}_{3\times1} & \mathbf{0}_3 \end{bmatrix} \qquad (50)$$

$$\mathbb{N}_{f_j} = \begin{bmatrix} \lfloor {}^G\mathbf{f}_j \rfloor \mathbf{g} & \mathbf{I}_3 \end{bmatrix} \qquad (51)$$

where $\mathbf{g}$ denotes the global gravity. For a more comprehensive explanation, we refer the reader to Appendix A.

3 Marginalization and FEJ

In this section, we discuss how the first-estimates Jacobian (FEJ) method is leveraged to prevent erroneous information gain due to marginalization. The key idea is to evaluate the Hessian using the first estimate $\hat{\mathbf{x}}_R(k)$ instead of the current estimate $\hat{\mathbf{x}}_R(k')$ for all states $\mathbf{x}_R$ involved with the marginal (the second cost in Eq. (51)):

$$\begin{aligned}\mathcal{C}(\mathbf{x}(k)) &\simeq \mathcal{C}(\hat{\mathbf{x}}_R(k'), \hat{\mathbf{x}}_N(k')) + \mathbf{b}(\hat{\mathbf{x}}_R(k), \hat{\mathbf{x}}_N(k'))^\top \delta\mathbf{x}(k') \\ &\quad + \frac{1}{2}\delta\mathbf{x}(k')^\top \mathbf{A}(\hat{\mathbf{x}}_R(k), \hat{\mathbf{x}}_N(k'))\delta\mathbf{x}(k')\end{aligned} \qquad (52)$$

where $\delta\mathbf{x}(k') = [\delta\mathbf{x}_R(k')^\top \ \delta\mathbf{x}_N(k')^\top]^\top$. Note that the linearization point of $\mathbf{x}_R$ only needs to be changed for the Hessian $\mathbf{A}(\hat{\mathbf{x}}_R(k), \hat{\mathbf{x}}_A(k'))$ and gradient $\mathbf{b}(\hat{\mathbf{x}}_R(k), \hat{\mathbf{x}}_N(k'))$ computation, while the residual and states which do not affect the observability properties can use the best estimates (e.g., biases, see Eq. (49)). Algorithm 1 outlines the process of performing FEJ.

Algorithm 1 Sliding-window optimization VINS with FEJ

```
Building factor graph and performing iterative optimization:
    • Construct optimization problem using all measurements at
      timestamp $t(k')$ [See Eq. (26)] and linearize the cost function:
        – If state has FEJ value: use its First estimate $\hat{\mathbf{x}}(k)$
        – Else: use its Update estimate $\hat{\mathbf{x}}(k')$
    • Correct the state iteratively [See Eq.(33)].
State marginalization:
    • Select the state $\mathbf{x}_M$ to be marginalize.
    • If $\mathbf{x}_R$ is connected to $\mathbf{x}_M$ && in the nullspace [Eq. (49)] &&
      not FEJ'ed: Set its current estimate as FEJ value.
    • Perform state marginalization to calculate the new prior information
      and gradient [see Eq. (39),(40)].
```

Implementation Guide: *We perform simple "bookkeeping" of the $\mathbf{x}_R$ states that need to use their first estimates. During marginalization, we record the current estimate as the FEJ for the states in $\mathbf{x}_R$ which appeared in the nullspace, see Eq. (49), as they impact the consistency. Subsequently, factors connected to $\mathbf{x}_R$ should use FEJ for the states which have been FEJ'ed, $\hat{\mathbf{x}}_R(k)$, during Jacobian evaluation, while the others leverage the best estimate, $\hat{\mathbf{x}}_R(k')$ or $\hat{\mathbf{x}}_N(k')$.*

For example, for a feature measurement that has not been involved in the marginalization but is observed from a pose connected to the prior, the Jacobians should be evaluated using the *best* feature estimate and the *first* pose estimate. In Appendix C, we furnish a straightforward code example illustrating the application of FEJ with the camera-bearing measurement factor.

In what follows, we detail how marginalization strategies impact the connectivity of states to the marginal prior and FEJ.

3.1 KEEP

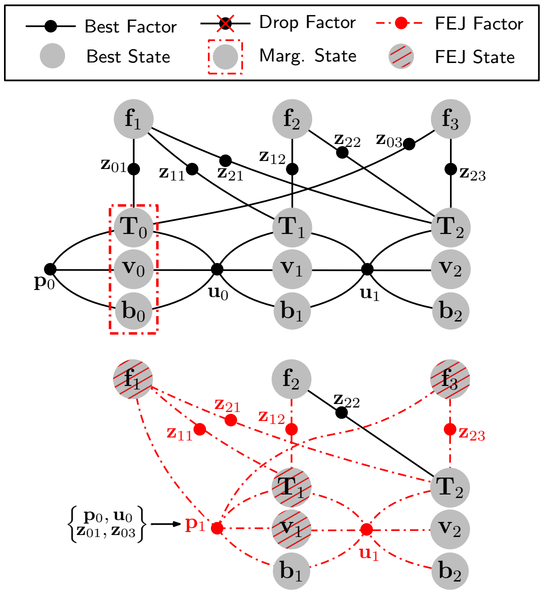


Figure 3: KEEP

The simplest case is marginalizing only the inertial states and keeping a map of environmental features. Shown in Figure 3, after marginalization, both features and the oldest inertial states become involved with the marginalized prior, thus requiring the fixing of many states. Specifically:

- **FEJ States**: $\mathbf{T}_1$, $\mathbf{v}_1$, $\mathbf{f}_1$, $\mathbf{f}_3$ since they connect to prior and are the "remaining" states (Markov blanket).
- **FEJ Factors**: All factors besides $\mathbf{z}_{22}$ since it connects to uninvolved states.

It is clear that while keeping features enables future observations to be included to improve accuracy, this method densifies the prior information which increases the computational cost, and requires the FEJ'ing of many states (e.g. most features).

3.2 DROP

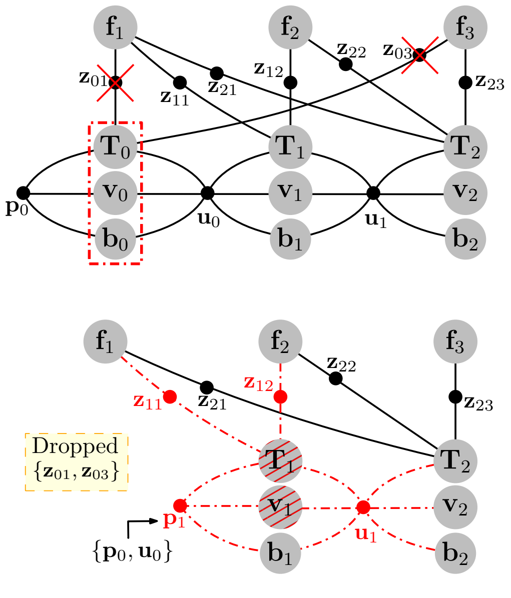


Figure 4: DROP

A method often leveraged is one which drops information to retain sparsity of the prior [10, 11, 12]. Figure 4 illustrates that by dropping $\mathbf{z}_{01}$ and $\mathbf{z}_{03}$, the new prior factor $\mathbf{p}_1$ is not involved with the feature states. Specifically:

- **FEJ States**: $\mathbf{T}_1$, $\mathbf{v}_1$ since they connect to prior and are the "remaining" states (Markov blanket).
- **FEJ Factors**: All factors besides $\mathbf{z}_{21}$, $\mathbf{z}_{22}$, $\mathbf{z}_{23}$ since they connect to uninvolved states.

This method offers the benefit of improved sparsity and avoids the need to perform FEJ on the feature states. However, it comes at the cost of a new sub-optimal problem containing less constrained information about the features.

3.3 MARG

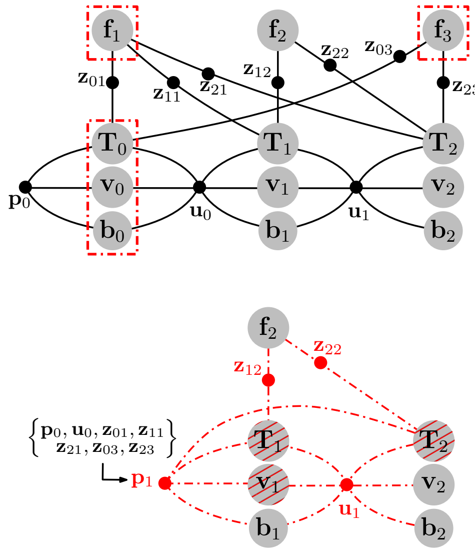


Figure 5: MARG

Another common case is to handle features that have been lost or to reduce the state size by marginalizing features alongside the inertial state. As shown in Figure **??**, this marginal prior now relates to all poses from which the features have been observed. Specifically:

- **FEJ States**: $\mathbf{T}_1$, $\mathbf{v}_1$, and $\mathbf{T}_2$ since they connect to prior and are the "remaining" states (Markov blanket).
- **FEJ Factors**: All factors have some portion of their Jacobians FEJ'ed

In this case, the prior density has increased, but the state size has decreased significantly, likely providing significant computational benefits. However, a key downside is that future feature observations cannot be leveraged (i.e., they will be treated as new features), and all poses need to be FEJ'ed, while the velocity $\mathbf{v}_2$, the remaining feature $\mathbf{f}_2$, and biases are still not needed.

3.4 CKLAM

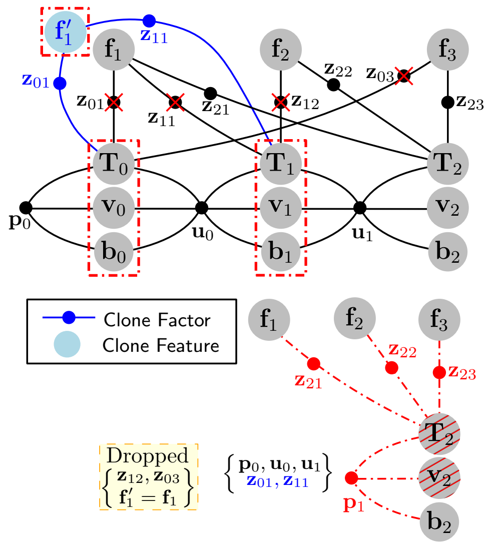


Figure 6: CKLAM

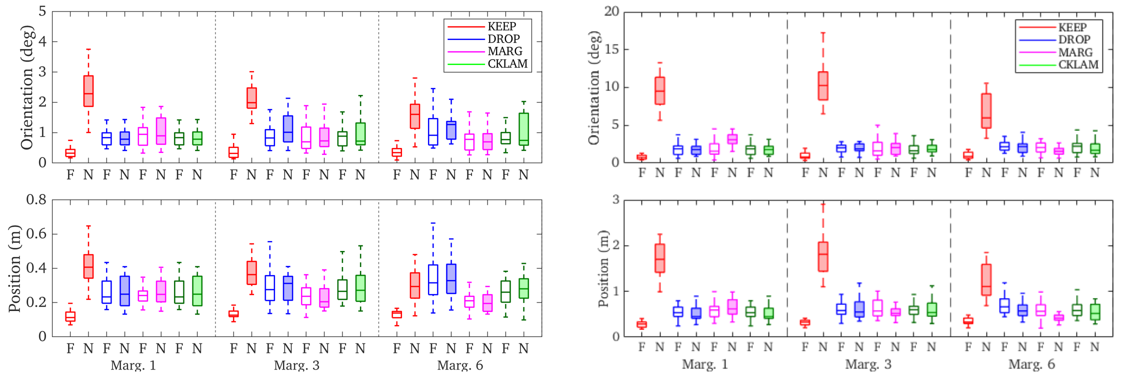
Another approach which we find promising is the CKLAM marginalization technique [13] (which has been recently adopted in [14]). As shown in Figure 6, to preserve the sparse structure of the optimization problem, feature $\mathbf{f}_1$ is duplicated along with its measurements $\mathbf{z}_{01}$ and $\mathbf{z}_{11}$ connecting to the to-be-marginalized inertial states, and then this new feature $\mathbf{f}'_1$ is marginalized alongside the inertial states. All other feature measurements (e.g., $\mathbf{z}_{12}$ and $\mathbf{z}_{03}$) related to the to-be-marginalized states – which do not sufficiently constrain their feature – are then dropped while introducing a loss of information. Specifically:

- **FEJ States**: $\mathbf{T}_2$, $\mathbf{v}_2$ since they connect to prior and are the "remaining" states (Markov blanket).
- **FEJ Factors**: All factors have some portion of their Jacobians FEJ'ed

CKLAM reduces to the DROP case, see Section 3.2, when only a single inertial state is marginalized (e.g. a sliding window), while for multiple states it enables the inclusion of feature observation information into the prior factor without increasing the computational complexity and thus minimizes information loss. This allows for both computational and accuracy gain for "shifting" window and keyframe-based VINS and makes CKLAM very alluring.

4 Numerical Study

We investigate how the different marginalization techniques are impacted by the use of FEJ and different shifting window sizes. The results are shown in Table 1 and Figure 7. In general, it can be seen that the KEEP method is able to have significant gains in accuracy and consistency when leveraging FEJ, while the other marginalization methods are less sensitive to the use of FEJ. In general FEJ is able to guarantee consistency and improve performance in most cases, thus we recommend its use. It is worth noting that this report presents more complete simulation results. For detailed simulation setups and discussions, we refer readers to the corresponding paper.



(a) 1 pixel noise (b) 3 pixel noise

Figure 7: Average ATE for 20 `Gore` dataset runs with different algorithms (see Table 1). FEJ denoted as "F" (empty) and No-FEJ denoted as "N" (shaded). Colors indices different marginalization methods.

Table 1: Average ATE over the 20 `Gore` dataset runs for different image noise levels (e.g. 1 and 3 pixel) and the number of marginalized inertial states. Time is reported just for optimization and marginalization (no covariance recovery).

σ	N		Algo.	ATE (deg/cm)	NEES (3/3)	Time (ms)
1 pixel image noise	marg. 1 clone	FEJ	KEEP	0.354 / 0.118	2.713 / 2.451	21.7 ± 8.1
			DROP	0.966 / 0.259	3.039 / 2.667	12.5 ± 4.6
			MARG	0.930 / 0.250	3.173 / 2.661	10.9 ± 4.5
			CKLAM	0.966 / 0.259	3.039 / 2.667	12.4 ± 4.5
		No-FEJ	KEEP	2.328 / 0.407	181.7 / 26.7	18.8 ± 6.6
			DROP	0.931 / 0.259	2.946 / 2.781	10.6 ± 3.3
			MARG	1.007 / 0.256	3.039 / 2.502	9.2 ± 3.8
			CKLAM	0.931 / 0.259	2.946 / 2.781	10.2 ± 3.1
	marg. 3 clones	FEJ	KEEP	0.380 / 0.132	3.057 / 2.733	8.8 ± 10.5
			DROP	0.919 / 0.289	2.661 / 3.103	5.2 ± 7.0
			MARG	0.816 / 0.235	2.979 / 2.619	4.3 ± 5.9
			CKLAM	0.907 / 0.280	2.664 / 3.079	5.1 ± 6.8
		No-FEJ	KEEP	2.123 / 0.376	155.3 / 22.1	8.2 ± 9.3
			DROP	1.104 / 0.299	3.027 / 3.010	4.7 ± 6.0
			MARG	0.879 / 0.233	3.037 / 2.476	3.9 ± 5.2
			CKLAM	1.002 / 0.287	2.952 / 2.926	4.5 ± 5.8
	marg. 6 clones	FEJ	KEEP	0.362 / 0.129	2.733 / 2.898	4.5 ± 7.6
			DROP	1.186 / 0.331	2.900 / 2.777	3.1 ± 6.2
			MARG	0.844 / 0.214	3.268 / 2.196	2.4 ± 5.1
			CKLAM	0.854 / 0.261	2.548 / 2.639	3.0 ± 6.0
		No-FEJ	KEEP	1.556 / 0.295	93.6 / 14.9	4.4 ± 7.1
			DROP	1.403 / 0.350	3.415 / 2.951	2.6 ± 5.1
			MARG	0.812 / 0.206	3.300 / 2.216	2.2 ± 4.5
			CKLAM	1.048 / 0.277	2.997 / 2.740	2.6 ± 5.1

σ	N		Algo.	ATE (deg/cm)	NEES (3/3)	Time (ms)
3 pixel image noise	marg. 1 clone	FEJ	KEEP	0.868 / 0.294	3.021 / 3.776	26.4 ± 9.2
			DROP	1.827 / 0.542	2.909 / 2.895	14.8 ± 4.4
			MARG	2.088 / 0.618	4.160 / 4.097	13.2 ± 4.7
			CKLAM	1.827 / 0.542	2.909 / 2.895	14.5 ± 4.5
		No-FEJ	KEEP	9.648 / 1.682	2226.5 / 115.1	21.6 ± 6.6
			DROP	1.940 / 0.510	2.701 / 2.670	12.1 ± 3.0
			MARG	3.143 / 0.657	4.504 / 3.572	10.0 ± 2.9
			CKLAM	1.940 / 0.510	2.701 / 2.670	12.1 ± 2.9
	marg. 3 clones	FEJ	KEEP	1.038 / 0.326	3.551 / 3.930	10.6 ± 12.9
			DROP	1.970 / 0.624	2.772 / 3.658	6.0 ± 7.9
			MARG	2.036 / 0.635	3.986 / 6.374	5.1 ± 6.9
			CKLAM	1.818 / 0.615	2.725 / 3.692	5.9 ± 7.8
		No-FEJ	KEEP	10.492 / 1.811	3000.2 / 137.0	8.5 ± 9.6
			DROP	2.210 / 0.626	3.371 / 3.639	5.8 ± 6.1
			MARG	2.066 / 0.541	3.343 / 3.480	4.2 ± 5.5
			CKLAM	2.104 / 0.601	3.244 / 3.462	4.9 ± 6.2
	marg. 6 clones	FEJ	KEEP	1.117 / 0.357	4.011 / 6.741	5.5 ± 9.6
			DROP	2.421 / 0.725	2.884 / 3.759	3.3 ± 6.6
			MARG	2.152 / 0.592	3.976 / 5.029	2.7 ± 5.6
			CKLAM	2.113 / 0.611	2.917 / 3.802	3.4 ± 6.8
		No-FEJ	KEEP	6.687 / 1.201	1381.9 / 57.6	4.5 ± 7.4
			DROP	2.194 / 0.629	3.013 / 3.170	2.7 ± 5.3
			MARG	1.748 / 0.443	3.414 / 2.893	2.3 ± 4.7
			CKLAM	2.072 / 0.554	3.101 / 3.167	2.8 ± 5.4

Appendix A: $\texttt{rank}(\mathbf{A}_{k'}^{\textbf{full}})$ and $\mathbb{N}(\mathbf{A}_{k'}^{\textbf{full}})$

We start by analyzing the full information matrix $\mathbf{A}_{k'}^{\text{full}}$, it can be rewrite as:

$$\mathbf{A}_{k'}^{\text{full}} = \sum_{k=0}^{k'-1} \mathbf{\Phi}_i(k')^\top \mathbf{Q}_i^{-1} \mathbf{\Phi}_i(k') + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_M} \mathbf{H}_{ij}(k')^\top \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k') + \sum_{\mathbf{z}_{ij}\in\mathcal{Z}_A} \mathbf{H}_{ij}(k')^\top \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k') \tag{53}$$

$$= \underbrace{\begin{bmatrix} \mathbf{\Phi}_0(k') \\ \vdots \\ \mathbf{\Phi}_{k'-1}(k') \\ \hline \vdots \\ \mathbf{H}_{ij}(k') \\ \vdots \end{bmatrix}^\top}_{\mathbf{\Xi}^\top(k')} \underbrace{\left[\begin{array}{cccccc} \mathbf{Q}_0^{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{Q}_{k'-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{R}_{ij}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots \end{array}\right]}_{\mathbf{S}} \underbrace{\begin{bmatrix} \mathbf{\Phi}_0(k') \\ \vdots \\ \mathbf{\Phi}_{k'-1}(k') \\ \hline \vdots \\ \mathbf{H}_{ij}(k') \\ \vdots \end{bmatrix}}_{\mathbf{\Xi}(k')} \tag{54}$$

$$= \mathbf{\Xi}^\top(k') \mathbf{S} \mathbf{\Xi}(k') \tag{55}$$

We can expend $\mathbf{\Xi}(k')$ to be:

$$\mathbf{\Xi}(k') = \begin{bmatrix} \mathbf{\Phi}_0(k') \\ \vdots \\ \mathbf{\Phi}_{k'-1}(k') \\ \hline \vdots \\ \mathbf{H}_{ij}(k') \\ \vdots \end{bmatrix} = \left[\begin{array}{ccccc} -\mathbf{\Phi}_{I_0}(k') & \mathbf{I}_r & \dots & \mathbf{0} & \mathbf{0}_{r\times F} \\ \mathbf{0} & \ddots & \ddots & \vdots & \mathbf{0}_{r\times F} \\ \mathbf{0} & \mathbf{0} & -\mathbf{\Phi}_{I_{k'-1}}(k') & \mathbf{I}_r & \mathbf{0}_{r\times F} \\ \hline \mathbf{H}_{x_0}(k') & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{f_1}(k') \\ \mathbf{0} & \mathbf{H}_{x_1}(k') & \mathbf{0} & \mathbf{0} & \mathbf{H}_{f_2}(k') \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \vdots \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{x_{k'}}(k') & \mathbf{H}_{f_g}(k') \end{array}\right] \tag{56}$$

It's worth noting that we slightly abuse the notation for the sake of brevity. Specifically, we have "group / stack " the measurements based on robot pose. If at timestamp t_i the robots observe l_i features, then $\mathbf{H}_{x_i}$ is an block vector, containing Jacobians with respect to robot state $\mathbf{x}_i$:

$$\mathbf{H}_{x_i} = \begin{bmatrix} \mathbf{H}_{x_i j_1} \\ \mathbf{H}_{x_i j_2} \\ \vdots \\ \mathbf{H}_{x_i j_{l_i}} \end{bmatrix}_{(m*l_i)\times r} \tag{57}$$

where m is the measurement size (i.e. $m = 2$ for camera bearing measurement, Eq. (13)) and $r = \texttt{dim}(\mathbf{x}_i)$ is the size of the robot state at a single timestamp. On the other hand, $\mathbf{H}_{f_i}$ contains l_i block rows, where each row contains the Jacobian with respect to jth feature:

$$\mathbf{H}_{f_i} = \begin{bmatrix} \mathbf{H}_{j_1} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_{j_2} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{H}_{j_{l_i}} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}_{(m*l_i)\times F} \tag{58}$$

where we have a total of g features represented in global and in 3D, the dimension of feature states can be denoted by $F := \texttt{dim}(\mathbf{x}_f) = 3g$. From the initial time t_0 to the final time $t_{k'}$, the size of the robot state is denoted by $R := \mathtt{dim}(\mathbf{x}_{0:k'}) = r * (k'-1)$. Using these dimensions, we can specify the dimensions of each block matrix in Eq. (56) as follows: $\mathtt{dim}(\mathbf{\Phi}_i) = r \times R$, $\mathtt{dim}(\mathbf{H}_{ij}) = m \times (R+F)$, $\mathtt{dim}(\mathbf{H}_{x_i}) = m \times R$, and $\mathtt{dim}(\mathbf{H}_{f_j}) = m \times F$.

It has been proved that the rank of the information matrix can be found as (see Lemma 1 [8]):

$$\mathtt{rank}(\mathbf{A}_{k'}^{\text{full}}) = \mathtt{rank}(\mathbf{\Xi}(k')) = k'r + \mathtt{rank}(\mathbf{M}_{k'}) \tag{59}$$

where $\mathbf{M}_{k'}$ can be found as:

$$\begin{aligned}\mathbf{M}_{k'} &= \begin{bmatrix}\mathbf{M}_{x_{k'}} & \mathbf{M}_{f_{k'}}\end{bmatrix} \qquad (60)\\ &= \begin{bmatrix} \mathbf{H}_{x_0}\mathbf{\Phi}_{I_0}(k') & \mathbf{H}_{f1}(k') \\ \mathbf{H}_{x_1}(k')\mathbf{\Phi}_{I_1}(k')\mathbf{\Phi}_{I_0}(k') & \mathbf{H}_{f2}(k') \\ \vdots & \vdots \\ \mathbf{H}_{x_{k'-1}}(k')\mathbf{\Phi}_{I'_k-1}(k')\cdots\mathbf{\Phi}_{I_1}(k')\mathbf{\Phi}_{I_0}(k') & \mathbf{H}_{f_{g-1}}(k') \\ \mathbf{H}_{x_{k'}}(k')\mathbf{\Phi}_{I'_k}(k')\mathbf{\Phi}_{I'_k-1}(k')\cdots\mathbf{\Phi}_{I_1}(k')\mathbf{\Phi}_{I_0}(k') & \mathbf{H}_{f_g}(k') \end{bmatrix}_{M\times(r+F)} \qquad (61)\end{aligned}$$

where M is the total size of feature measurements in the time period. One can note this is the observability matrix that has an Identity state translation matrix for the features which are zero dynamics. We can derive each block row as:

$$\mathbf{H}_{x_{k'}}(k')\mathbf{\Phi}_{I'_k}(k')\cdots\mathbf{\Phi}_{I_0}(k') = \mathbf{H}_{proj,k'}{}^{k'}_{G}\mathbf{R}\begin{bmatrix}\mathbf{\Gamma}_1 & -\mathbf{I}_3 & -(t_{k'}-t_0)\mathbf{I}_3 & \mathbf{\Gamma}_4 & \mathbf{\Gamma}_5\end{bmatrix} \tag{62}$$

$$\mathbf{H}_{f_j}(k') = \mathbf{H}_{proj,k'}{}^{k'}_{G}\mathbf{R}\begin{bmatrix}\mathbf{0} & \cdots & \mathbf{I}_3 & \cdots & \mathbf{0}\end{bmatrix} \tag{63}$$

$$\mathbf{\Gamma}_1 = \lfloor {}^G\mathbf{f}_j - {}^G\mathbf{p}_0 - {}^G\mathbf{v}_0 - \frac{1}{2}\mathbf{g}(t_{k'}-t_0)^2\rfloor {}^0_G\mathbf{R}^\top \tag{64}$$

$$\mathbf{\Gamma}_4 = \lfloor {}^G\mathbf{f}_j - {}^G\mathbf{p}_0\rfloor {}^{k'}_G\mathbf{R}^\top\mathbf{\Phi}_{14} - \mathbf{\Phi}_{24} \tag{65}$$

$$\mathbf{\Gamma}_5 = -\mathbf{\Phi}_{25} \tag{66}$$

Note that $\mathbf{M}_{k'}$ can be decomposed as:

$$\mathbf{M}_{k'} = \mathbf{D}_{k'}\mathbf{K}_{k'} \tag{67}$$

where $\mathbf{D}_{k'}$ is a block diagonal matrix:

$$\mathbf{D}_{k'} = \begin{bmatrix}\ddots & & \\ & \mathbf{H}_{proj,k'}{}^{k'}_{G}\mathbf{R} & \\ & & \ddots\end{bmatrix} \tag{68}$$

To find the rank of matrix $\mathbf{M}_{k'}$, we have (details are refer to 4.5.1 [15] and 3.2.2 [8]):

$$\mathtt{rank}(\mathbf{M}_{k'}) = \mathtt{rank}(\mathbf{K}_{k'}) \tag{69}$$

where the ith block row of $\mathbf{K}_{k'}$ can be computed as:

$$\mathbf{K}_{k'_i} = \left[\begin{array}{ccccc|ccccc}\mathbf{\Gamma}_1 & -\mathbf{I}_3 & -(t_{k'}-t_0)\mathbf{I}_3 & \mathbf{\Gamma}_4 & \mathbf{\Gamma}_5 & \mathbf{0} & \cdots & \mathbf{I}_3 & \cdots & \mathbf{0}\end{array}\right] \tag{70}$$

We can interpret that $\mathtt{rank}(\mathbf{K}_{k'}) = 3g+11$, where g is the total number of features [See Section [8], Section 3.3.2, Lemma 7]. The rank of the information matrix can thus be found as $\mathtt{rank}(\mathbf{A}_{k'}^{\text{full}}) =$

$k'r + 3g + 11$. The nullspace for $\mathbf{A}_{k'}^{\text{full}}$ can be computed as:

$$\mathbb{N}(\mathbf{A}_{k'}^{\text{full}}) = \begin{bmatrix} \mathbb{N}_{x_0} \\ \vdots \\ \mathbb{N}_{x_{k'}} \\ \mathbb{N}_{f_0} \\ \vdots \\ \mathbb{N}_{f_g} \end{bmatrix} = \left[\begin{array}{cc} \mathbf{0}_3 & {}^{0}_{G}\mathbf{R}(k')^{G}\mathbf{g} \\ \mathbf{I}_3 & -\lfloor {}^{G}\mathbf{p}_0(k') \rfloor {}^{G}\mathbf{g} \\ \mathbf{0}_3 & -\lfloor {}^{G}\mathbf{v}_0(k') \rfloor {}^{G}\mathbf{g} \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \hline \vdots & \vdots \\ \hline \mathbf{0}_3 & {}^{k'}_{G}\mathbf{R}(k')^{G}\mathbf{g} \\ \mathbf{I}_3 & -\lfloor {}^{G}\mathbf{p}_{k'}(k') \rfloor {}^{G}\mathbf{g} \\ \mathbf{0}_3 & -\lfloor {}^{G}\mathbf{v}_{k'}(k') \rfloor {}^{G}\mathbf{g} \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \hline \vdots & \vdots \\ \mathbf{I}_3 & -\lfloor {}^{G}\mathbf{f}_j(k') \rfloor {}^{G}\mathbf{g} \\ \vdots & \vdots \end{array}\right] \tag{71}$$

This shows that the nullspace is of dimension 4. Note that this analytical nullspace can be related to the ones well-studied in filter-based (e.g. an extended Kalman filter) systems [16, 17]. The ideal VINS has been proved to have 4 DoF unobservable [16], the above analysis thus demonstrates that if all states are linearized at the same time (e.g. batch-MAP), the linearized system has the same number of unobservable directions.

Appendix B: $\texttt{rank}(\mathbf{A}_{k'}^{\mathbf{marg}})$

The information matrix $\mathbf{A}_{k'}^{\mathrm{marg}}$ can be derived as:

$$
\begin{aligned}
\mathbf{A}_{k'}^{\mathrm{marg}} &= \sum_{k=0}^{m-1} \boldsymbol{\Phi}_i(k)^\top \mathbf{Q}_i^{-1} \boldsymbol{\Phi}_i(k) + \sum_{i=m}^{k'-1} \boldsymbol{\Phi}_i(k')^\top \mathbf{Q}_i^{-1} \boldsymbol{\Phi}_i(k') && (72) \\
&+ \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_M} \mathbf{H}_{ij}(k)^\top \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k) + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_A} \mathbf{H}_{ij}(k')^\top \mathbf{R}_{ij}^{-1} \mathbf{H}_{ij}(k') && (73) \\
&= \underbrace{\begin{bmatrix} \boldsymbol{\Phi}_0(k) \\ \vdots \\ \boldsymbol{\Phi}_{m-1}(k) \\ \hline \boldsymbol{\Phi}_m(k') \\ \vdots \\ \boldsymbol{\Phi}_{k'-1}(k') \\ \hline \vdots \\ \mathbf{H}_{ij}(k) \\ \vdots \\ \hline \vdots \\ \mathbf{H}_{ij}(k') \\ \vdots \end{bmatrix}^\top}_{\boldsymbol{\Xi}^\top(k,k')}
\underbrace{\left[\begin{array}{cccccc} \mathbf{Q}_0^{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{Q}_{k'-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{R}_{ij}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots \end{array}\right]}_{\mathbf{S}}
\underbrace{\begin{bmatrix} \boldsymbol{\Phi}_0(k) \\ \vdots \\ \boldsymbol{\Phi}_{m-1}(k) \\ \hline \boldsymbol{\Phi}_m(k') \\ \vdots \\ \boldsymbol{\Phi}_{k'-1}(k') \\ \hline \vdots \\ \mathbf{H}_{ij}(k) \\ \vdots \\ \hline \vdots \\ \mathbf{H}_{ij}(k') \\ \vdots \end{bmatrix}}_{\boldsymbol{\Xi}(k,k')} && (74) \\
&= \boldsymbol{\Xi}^\top(k,k') \mathbf{S} \boldsymbol{\Xi}(k,k') && (75)
\end{aligned}
$$

We can expend $\boldsymbol{\Xi}(k,k')$ to be:

$$
\boldsymbol{\Xi}(k,k') = \left[\begin{array}{cccccccc}
-\boldsymbol{\Phi}_{I_0}(k) & \mathbf{I} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \ddots & \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \vdots & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & -\boldsymbol{\Phi}_{I_{m-1}}(k) & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\
\hline
\mathbf{0} & \mathbf{0} & \mathbf{0} & -\boldsymbol{\Phi}_{I_m}(k') & \mathbf{I} & \cdots & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \ddots & \vdots & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\boldsymbol{\Phi}_{I'_k}(k') & \mathbf{I} & \mathbf{0} \\
\hline
\mathbf{H}_{x_0}(k) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{f_1}(k) \\
\mathbf{0} & \mathbf{H}_{x_1}(k) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \vdots \\
\mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \vdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{x_{m-1}}(k) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{f_{m-1}}(k) \\
\hline
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{x_m}(k') & \mathbf{0} & \mathbf{0} & \mathbf{H}_{f_m}(k') \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} & \vdots \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{x_{k'}}(k') & \mathbf{H}_{f_g}(k')
\end{array}\right]
$$

Similarly, the rank of the information matrix can be found as (See [8] Appendix **A**):

$$\texttt{rank}(\mathbf{A}_{k'}^{\text{marg}}) = k'r + \texttt{rank}(\mathbf{M}(k,k')) \tag{76}$$

where

$$\mathbf{M}(k,k') = \begin{bmatrix} \mathbf{M}_x(k,k') & \mathbf{M}_f(k,k') \end{bmatrix} \tag{77}$$

$$= \left[\begin{array}{cc} \mathbf{H}_{x_0}(k)\mathbf{\Phi}_{I_0}(k) & \mathbf{H}_{f1}(k) \\ \mathbf{H}_{x_1}(k)\mathbf{\Phi}_{I_1}(k)\mathbf{\Phi}_{I_0}(k) & \mathbf{H}_{f2}(k) \\ \vdots & \vdots \\ \mathbf{H}_{x_{m-1}}(k)\mathbf{\Phi}_{I_{m-1}}(k)\cdots\mathbf{\Phi}_{I_0}(k) & \mathbf{H}_{f_{m-1}}(k) \\ \hline \mathbf{H}_{x_m}(k')\mathbf{\Phi}_{I_m}(k')\cdots\mathbf{\Phi}_{I_0}(k') & \mathbf{H}_{f_m}(k') \\ \vdots & \vdots \\ \mathbf{H}_{x_{k'}}(k')\mathbf{\Phi}_{I_{k'}}(k')\cdots\mathbf{\Phi}_{I_0}(k') & \mathbf{H}_{f_g}(k') \end{array}\right] \tag{78}$$

Similarly, the rank of $\mathbf{M}(k,k')$ has been proved to be $3g+6$ by Dong-Si and Mourikis [8], where g is the number of features. Compare with $\texttt{rank}(\mathbf{A}_{k'}^{\text{full}})$ mentioned in Appendix A, we notice that when measurement Jacobians are estimated using two different state estimates for the state variables, the rank of the information matrix for the entire states has increased by 1. However, this increase in rank is not a result of any new measurements and is erroneous, leading to inconsistency. This issue has also been discussed and addressed in filter-based frameworks [18, 16]. In a filter, the state variables are not relinearized. As such, each state will have a different linearization point with respect to the others.

Appendix C: Example Bearing Factor Implementation

```
class Factor_ImageReproj : public ceres::CostFunction {
public:

  // Measurement observation of the feature (normalized coordinates)
  Eigen::Vector2d uv_meas;

  // Sqrt information matrix for this measurement
  Eigen::Matrix<double, 2, 2> sqrtQ;

  // FEJ values for each parameter (valid if is_fej is true)
  // NOTE: is_fej should be set when performing marginalization
  std::vector<double *> x_fej;
  std::vector<bool *> x_is_fej;

  // Camera to IMU transformation calibration (should be non-zero)
  Eigen::Matrix3d R_ItoC = Eigen::Matrix3d::Identity();
  Eigen::Vector3d p_IinC = Eigen::Vector3d::Zero();

  Factor_ImageReproj(const Eigen::Vector2d &uv_meas_, double pix_sigma_,
                     const std::vector<double *> &x_fej_,
                     const std::vector<bool *> &x_is_fej_)
                     : uv_meas(uv_meas_), x_fej(x_fej_), x_is_fej(x_is_fej_) {

    // Square root information inverse
    sqrtQ = Eigen::Matrix<double, 2, 2>::Identity();
    sqrtQ(0, 0) *= 1.0 / pix_sigma;
    sqrtQ(1, 1) *= 1.0 / pix_sigma;

    // Setup ceres
    set_num_residuals(2);
    mutable_parameter_block_sizes()->push_back(4); // q_GtoIi
    mutable_parameter_block_sizes()->push_back(3); // p_IiinG
    mutable_parameter_block_sizes()->push_back(3); // p_FinG

  }

  virtual ~Factor_ImageReproj() {}

  bool Evaluate(double const *const *parameters, double *residuals,
                     double **jacobians) const override {

    // Recover the current state from our parameters
    // NOTE: these are the best estimates of our system state
    Eigen::Vector4d q_GtoIi = Eigen::Map<const Eigen::Vector4d>(parameters[0]);
    Eigen::Matrix3d R_GtoIi = ov_core::quat_2_Rot(q_GtoIi);
    Eigen::Vector3d p_IiinG = Eigen::Map<const Eigen::Vector3d>(parameters[1]);
    Eigen::Vector3d p_FinG = Eigen::Map<const Eigen::Vector3d>(parameters[2]);

    // Transform the feature into the current camera frame of reference
    Eigen::Vector3d p_FinIi = R_GtoIi * (p_FinG - p_IiinG);
    Eigen::Vector3d p_FinCi = R_ItoC * p_FinIi + p_IinC;

    // Normalized projected feature bearing
    Eigen::Vector2d uv_norm;
    uv_norm << p_FinCi(0) / p_FinCi(2), p_FinCi(1) / p_FinCi(2);

    // Compute residual
    // NOTE: we make this negative ceres cost function is only min||f(x)||^2
    // NOTE: we have found the derivative of uv_meas = f(x) + n and thus
    // NOTE: reformulate into a zero error constraint 0 = f(x) + n - uv_meas
    Eigen::Vector2d res = uv_norm - uv_meas;
    res = sqrtQ * res;
    residuals[0] = res(0);
    residuals[1] = res(1);

    // Compute jacobians if requested by ceres
    if (jacobians) {

      // get the fej value if we have them
      if (x_is_fej.at(0)[0]) {
        q_GtoIi = Eigen::Map<const Eigen::Vector4d>(x_fej[0]);
        R_GtoIi = ov_core::quat_2_Rot(q_GtoIi);
      }
      if (x_is_fej.at(1)[0]) {
        p_IiinG = Eigen::Map<const Eigen::Vector3d>(x_fej[1]);
      }
      if (x_is_fej.at(2)[0]) {
        p_FinG = Eigen::Map<const Eigen::Vector3d>(x_fej[2]);
      }

      // Recompute with FEJ'ed state values
      p_FinIi = R_GtoIi * (p_FinG - p_IiinG);
      p_FinCi = R_ItoC * p_FinIi + p_IinC;
      uv_norm << p_FinCi(0) / p_FinCi(2), p_FinCi(1) / p_FinCi(2);

      // Pinhole projection Jacobian
      H_dz_dzn = sqrtQ * H_dz_dzn;
      H_dz_dzeta = sqrtQ * H_dz_dzeta;
      Eigen::MatrixXd H_dzn_dpfc = Eigen::MatrixXd::Zero(2, 3);
      H_dzn_dpfc << 1.0 / p_FinCi(2), 0, -p_FinCi(0) / std::pow(p_FinCi(2), 2),
                    0, 1.0 / p_FinCi(2), -p_FinCi(1) / std::pow(p_FinCi(2), 2);
      Eigen::MatrixXd H_dz_dpfc = sqrtQ * H_dzn_dpfc;

      // Jacobian wrt q_GtoIi
      if (jacobians[0]) {
        Eigen::Map<Eigen::Matrix<double, 2, 4, Eigen::RowMajor>>
            jacobian(jacobians[0]);
        jacobian.block(0, 0, 2, 3) =
            H_dz_dpfc * R_ItoC * ov_core::skew_x(p_FinIi);
        jacobian.block(0, 3, 2, 1).setZero();
      }

      // Jacobian wrt p_IiinG
      if (jacobians[1]) {
        Eigen::Map<Eigen::Matrix<double, 2, 3, Eigen::RowMajor>>
            jacobian(jacobians[1]);
        jacobian.block(0, 0, 2, 3) = -H_dz_dpfc * R_ItoC * R_GtoIi;
      }

      // Jacobian wrt feature p_FinG
      if (jacobians[2]) {
        Eigen::Map<Eigen::Matrix<double, 2, 3, Eigen::RowMajor>>
            jacobian(jacobians[2]);
        jacobian.block(0, 0, 2, 3) = H_dz_dpfc * R_ItoC * R_GtoIi;
      }
    }
    return true;
  }
};
```

References

[1] Nikolas Trawny and Stergios I. Roumeliotis. *Indirect Kalman Filter for 3D Attitude Estimation.* Tech. rep. University of Minnesota, Dept. of Comp. Sci. & Eng., Mar. 2005.

[2] Christoph Hertzberg, René Wagner, Udo Frese, and Lutz SchröDer. "Integrating Generic Sensor Fusion Algorithms with Sound State Representations Through Encapsulation of Manifolds". In: *Information Fusion* 14.1 (Jan. 2013), pp. 57–77. ISSN: 1566-2535.

[3] Averil B. Chatfield. *Fundamentals of High Accuracy Inertial Navigation.* AIAA, 1997.

[4] Yulin Yang, B. P. W. Babu, Chuchu Chen, Guoquan Huang, and Liu Ren. "Analytic Combined IMU Integrator for Visual-Inertial Navigation". In: *Proc. of the IEEE International Conference on Robotics and Automation.* Paris, France, 2020.

[5] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "Closed-form Preintegration Methods for Graph-based Visual-Inertial Navigation". In: *International Journal of Robotics Research* 38.5 (2019), pp. 563–586.

[6] Patrick Geneva, Kevin Eckenhoff, Woosik Lee, Yulin Yang, and Guoquan Huang. "OpenVINS: A Research Platform for Visual-Inertial Estimation". In: *Proc. of the IEEE International Conference on Robotics and Automation.* Paris, France, 2020. URL: https://github.com/rpng/open_vins.

[7] Tue-Cuong Dong-Si and Anastasios I Mourikis. "Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis". In: *2011 IEEE International Conference on Robotics and Automation.* IEEE. 2011, pp. 5655–5662.

[8] Tue-Cuong Dong-Si and Anastasios I. Mourikis. *Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis.* Tech. rep. University of California, Riverside, 2010. URL: https://intra.ece.ucr.edu/~mourikis/tech_reports/fixed_lag.pdf.

[9] Guoquan Huang, Anastasios I. Mourikis, and Stergios I. Roumeliotis. "An Observability Constrained Sliding Window Filter for SLAM". In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems.* San Francisco, CA, Sept. 2011, pp. 65–72. DOI: 10.1109/IROS.2011.6095161.

[10] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. "Keyframe-based visual–inertial odometry using nonlinear optimization". In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334.

[11] T. Qin, P. Li, and S. Shen. "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator". In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 1004–1020.

[12] Vladyslav Usenko, Nikolaus Demmel, David Schubert, Jörg Stückler, and Daniel Cremers. "Visual-inertial mapping with non-linear factor recovery". In: *IEEE Robotics and Automation Letters* 5.2 (2019), pp. 422–429.

[13] Esha D Nerurkar, Kejian J Wu, and Stergios I Roumeliotis. "C-KLAM: Constrained keyframe-based localization and mapping". In: *2014 IEEE international conference on robotics and automation (ICRA).* IEEE. 2014, pp. 3638–3643.

[14] Haomin Liu, Mingyu Chen, Guofeng Zhang, Hujun Bao, and Yingze Bao. "ICE-BA: Incremental, consistent and efficient bundle adjustment for visual-inertial slam". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 2018, pp. 1974–1982.

[15] Carl D Meyer. *Matrix analysis and applied linear algebra.* Vol. 71. Siam, 2000.

[16] J.A. Hesch, D.G. Kottas, S.L. Bowman, and S.I. Roumeliotis. “Consistency Analysis and Improvement of Vision-aided Inertial Navigation”. In: *IEEE Transactions on Robotics* 30.1 (2013), pp. 158–176.

[17] Chuchu Chen and Guoquan Huang. *FEJ2-EKF: A Consistent Estimator for SLAM (technical report)*. Tech. rep. RPNG-2022-FEJ2. University of Delaware, 2022. URL: http://chuchuchen.net/techreport/fej2.pdf.

[18] Guoquan Huang, Anastasios I. Mourikis, and Stergios I. Roumeliotis. “Observability-based Rules for Designing Consistent EKF SLAM Estimators”. In: *International Journal of Robotics Research* 29.5 (Apr. 2010), pp. 502–528. DOI: 10.1177/0278364909353640.