Null-Space-based Marginalization: Analysis and Algorithm

Yulin Yang - yuyang@udel.edu
James Maley - james.m.laley2.civ@mail.mil
Guoquan Huang - ghuang@udel.edu

Department of Mechanical Engineering
University of Delaware, Delaware, USA

RPNG

Robot Perception and Navigation Group (RPNG)
Derivation - RPNG-2017-001
Last Updated - Oct. 15, 2017

Contents

Abstract

In SLAM, the size of the state vector tends to grow when exploring unknown environments, causing the ever-increasing computational complexity. To reduce the computational cost, one needs to continuously marginalize part of previous states (features and/or poses). This can be achieved by using either the null-space operation or Schur complement based marginalization. The former was originally developed particularly for visual-inertial odometry (VIO), while the latter is the standard approach to reduce the state vector in bundle adjustment (BA). In this paper, for the first time ever, we prove that under mild assumptions (i.i.d. white Gaussian noise model and same linearization points) these two techniques retain the same amount of information about the state, which is validated with real-world experiments. Moreover, based on this key insight, we derive analytically the left null-space expressions for multi-state constraint Kalman filter (MSCKF)-based VIO, which is verified through Monte-Carlo simulations.

1 Introduction

It is essential for robots to perform SLAM when navigating in the unknown environments. However, a SLAM solution typically suffers from ever-increasing computational cost. To address this issue, the marginalization technique is often used. One common approach is to use the *Schur complement* to marginalize out part of the states while preserving all information about the remaining states [1, 2, 3, 4, 5, 6]. For example, in visual-inertial navigation systems (VINS) [1, 2, 3], visual point features are continuously marginalized by the Schur complement to prevent the state vector from growing too large while retaining relative camera motion constraints, which are then fused with (preintegrated) IMU measurements. In the graph-based SLAM, Schur complement is also employed to reduce the graph by marginalizing some of the graph nodes (features and/or poses) [4, 5, 6]. Alternatively, the *null-space* based marginalization can be used, which was originally developed in the multi-state constraint Kalman filter (MSCKF) [7] particularly for visual-inertial odometry (VIO) and recently has been applied to different VINS [7, 8, 9, 10, 11, 12, 13, 14, 15, 16]. The basic idea is to use the left null-space of the measurement Jacobian to marginalize feature components from the state vector. Note that a similar idea has also been employed in graph-based VINS [17] and our recent work on acoustic underwater navigation [16].

While the aforementioned work has shown that the null-space marginalization is efficient. In this work, we move one step further and for the first time ever, establish a link between the null-space operation and the Schur compliment. Besides using the null-space marginalization for features, we also adapt the null-space marginalization for both poses and features. In particular, the main theoretical contributions of this work include:

- We analytically show that the null-space operation and the Schur complement operation for feature marginalization retain the same amount of information, under the mild assumptions of independent and identically distributed (i.i.d.) white Gaussian noise and using the same linearization points, which motivates us to apply the null-space operation for feature marginalization in graph-based SLAM.
- We show that the null-space-based marginalization can be applicable for both poses and features under the white Gaussian noise assumption.
- We analytically drive the well-structured null-space expression for commonly-used sensors such as stereo and RGBD cameras, which is shown to perform better than the numerically-computed counterpart.

The rest of the paper is structured as follows: After briefly overview of the null space and the Schur complement operations in next section, we present our main result in Section III that the two marginalization techniques are equivalent up to certain mild conditions. In Section IV we apply the null space operation to both pose and feature marginalization, and in Section V we derive the analytical null space expressions for

MSCKF. Our analysis and algorithms are validated with simulations and experiments in Section VI. Section VII concludes the paper and outlines the future work.

2 Marginalization

In this section, we briefly explain the null-space marginalization and Schur complement operation in the context of SLAM, which will form the basis for our ensuing analysis. In particular, the SLAM state of a robot exploring an unknown environment propagates according to the motion model:

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{w}_k \tag{1}$$

where $\mathbf{x}_k$ represents the robot poses at time step k with $k \in 1 \dots K$, $\mathbf{u}_k$ represents measurements from odometry, inertial or other motion sensors, and $\mathbf{w}_k$ is additive white Gaussian noise with covariance $\mathbf{Q}_k$.The robot also observes features based on the measurement model:

$$\mathbf{z}_{kj} = \mathbf{h}_k(\mathbf{x}_k, \mathbf{x}_{\mathbf{f}_j}) + \mathbf{n}_{kj} \tag{2}$$

where $\mathbf{z}_{kj}$ denotes the measurements to feature $\mathbf{x}_{\mathbf{f}_j}$ ($j \in 1 \dots N$) at time step k, and $\mathbf{n}_{kj}$ represents the white Gaussian noise with covariance $\mathbf{R}_{kj}$.

Based on (1) and (2), a SLAM solution is to estimate both robot pose $\mathbf{x}_k$ and all observed features $\mathbf{x}_{\mathbf{f}_j}(j = 1 \dots N)$. clearly, if N is large (which is often the case for visual navigation), the estimation process of SLAM may become cost prohibitive. Thus, we will marginalize all the N features.

2.1 Null-Space Marginalization

We first review the null-space marginalization that was introduced in the MSCKF [7]. Specifically, the state vector of the MSCKF is given by:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_I^\top & \mathbf{x}_{C_1}^\top & \mathbf{x}_{C_2}^\top & \dots & \mathbf{x}_{C_m}^\top \end{bmatrix}^\top \tag{3}$$

where $\mathbf{x}_I$ represents the current IMU states, and $\mathbf{x}_{C_i}$, $i \in 1 \dots m$ represents the cloned robot (sensor) poses in m previous time steps when the sensor get measurements according to (2).

During MSCKF update, we assume a feature $\mathbf{x}_{\mathbf{f}_j}$ has been observed m_j ($m_j \leq m$) times by the robot within the cloned state window. With all these measurements, we can compute feature estimates $\widehat{\mathbf{x}}_{\mathbf{f}_j}$. By linearizing the nonlinear measurement model (2) about the current state estimate $\widehat{\mathbf{x}}$ and feature estimates $\widehat{\mathbf{x}}_{\mathbf{f}_j}$, we can stack all the linearized measurements and multiply by the left null space $\mathbf{U}$ of feature Jacobian $\mathbf{H}_{\mathbf{f}j}$ as follows:

$$\underbrace{\mathbf{U}^\top \begin{bmatrix} \widetilde{\mathbf{z}}_{1j} \\ \vdots \\ \widetilde{\mathbf{z}}_{ij} \\ \vdots \\ \widetilde{\mathbf{z}}_{m_j j} \end{bmatrix}}_{\widetilde{\mathbf{z}}^{(j)}} \simeq \underbrace{\mathbf{U}^\top \begin{bmatrix} \mathbf{H}_{1j} \\ \vdots \\ \mathbf{H}_{ij} \\ \vdots \\ \mathbf{H}_{m_j j} \end{bmatrix}}_{\mathbf{H}_{\mathbf{x}}^{(j)}} \widetilde{\mathbf{x}} + \underbrace{\mathbf{U}^\top \begin{bmatrix} \mathbf{H}_{\mathbf{f}1j} \\ \vdots \\ \mathbf{H}_{\mathbf{f}ij} \\ \vdots \\ \mathbf{H}_{\mathbf{f}m_j j} \end{bmatrix}}_{\mathbf{H}_{\mathbf{f}j}} \widetilde{\mathbf{x}}_{\mathbf{f}_j} + \underbrace{\mathbf{U}^\top \begin{bmatrix} \mathbf{n}_{1j} \\ \vdots \\ \mathbf{n}_{ij} \\ \vdots \\ \mathbf{n}_{m_j j} \end{bmatrix}}_{\mathbf{n}^{(j)}} \tag{4}$$

where $i = 1 \dots m_j$, $\widetilde{\mathbf{z}}_{ij} = \mathbf{z}_{ij} - \mathbf{h}_i(\widehat{\mathbf{x}}_{C_i}, \widehat{\mathbf{x}}_{\mathbf{f}_j})$ denotes the measurement residual for feature $\mathbf{x}_{\mathbf{f}_j}$ at the i-th camera pose within the window, $\mathbf{H}_{ij}$ and $\mathbf{H}_{\mathbf{f}ij}$ denote the state and feature Jacobians of the measurement $\mathbf{z}_{ij}$. In [7], $\mathbf{n}_{ij}$ is white Gaussian noise with covariances as $\mathbf{R}_{ij} = \sigma^2 \mathbf{I}_{\mathbf{R}_{ij}}$, where $\mathbf{I}_{\mathbf{R}_{ij}}$ denotes identity matrix with same dimension as $\mathbf{R}_{ij}$. Since that $\mathbf{U}^\top \mathbf{H}_{\mathbf{f}j} = \mathbf{0}$, the feature $\widetilde{\mathbf{x}}_{\mathbf{f}_j}$ is marginalized from (4) by null space operation. Finally, stacking all the n feature measurements observed within the sliding window, we obtain:

$$\widetilde{\mathbf{z}}^o = \mathbf{H}^o \widetilde{\mathbf{x}} + \mathbf{n}^o \tag{5}$$

where $\widetilde{\mathbf{z}}^o$, $\mathbf{H}^o$ and $\mathbf{n}^o$ are all stacked measurement residuals, state Jacobians and measurement noise from $\widetilde{\mathbf{z}}^{(j)}$, $\mathbf{H}_{\mathbf{x}}^{(j)}$ and $\mathbf{n}^{(j)}$ ($j = 1 \dots n$), respectively [see (4)]. Up to this point, we essentially have marginalized all the features by the null-space operation (i.e., null-space marginalization), and obtained the inferred measurement equations (5) with only robot states involved, which is ready for EKF update.

2.2 Schur Complement

We optimally formulate the SLAM problem as maximum likelihood estimation (MLE). Assuming that all the measurements are independent, under the Gaussian noise assumption, this MLE problem is equivalent to a weighted nonlinear least squares problem [2, 18]:

$$\min_{\mathbf{x},\mathbf{x_f}} \sum_{k=1}^{K} \|\mathbf{x}_{k+1} - \mathbf{f}_k(\mathbf{x}_k)\|^2_{\mathbf{Q}_k^{-1}} + \sum_{k,j} \left\|\mathbf{z}_{kj} - \mathbf{h}_k(\mathbf{x}_k, \mathbf{x}_{\mathbf{f}_j})\right\|^2_{\mathbf{R}_{kj}^{-1}} \tag{6}$$

where $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1^\top \dots \mathbf{x}_K^\top\end{bmatrix}^\top$ denotes all the robot poses, $\mathbf{x_f} = \begin{bmatrix}\mathbf{x}_{\mathbf{f}_1}^\top \dots \mathbf{x}_{\mathbf{f}_N}^\top\end{bmatrix}^\top$ denotes all the landmarks. Iterative methods, such as Gaussian-Newton or Levenberg-Marquardt algorithms, are often used to solve (6) for $\widehat{\mathbf{x}}$ and $\widehat{\mathbf{x}}_\mathbf{f}$.Upon this linearization point $(\widehat{\mathbf{x}}, \widehat{\mathbf{x}}_\mathbf{f})$, the system information matrix Σ can be computed as:

$$\Sigma = \sum_{k=1}^{K} \mathbf{F}_k^\top \mathbf{Q}_k^{-1} \mathbf{F}_k + \sum_{k,j} \mathbf{H}_{kj}^\top \mathbf{R}_{kj}^{-1} \mathbf{H}_{kj} =: \begin{bmatrix}\Sigma_{\mathbf{xx}} & \Sigma_{\mathbf{xf}} \\ \Sigma_{\mathbf{fx}} & \Sigma_{\mathbf{ff}}\end{bmatrix} \tag{7}$$

where $\mathbf{F}_k$ and $\mathbf{H}_{kj}$ are the Jacobians of the system model (1) and (2) with respect to $\mathbf{x}_k$ and $\mathbf{x}_{\mathbf{f}_j}$, respectively, and Σ is partitioned based on the dimension of $\mathbf{x}$ and $\mathbf{x_f}$. Since we are more interested in the robot states, the information matrix $\Sigma_\mathbf{x}$ of the robot states can be easily obtained through Schur complement by marginalizing feature $\mathbf{x_f}$:

$$\Sigma_\mathbf{x} = \Sigma_{\mathbf{xx}} - \Sigma_{\mathbf{xf}} \Sigma_{\mathbf{ff}}^{-1} \Sigma_{\mathbf{fx}} \tag{8}$$

which implies that in spite of the feature marginalization, the remaining information matrix $\Sigma_\mathbf{x}$ contains all the information inherited from (7) about $\mathbf{x}$.

3 Analysis of Null-Space Marginalization

In this section, from the information perspective, we analytically study the connection between null-space marginalization and Schur complement based marginalization. To make the presentation concise, we assume a simple SLAM scenario: A robot takes measurements of feature $\mathbf{x_f}$ at poses $\mathbf{x_1}$ and $\mathbf{x_2}$ respectively while this result can be generalized to any SLAM cases:

$$\mathbf{x}_2 = \mathbf{f}_1(\mathbf{x_1}) + \mathbf{w}_1 \tag{9}$$

$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x_k}, \mathbf{x_f}) + \mathbf{n}_k \tag{10}$$

where $k = 1, 2$. Given the prior knowledge of $\mathbf{x}_1 \sim \mathcal{N}(\widehat{\mathbf{x}}_1, \mathbf{P}_1)$, we can solve the SLAM problem and get the estimate of robot pose by marginalizing the feature. Given the linearization point $(\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2, \widehat{\mathbf{x}}_\mathbf{f})$, the linearized system can be written as:

$$\begin{cases} \widetilde{\mathbf{x}}_2 \simeq \mathbf{F}_1 \widetilde{\mathbf{x}}_1 + \mathbf{w}_1 \\ \widetilde{\mathbf{z}}_k \simeq \mathbf{H}_k \widetilde{\mathbf{x}}_k + \mathbf{H}_{\mathbf{f}k} \widetilde{\mathbf{x}}_\mathbf{f} + \mathbf{n}_k \end{cases} \tag{11}$$

We will marginalize features with null-space (NS) and Schur complement (SC) operations and compare the resulting information matrix for robot poses.

3.1 Information Matrix After Null-Space Marginalization

We solve this problem within the MSCKF framework. The system propagates from $\mathbf{x}_1$ to $\mathbf{x}_2$ according to (1) and clones the state of $\mathbf{x}_1$. At the same time, the system stacks all the measurements and updates at $\mathbf{x}_2$. The covariance matrix for the robot poses after the propagation is:

$$\mathbf{P}(\mathbf{x}_1, \mathbf{x}_2) = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_1 \mathbf{F}_1^\top \\ \mathbf{F}_1 \mathbf{P}_1 & \mathbf{F}_1 \mathbf{P}_1 \mathbf{F}_1^\top + \mathbf{Q}_1 \end{bmatrix} \tag{12}$$

After linearization of the measurement model, we perform the null-space operation:

$$\begin{bmatrix}\widetilde{\mathbf{z}}_1 \\ \widetilde{\mathbf{z}}_2\end{bmatrix} = \begin{bmatrix}\mathbf{H}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_2\end{bmatrix} \begin{bmatrix}\widetilde{\mathbf{x}}_1 \\ \widetilde{\mathbf{x}}_2\end{bmatrix} + \begin{bmatrix}\mathbf{H}_{\mathbf{f}1} \\ \mathbf{H}_{\mathbf{f}2}\end{bmatrix} \widetilde{\mathbf{x}}_\mathbf{f} + \begin{bmatrix}\mathbf{n}_1 \\ \mathbf{n}_2\end{bmatrix} \tag{13}$$

$$\Rightarrow \mathbf{U}_n^\top \begin{bmatrix}\widetilde{\mathbf{z}}_1 \\ \widetilde{\mathbf{z}}_2\end{bmatrix} = \mathbf{U}_n^\top \mathbf{H}_\mathbf{x} \begin{bmatrix}\widetilde{\mathbf{x}}_1 \\ \widetilde{\mathbf{x}}_2\end{bmatrix} + \mathbf{U}_n^\top \begin{bmatrix}\mathbf{n}_1 \\ \mathbf{n}_2\end{bmatrix} \tag{14}$$

where $\mathbf{H_x} = \begin{bmatrix} \mathbf{H}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_2 \end{bmatrix}$ and $\mathbf{H_f} = \begin{bmatrix} \mathbf{H_{f1}} \\ \mathbf{H_{f2}} \end{bmatrix}$. $\mathbf{U_n}$ represents the left null space for $\mathbf{H_f}$, and can be computed by QR:

$$\mathbf{H_f} = \begin{bmatrix} \mathbf{U_e} & \mathbf{U_n} \end{bmatrix} \begin{bmatrix} \mathbf{R}_\Delta \\ \mathbf{0} \end{bmatrix} = \mathbf{U_e R}_\Delta \tag{15}$$

where $\begin{bmatrix} \mathbf{U_e} & \mathbf{U_n} \end{bmatrix}$ is a unitary matrix and $\mathbf{R}_\Delta$ is an upper triangular matrix. We can get the information matrix for the robot poses $(\mathbf{x}_1, \mathbf{x}_2)$ after the EKF update as:

$$\Sigma_{\text{NS}} = \underbrace{\begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_1\mathbf{F}_1^\top \\ \mathbf{F}_1\mathbf{P}_1 & \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1^\top + \mathbf{Q}_1 \end{bmatrix}^{-1}}_{\Sigma_1^{(\text{NS})}} + \underbrace{\mathbf{H_x}^\top \mathbf{U_n} (\mathbf{U_n}^\top \mathbf{R} \mathbf{U_n})^{-1} \mathbf{U_n}^\top \mathbf{H_x}}_{\Sigma_2^{(\text{NS})}} \tag{16}$$

where $\Sigma_1^{(\text{NS})}$ and $\Sigma_2^{(\text{NS})}$ denote the information from the motion model (9) and the measurement model (10), respectively. $\mathbf{R} = \begin{bmatrix} \mathbf{R}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_2 \end{bmatrix}$ denotes the stacked noise covariances.

3.2 Information Matrix After Schur Complement

We now formulate the SLAM problem as a MLE where cost function is given by:

$$\mathbf{J} = \|\mathbf{x_1} - \widehat{\mathbf{x}}_1\|^2_{\mathbf{P}_1^{-1}} + \|\mathbf{x_2} - \mathbf{f}_1(\mathbf{x_1})\|^2_{\mathbf{Q}_1^{-1}} + \sum_{k=1}^{2} \|\mathbf{z_k} - \mathbf{h_k}(\mathbf{x_k}, \mathbf{x_f})\|^2_{\mathbf{R}_k^{-1}} \tag{17}$$

The information matrix $\Sigma_\mathbf{J}$ for the system can be computed according to (7), and for simplicity, we partitioned $\Sigma_\mathbf{J}$ as:

$$\Sigma_\mathbf{J} = \begin{bmatrix} \Sigma_{\mathbf{J}1} & \Sigma_{\mathbf{J}2} \\ \Sigma_{\mathbf{J}3} & \Sigma_{\mathbf{J}4} \end{bmatrix} \tag{18}$$

where:

$$\Sigma_{\mathbf{J}1} = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_1\mathbf{F}_1^\top \\ \mathbf{F}_1\mathbf{P}_1 & \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1^\top + \mathbf{Q}_1 \end{bmatrix}^{-1} + \mathbf{H_x}^\top \mathbf{R}^{-1} \mathbf{H_x} \tag{19}$$

$$\Sigma_{\mathbf{J}2} = \begin{bmatrix} \mathbf{H}_1^\top \mathbf{R}_1^{-1} \mathbf{H_{f1}} \\ \mathbf{H}_2^\top \mathbf{R}_2^{-1} \mathbf{H_{f2}} \end{bmatrix} = \mathbf{H_x}^\top \mathbf{R}^{-1} \mathbf{H_f} \tag{20}$$

$$\Sigma_{\mathbf{J}3} = \begin{bmatrix} \mathbf{H_{f1}}^\top \mathbf{R}_1^{-1} \mathbf{H}_1 & \mathbf{H_{f2}}^\top \mathbf{R}_2^{-1} \mathbf{H}_2 \end{bmatrix} = \mathbf{H_f}^\top \mathbf{R}^{-1} \mathbf{H_x} \tag{21}$$

$$\Sigma_{\mathbf{J}4} = \mathbf{H_{f1}}^\top \mathbf{R}_1^{-1} \mathbf{H_{f1}} + \mathbf{H_{f2}}^\top \mathbf{R}_2^{-1} \mathbf{H_{f2}} = \mathbf{H_f}^\top \mathbf{R}^{-1} \mathbf{H_f} \tag{22}$$

We marginalize the feature $\mathbf{x_f}$ with Schur complement and obtain the information matrix of the robot poses $(\mathbf{x}_1, \mathbf{x}_2)$ as:

$$\begin{aligned} \Sigma_{\text{SC}} &= \Sigma_{\mathbf{J}1} - \Sigma_{\mathbf{J}2} \Sigma_{\mathbf{J}4}^{-1} \Sigma_{\mathbf{J}3} \\ &= \underbrace{\begin{bmatrix} \mathbf{P}_1^{-1} + \mathbf{F}_1^\top \mathbf{Q}_1^{-1} \mathbf{F}_1 & -\mathbf{F}_1^\top \mathbf{Q}_1^{-1} \\ -\mathbf{Q}_1^{-1} \mathbf{F}_1 & \mathbf{Q}_1^{-1} \end{bmatrix}}_{\Sigma_1^{(\text{SC})}} + \underbrace{\mathbf{H_x}^\top \mathbf{R}^{-1} \mathbf{H_x} - \mathbf{H_x}^\top \mathbf{R}^{-1} \mathbf{H_f} (\mathbf{H_f}^\top \mathbf{R}^{-1} \mathbf{H_f})^{-1} \mathbf{H_f}^\top \mathbf{R}^{-1} \mathbf{H_x}}_{\Sigma_2^{(\text{SC})}} \end{aligned} \tag{23}$$

where $\Sigma_1^{(\text{SC})}$ and $\Sigma_2^{(\text{SC})}$ denote the information from the motion model (9) and measurement model (10), respectively.

3.3 Equivalence of Two Marginalizations

Based on the above, we present our main result:

Lemma 3.1. *Under the i.i.d. white Gaussian noise assumption, given the same linearization points, the information matrix* (23) *after Schur complement operation is equivalent to the information matrix* (16) *after null-space marginalization.*

Proof. See Appendix A. □

Note that since we have separated the information from the motion model (9) and measurement model (10), this analysis can be easily extended to multiple poses and features (see Appendix B). We also want to point out the following:

(i) The assumption that the noise is i.i.d. white Gaussian noise is widely used in most SLAM systems (e.g., [7, 9, 10]). If there is a difference between the null-space marginalization and Schur complement, it is due to different linearization points used.

(ii) The null-space marginalization for features can be applied to pre-integration based VINS [3, 17]. We can provide a simplified expression for the information matrix $\Sigma_{\mathbf{NS}}$ for robot poses after marginalization:

$$\Sigma_{\mathrm{NS}} = \underbrace{\begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_1\mathbf{F}_1^\top \\ \mathbf{F}_1\mathbf{P}_1 & \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1^\top + \mathbf{Q}_1 \end{bmatrix}^{-1}}_{\Sigma_1^{(\mathrm{NS})}} + \underbrace{\frac{1}{\sigma^2}(\mathbf{U}_n^\top\mathbf{H}_\mathbf{x})^\top\mathbf{U}_n^\top\mathbf{H}_\mathbf{x}}_{\Sigma_2^{(\mathrm{NS})}} \tag{24}$$

where $\Sigma_1^{(\mathrm{NS})}$ represents the information from the inertial measurements by pre-integration, while $\Sigma_2^{(\mathrm{NS})}$ represents the information from the visual measurements after null-space feature marginalization. Note that for $\Sigma_2^{(\mathrm{NS})}$, we do not need to calculate the null space $\mathbf{U_n}$ explicitly. $\mathbf{U_n^\top H_x}$ can be directly computed by Givens rotations.

(iii) With (42) in the proof, we find that $\mathbf{I} - \mathbf{U_e}\mathbf{U_e^\top}$ is also the null space of the feature Jacobian $\mathbf{H_f}$. Inspired by the this finding, we propose an algorithm for constructing analytical null space expression in Section 5.

3.4 Computational Complexity

We further compare the computational costs for Schur component and null-space marginalization. From the proof we know that Σ_1^{SC} and Σ_1^{NS} undertake the same operations, we only need to compare the computation of Σ_2^{SC} and Σ_2^{NS}. We assume there are n features that are all tracked by the m robot poses. Given the i.i.d. white Gaussian noise assumption, the computational complexity are outlined in Table 1 and 2.

It is not difficult to see that the computation for null-space operation is much easier. The dominant step for Schur complement operation is Step SC3 $O(mn^3)$, while the dominant step for null-space operation is Step NS2 $O(m^3n)$. Since features number n is much larger than that of the robot poses m, Step SC3 is more costly than Step NS2. Therefore, the null-space operation is more efficient. In Step NS1, $O(mn)$ represents the computational complexity with Givens rotations. Note that the complexity in Table 1 represents the most general case, but if taking into account the sparse structure of information matrix and using techniques such as column re-ordering in [18], the computational cost of the Schur complement can be greatly reduced.

4 Pose-and-Feature Marginalization

We discover that the null-space marginalization can also be applied to pose and feature marginalization. To see this, we consider the same SLAM system (1) and (2) as before. We want to marginalize the the pose $\mathbf{x}_2$ and the feature $\mathbf{x_f}$. The stacked linearized system can be written as:

$$\underbrace{\begin{bmatrix} \mathbf{0} \\ \widetilde{\mathbf{z}}_1 \\ \widetilde{\mathbf{z}}_2 \end{bmatrix}}_{\widetilde{\mathbf{z}}} \simeq \underbrace{\begin{bmatrix} \mathbf{F}_1 \\ \mathbf{H}_1 \\ \mathbf{0} \end{bmatrix}}_{\mathbf{H_k}} \widetilde{\mathbf{x}}_1 + \underbrace{\begin{bmatrix} -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_{\mathrm{f}1} \\ \mathbf{H}_2 & \mathbf{H}_{\mathrm{f}2} \end{bmatrix}}_{\mathbf{H_m}} \begin{bmatrix} \widetilde{\mathbf{x}}_2 \\ \widetilde{\mathbf{x}}_\mathbf{f} \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{w}_1 \\ \mathbf{n}_1 \\ \mathbf{n}_2 \end{bmatrix}}_{\mathbf{n}} \tag{25}$$

Table 1: Computational Complexity For Schur Component Operation

Steps	Operation	Complexity
Step SC1	$\mathbf{H_x^\top H_x}$	$O(m^3 n)$
Step SC2	$\mathbf{H_x^\top H_f}$	$O(m^2 n^2)$
Step SC3	$\mathbf{H_f^\top H_f}$	$O(mn^3)$
Step SC4	$(\mathbf{H_f^\top H_f})^{-1}$	$O(n^3)$
Step SC5	$\mathbf{H_x^\top H_f}\ (\mathbf{H_f^\top H_f})^{-1}$	$O(mn^2)$
Step SC6	$\mathbf{H_x^\top H_f}\ (\mathbf{H_f^\top H_f})^{-1}\ \mathbf{H_f^\top H_x}$	$O(m^2 n)$
Step SC7	$\mathbf{H_x^\top H_x} - \mathbf{H_x^\top H_f}\ (\mathbf{H_f^\top H_f})^{-1}\ \mathbf{H_f^\top H_x}$	$O(m)$

Table 2: Computational Complexity For Null Space Operation

Steps	Operation	Complexity
Step NS1	$\mathbf{U_n^\top H_x}$	$O(mn)$
Step NS2	$(\mathbf{U_n^\top H_x})^\top\ \mathbf{U_n^\top H_x}$	$O(m^3 n)$

In order to simplify the derivation, we denote $\mathbf{x_k}$ the state that we are going to keep and $\mathbf{x_m}$ the state that are going to be marginalized, $\mathbf{H_k}$ and $\mathbf{H_m}$ are their related Jacobians. $\mathbf{n}$ is the stacked noise with covariances $\mathbf{R}$ as $\begin{bmatrix} \mathbf{Q}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R}_2 \end{bmatrix}$. If $\mathbf{H_m}$ is of full column rank, then we can also use the left null space of $\mathbf{U_m}$ to marginalize the state $\mathbf{x}_m$ with $\mathbf{U_m^\top H_m} = \mathbf{0}$. Thus we have:

$$\mathbf{U_m^\top \widetilde{z}} = \mathbf{U_m^\top H_k \widetilde{x}_k} + \mathbf{U_m^\top n} \tag{26}$$

where $\mathbf{U_m}$ can computed by QR factorization of $\mathbf{H_m}$ as:

$$\mathbf{H_m} = \begin{bmatrix} \mathbf{U_k} & \mathbf{U_m} \end{bmatrix} \begin{bmatrix} \mathbf{R}_{\Delta_k} \\ \mathbf{0} \end{bmatrix} \tag{27}$$

The information for (26) is described as:

$$\Sigma^{(\mathrm{NS})} = \mathbf{P}_1^{-1} + \mathbf{H_k^\top U_m} (\mathbf{U_m^\top R U_m})^{-1} \mathbf{U_m^\top H_k} \tag{28}$$

By Schur complement, we get the information for $\mathbf{x_k}$ as:

$$\Sigma^{(\mathrm{SC})} = \Sigma_{\mathbf{kk}} - \Sigma_{\mathbf{km}} \Sigma_{\mathbf{mm}}^{-1} \Sigma_{\mathbf{mk}} \tag{29}$$

where we have partitioned the information matrix Σ according to the $\mathbf{x_k}$ and $\mathbf{x_m}$ as:

$$\Sigma = \begin{bmatrix} \Sigma_{\mathbf{kk}} & \Sigma_{\mathbf{km}} \\ \Sigma_{\mathbf{mk}} & \Sigma_{\mathbf{mm}} \end{bmatrix} \tag{30}$$

Then we can arrive at:

$$\Sigma^{(\mathrm{SC})} = \mathbf{P}_1^{-1} + \mathbf{H_k^\top R^{-1} H_k} - \mathbf{H_k^\top R^{-1} H_m} (\mathbf{H_m^\top R^{-1} H_m})^{-1} \mathbf{H_m^\top R^{-1} H_k} \Rightarrow$$

$$\Sigma^{(\mathrm{SC})} = \mathbf{P}_1^{-1} + \mathbf{H}_\mathbf{k}^\top \left[\mathbf{R}^{-1} - \mathbf{R}^{-1}\mathbf{U}_\mathbf{k}(\mathbf{U}_\mathbf{k}^\top \mathbf{R}^{-1}\mathbf{U}_\mathbf{k})^{-1}\mathbf{U}_\mathbf{k}^\top \mathbf{R}^{-1}\right] \mathbf{H}_\mathbf{k} \tag{31}$$

From (41) and (42), we can use the same proof for Lemma 3.1 to prove that (28) and (31) are equivalent under the i.i.d. assumption for the noises. Therefore, with the same assumption, we can extend the null-space operation for both pose and feature marginalization given that $\mathbf{H_m}$ is of full column rank, which is often needed in reducing the cost of graph SLAM (e.g., [6]).

5 Analytical Null-Space

In this section, we derive an analytical expression for null space in the MSCKF. In particular, the linearized measurement model for feature $\mathbf{x}_{\mathbf{f}_j}$ at time step k of the MSCKF can be structured as follows:

$$\widetilde{\mathbf{z}}_k = \underbrace{\frac{\partial \mathbf{h}_k}{\partial {}^{C_k}\mathbf{x}_{\mathbf{f}_j}}}_{\mathbf{H}_{C_k}} \underbrace{\frac{\partial {}^{C_k}\mathbf{x}_{\mathbf{f}_j}}{\partial \mathbf{x}}}_{\mathbf{H}_{\mathbf{x}_k}} \tilde{\mathbf{x}} + \underbrace{\frac{\partial \mathbf{h}_k}{\partial {}^{C_k}\mathbf{x}_{\mathbf{f}_j}}}_{\mathbf{H}_{C_k}} \underbrace{\frac{\partial {}^{C_k}\mathbf{x}_{\mathbf{f}_j}}{\partial \mathbf{x}_{\mathbf{f}_j}}}_{\mathbf{H}_{k\mathbf{f}j}} \widetilde{\mathbf{x}}_{\mathbf{f}_j} + \mathbf{n}_k \tag{32}$$

where ${}^{C_k}\mathbf{x}_{\mathbf{f}_j} = {}^{C_k}_{G}\mathbf{R}(\mathbf{x}_{\mathbf{f}_j} - {}^{G}\mathbf{p}_{C_k})$, $\mathbf{H}_{k\mathbf{f}j} = \frac{\partial {}^{C_k}\mathbf{x}_{\mathbf{f}_j}}{\partial \mathbf{x}_{\mathbf{f}_j}} = {}^{C_k}_{G}\mathbf{R}$, ${}^{G}\mathbf{p}_{C_k}$ and ${}^{C_k}_{G}\mathbf{R}$ are position and orientation from the cloned state $\mathbf{x}_{C_k}$, $\mathbf{H}_{C_k}$ represents the Jacobians of the sensor measurements (i.e., image key points) regarding to the local feature position ${}^{C_k}\mathbf{x}_{\mathbf{f}_j}$. Hence, if the sensor(s) can get the relative position estimate ${}^{C_k}\hat{\mathbf{x}}_{\mathbf{f}_j}$ of feature $\mathbf{x}_{\mathbf{f}_j}$ with only one measurement, $\mathbf{H}_{C_k}$ is of full column rank (e.g., stereo camera, RGBD camera). Thus, we can assume that $\mathbf{H}_{C_k}^{-1}$ is the inverse (or pseudo inverse) of $\mathbf{H}_{C_k}$, then multiply by (32) we get:

$$\mathbf{H}_{C_k}^{-1}\widetilde{\mathbf{z}}_k = \mathbf{H}_{\mathbf{x}_k}\widetilde{\mathbf{x}} + \mathbf{H}_{k\mathbf{f}j}\widetilde{\mathbf{x}}_{\mathbf{f}_j} + \mathbf{H}_{C_k}^{-1}\mathbf{n}_k \tag{33}$$

Assuming that we have m measurements for feature $\mathbf{x}_{\mathbf{f}_j}$, the system can be linearized regarding to the cloned states and features as:

$$\underbrace{\begin{bmatrix} \mathbf{H}_{C_1}^{-1}\widetilde{\mathbf{z}}_1 \\ \mathbf{H}_{C_2}^{-1}\widetilde{\mathbf{z}}_2 \\ \vdots \\ \mathbf{H}_{C_m}^{-1}\widetilde{\mathbf{z}}_m \end{bmatrix}}_{\widetilde{\mathbf{z}}} = \underbrace{\begin{bmatrix} \mathbf{H}_{\mathbf{x}_1} \\ \mathbf{H}_{\mathbf{x}_2} \\ \vdots \\ \mathbf{H}_{\mathbf{x}_m} \end{bmatrix}}_{\mathbf{H}_\mathbf{x}} \widetilde{\mathbf{x}} + \underbrace{\begin{bmatrix} {}^{C_1}_{G}\mathbf{R} \\ {}^{C_2}_{G}\mathbf{R} \\ \vdots \\ {}^{C_m}_{G}\mathbf{R} \end{bmatrix}}_{\mathbf{H}_{\mathbf{f}j}} \widetilde{\mathbf{x}}_{\mathbf{f}_j} + \underbrace{\begin{bmatrix} \mathbf{H}_{C_1}^{-1}\widetilde{\mathbf{n}}_1 \\ \mathbf{H}_{C_2}^{-1}\widetilde{\mathbf{n}}_2 \\ \vdots \\ \mathbf{H}_{C_m}^{-1}\widetilde{\mathbf{n}}_m \end{bmatrix}}_{\widetilde{\mathbf{n}}} \tag{34}$$

We now can find two analytical solutions for the left null space $\mathbf{U}$ of $\mathbf{H}_{\mathbf{f}j}$ such that $\mathbf{U}^\top\mathbf{H}_{\mathbf{f}j} = \mathbf{0}$.

(i) One analytical left null space solution $\mathbf{U}_{3m\times 3m}$ is:

$$\mathbf{U} = \mathbf{I}_{3m\times 3m} - \frac{\mathbf{H}_{\mathbf{f}j}\mathbf{H}_{\mathbf{f}j}^\top}{m} \tag{35}$$

Note that $m\mathbf{I}_3 = \mathbf{H}_{\mathbf{f}j}^\top\mathbf{H}_{\mathbf{f}j} = \sum_{k=1}^{m} {}^{C_k}_{G}\mathbf{R}^\top {}^{C_k}_{G}\mathbf{R}$.

(ii) Another analytical left null space $\mathbf{U}_{3(m-1)\times 3m}$ is:

$$\mathbf{U}^\top = \begin{bmatrix} -{}^{C_1}_{G}\mathbf{R}^\top & {}^{C_2}_{G}\mathbf{R}^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -{}^{C_1}_{G}\mathbf{R}^\top & \mathbf{0} & {}^{C_3}_{G}\mathbf{R}^\top & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ -{}^{C_1}_{G}\mathbf{R}^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} & {}^{C_m}_{G}\mathbf{R}^\top \end{bmatrix} \tag{36}$$

Note that many other null spaces can be formulated. Compared with (35), (36) has fewer elements and a lower dimension, thus less computation cost. Thus, it is used in our experiments.

6 Experimental Results

In this section, we present three sets of experiments to validate our analysis, in particular, Lemma 3.1 and the analytical null-space. Specifically, we evaluate with the MSCKF on real data to show that the null-space marginalization and Schur complement produce the same estimation accuracy. We then examine a factor graph-based visual odometry (VO) implementation to show that the information retained after both operations is identical. Lastly, we compare the analytical and numerical null-space operations implemented in the MSCKF evaluated in Monte-Carlo simulations.

6.1 MSCKF Case

We use the EuRoC data sets [19] to experimentally validate the equivalence of the null space and Schur compliment operations. The datasets were created by flying a AscTex Firefly MAV equipped with two global shutter greyscale cameras and an ADIS16448 IMU through two separate indoor environments. Position and attitude ground truth measurements are provided with the datasets from a post processing solution aided by a VICON system or a Leica laser tracker depending on the environment. The IMU data was provided at 200Hz, and the image data at 20Hz. Camera and IMU calibrations were provided.

We implemented the conventional covariance form of the MSCKF as described in [7] which utilizes the null space operation. We also implemented the MSCKF in information form, so that we could use the Schur complement operation. To clarify, we do not eliminate the feature sensitivity from the measurement equation as in (4) and (5). Instead, after the propagation step we obtain the total information matrix:

$$\Sigma_k = \begin{bmatrix} \mathbf{P}_{k|k-1}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{H}_\mathbf{x}^\top \\ \mathbf{H}_\mathbf{f}^\top \end{bmatrix} \mathbf{R}^{-1} \begin{bmatrix} \mathbf{H}_\mathbf{x} & \mathbf{H}_\mathbf{f} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{\mathbf{xx}} & \mathbf{A}_{\mathbf{xf}} \\ \mathbf{A}_{\mathbf{fx}} & \mathbf{A}_{\mathbf{ff}} \end{bmatrix} \tag{37}$$

We then obtain the updated covariance matrix and state correction from:

$$\mathbf{P}_{k|k} = \left(\mathbf{A}_{\mathbf{xx}} - \mathbf{A}_{\mathbf{xf}} \mathbf{A}_{\mathbf{ff}}^{-1} \mathbf{A}_{\mathbf{fx}} \right)^{-1} \tag{38}$$

$$\widetilde{\mathbf{x}}_{k|k} = \begin{bmatrix} \mathbf{P}_{k|k} & -\mathbf{P}_{k|k} \mathbf{A}_{\mathbf{xf}} \mathbf{A}_{\mathbf{ff}}^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{H}_\mathbf{x}^\top \\ \mathbf{H}_\mathbf{f}^\top \end{bmatrix} \mathbf{R}^{-1} \widetilde{\mathbf{z}}_k \tag{39}$$

Both filters were tuned the same, and had the same sliding window size of 7. The filters were run side-by-side so that the information form filter could be forced to use the same features in the update step (although each filter performed its own triangulation). The output of the two filters was identical up to machine precision. Figure 1 shows the similarity in position root mean squared error (RMSE) and normalized estimation error squared (NEES) [20]. The former measures the estimation consistency while the latter evaluates the estimation accuracy.

6.2 Factor Graph Case

We also implemented a simple example of a typical vision factor (Fig. 2) to validate Lemma 3.1. As shown in the photo from the KITTI odometry data set [21], the car is moving from time step 1 to time step 2. In oder to fuse the visual information with other sensors (eg., Lidar, GPS), we need to get the estimate of the odometry information ${}_2^1\mathbf{x}$ and its covariance (or its information Σ). For this single vision factor, there is no motion model information. Therefore, we only need to compare the visual information $\Sigma_2^{(\mathrm{SC})}$ from (23) and $\Sigma_2^{(\mathrm{NS})}$ from (16). We choose 2 pairs of stereo images from this dataset, extracted and matched 121 pairs of stereo features. We then formulate the measurement model with respect to the relative transformation ${}_2^1\mathbf{x}$ and all the features. The information matrix after the Schur complement operation and null space operation are shown in the 2. Note that σ^2 represents the normalized image noise variance. The differences $\Sigma_{Diff} (= \Sigma_2^{\mathrm{SC}} - \Sigma_2^{\mathrm{NS}})$ have orders of magnitude less than 10^{-12}, which is identical to machine precision. The

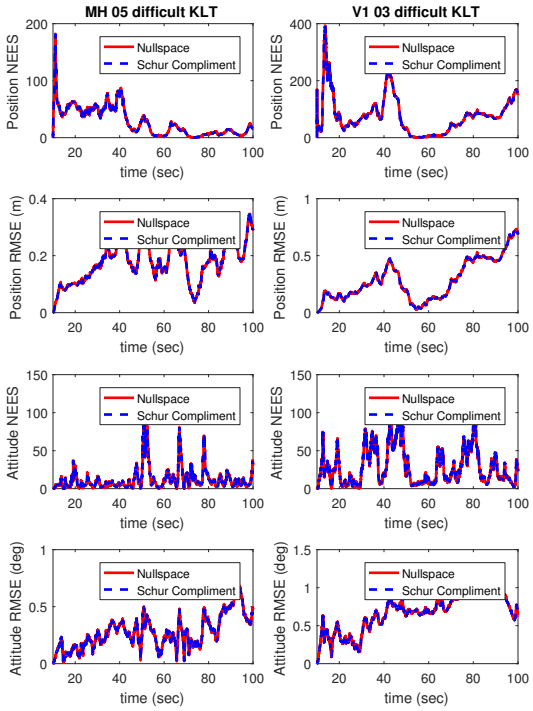

Figure 1: Comparison between nullspace and Schur compliment methods for feature marginalization in two EuRoC dataset sequences.

differences are shown below:

$$\Sigma_{Diff} = \frac{1}{\sigma^2}\begin{bmatrix} -5.04\mathbf{e}^{-14} & -1.25\mathbf{e}^{-13} & -1.05\mathbf{e}^{-13} & -4.44\mathbf{e}^{-15} & 3.64\mathbf{e}^{-14} & 3.05\mathbf{e}^{-15} \\ -5.15\mathbf{e}^{-14} & 3.06\mathbf{e}^{-12} & -1.60\mathbf{e}^{-14} & 9.41\mathbf{e}^{-14} & 2.73\mathbf{e}^{-15} & 1.17\mathbf{e}^{-13} \\ -9.26\mathbf{e}^{-14} & -6.39\mathbf{e}^{-14} & -3.09\mathbf{e}^{-13} & -3.77\mathbf{e}^{-15} & -8.33\mathbf{e}^{-17} & 2.18\mathbf{e}^{-15} \\ 6.63\mathbf{e}^{-15} & 1.51\mathbf{e}^{-13} & 2.22\mathbf{e}^{-15} & 5.11\mathbf{e}^{-15} & -3.94\mathbf{e}^{-16} & 2.69\mathbf{e}^{-15} \\ 3.91\mathbf{e}^{-14} & -1.66\mathbf{e}^{-16} & 3,33\mathbf{e}^{-16} & 8.71\mathbf{e}^{-17} & -3.22\mathbf{e}^{-15} & -2.84\mathbf{e}^{-16} \\ 1.17\mathbf{e}^{-14} & 1.15\mathbf{e}^{-13} & -1.90\mathbf{e}^{-15} & 5.22\mathbf{e}^{-15} & -4.99\mathbf{e}^{-16} & 2.96\mathbf{e}^{-15} \end{bmatrix}$$

6.3 Null-Space Comparison

In order to validate our proposed analytical null space (36), we run 50 Monte Carlo simulations of MSCKF with both numerical and analytical null spaces and compare the NEES and RMSE. In particular, Figure 3 shows the average NEES of 50 Monte-Carlo simulations for robot's position, attitude and orientation. It shows that numerical and analytical null space solutions work well for the estimate of attitude and velocity, and the results (average NEES) are almost the same. However,the MSCKF with analytical null space can generate more consistent position estimates. Similarly in Figure 4, both null space solutions have similar accuracy for attitude estimate, while the analytical solution generates a more accurate position estimate. The better performance of the proposed analytical null space is probably because the matrix has a sparse structure and utilizes the orthogonality of the state rotation matrix. Therefore, applying this analytical null space is similar to adding an implicit motion constraint to the estimator.

7 Conclusions and Future Work

We have analytically shown that the null-space marginalization and Schur complement preserve the same information about the remaining states under certain mild assumption (i.i.d. Gaussian noise model and same linearization points). This validates the application of null space marginalization to graph SLAM for

Figure 2: A typical vision factor for visual odometry. $\frac{1}{2}\mathbf{x}({}^{2}_{1}\mathbf{R}, {}^{1}\mathbf{p}_2)$ represents the transformation from stereo camera pose 1 to pose 2, where ${}^{2}_{1}\mathbf{R}$ and ${}^{1}\mathbf{p}_2$ denote the rotation and translation between pose 1 and pose 2,respectively.

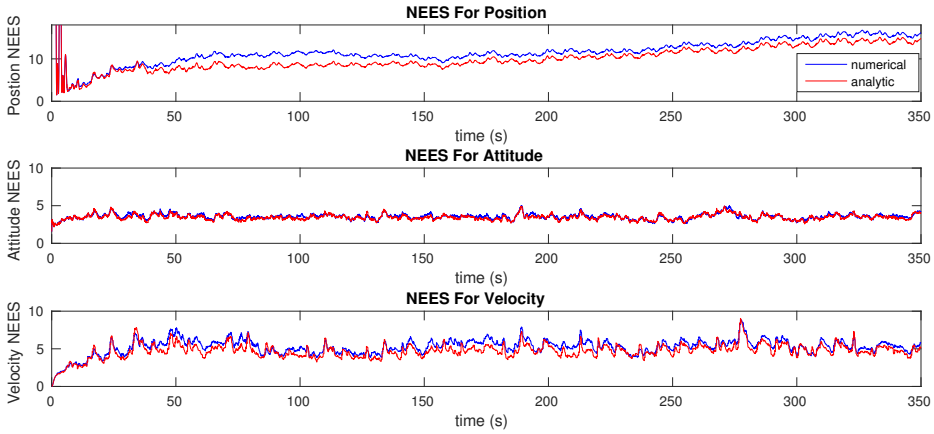

Figure 3: Average NEES of 50 Monte-Carlo Simulations for Robot's Position, Attitude and Velocity. The blue and red lines represent the results for the MSCKF with numerical null space and analytical null space, respectively.

potential efficiency gain. Moreover, we have offered the analytical null space expression for commonly used sensors such as stereo and RGBD cameras, which has been shown to have better performance than the numerically computed one. As for the future work, we plan to apply the null-space marginalization to a broader family of estimation problems that require marginalization to gain better efficiency.

A Proof of Lemma 3.1

To show $\Sigma_1^{(\mathrm{NS})} = \Sigma_1^{(\mathrm{SC})}$, matrix inversion lemma yields:

$$\Sigma_1^{(\mathrm{NS})} = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_1\mathbf{F}_1^\top \\ \mathbf{F}_1\mathbf{P}_1 & \mathbf{F}_1\mathbf{P}_1\mathbf{F}_1^\top + \mathbf{Q}_1 \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{P}_1^{-1} + \mathbf{F}_1^\top\mathbf{Q}_1^{-1}\mathbf{F}_1 & -\mathbf{F}_1^\top\mathbf{Q}_1^{-1} \\ -\mathbf{Q}_1^{-1}\mathbf{F}_1 & \mathbf{Q}_1^{-1} \end{bmatrix} = \Sigma_1^{(\mathrm{SC})} \tag{40}$$

To show $\Sigma_2^{(\mathrm{NS})} = \Sigma_2^{(\mathrm{SC})}$, we consider the following cases:

A.0.1 Case I

We start with $\mathbf{R}_k = \sigma^2\mathbf{I}_{\mathbf{R}_k}$, where σ^2 is the noise variance and $\mathbf{I}_{\mathbf{R}_k}$ is an identity matrix with size of $\mathbf{R}_k$. Under the assumption of i.i.d. Gaussian noise, we can easily get the stacked noise covariances as $\mathbf{R} = \sigma^2\mathbf{I}$.

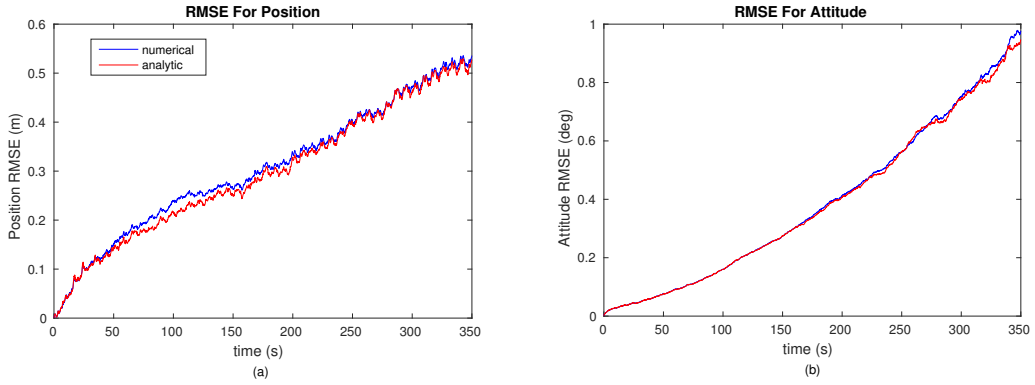

Figure 4: Average RMSE of 50 Monte-Carlo Simulations for Robot's Position and Attitude. The blue and red lines represents the results for the MSCKF with numerical null space and analytical null space, respectively.

Then, from (15) (16) and (23), we have:

$$\begin{aligned}\Sigma_2^{(\mathrm{SC})} &= \mathbf{H}_\mathbf{x}^\top \left[\mathbf{R}^{-1} - \mathbf{R}^{-1}\mathbf{U}_\mathbf{e}(\mathbf{U}_\mathbf{e}^\top\mathbf{R}^{-1}\mathbf{U}_\mathbf{e})^{-1}\mathbf{U}_\mathbf{e}^\top\mathbf{R}^{-1}\right]\mathbf{H}_\mathbf{x} \\ &= \mathbf{H}_\mathbf{x}^\top \left[\frac{1}{\sigma^2}(\mathbf{I} - \mathbf{U}_\mathbf{e}\mathbf{U}_\mathbf{e}^\top)\right]\mathbf{H}_\mathbf{x} \qquad (41)\\ \Sigma_2^{(\mathrm{NS})} &= \mathbf{H}_\mathbf{x}^\top \left[\mathbf{U}_\mathbf{n}(\mathbf{U}_\mathbf{n}^\top\mathbf{R}\mathbf{U}_\mathbf{n})^{-1}\mathbf{U}_\mathbf{n}^\top\right]\mathbf{H}_\mathbf{x} \\ &= \mathbf{H}_\mathbf{x}^\top \left[\frac{1}{\sigma^2}\mathbf{U}_\mathbf{n}\mathbf{U}_\mathbf{n}^\top\right]\mathbf{H}_\mathbf{x} \qquad (42)\end{aligned}$$

From QR, we know $\mathbf{U}_\mathbf{e}\mathbf{U}_\mathbf{e}^\top + \mathbf{U}_\mathbf{n}\mathbf{U}_\mathbf{n}^\top = \mathbf{I}$. Thus, upon the same linearization point, we have $\Sigma_2^{(\mathrm{SC})} = \Sigma_2^{(\mathrm{NS})}$. Therefore, we conclude that (23) and (16) are equivalent, that is: information from NS operation equals to that with SC operation.

A.0.2 Case II

If the noise covariances matrix $\mathbf{R}_k$ are full matrix, we can perform pre-whitening for the measurement equations before the null space operation. Since $\mathbf{R}_k$ is symmetrical, positive and definite, we can factorized $\mathbf{R}_k$ as:

$$\mathbf{R}_k = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^\top = (\mathbf{V}\mathbf{\Lambda}^{\frac{1}{2}})(\mathbf{V}\mathbf{\Lambda}^{\frac{1}{2}})^\top = \widehat{\mathbf{V}}\widehat{\mathbf{V}}^\top \qquad (43)$$

$$\Rightarrow \widehat{\mathbf{V}}^{-1}\mathbf{R}_k(\widehat{\mathbf{V}}^\top)^{-1} = \mathbf{I}_\Lambda \qquad (44)$$

where $\mathbf{\Lambda}$ is a diagonal matrix and $\widehat{\mathbf{V}} = \mathbf{V}\mathbf{\Lambda}^{\frac{1}{2}}$. Left multiplication of $\widehat{\mathbf{V}}^{-1}$ to the linearized measurement equation yields:

$$\underbrace{\widehat{\mathbf{V}}^{-1}\widetilde{\mathbf{z}}_k}_{\check{\mathbf{z}}_k} = \underbrace{\widehat{\mathbf{V}}^{-1}\mathbf{H}_k}_{\check{\mathbf{H}}_k}\widetilde{\mathbf{x}}_k + \underbrace{\widehat{\mathbf{V}}^{-1}\mathbf{H}_{\mathbf{f}k}}_{\check{\mathbf{H}}_{\mathbf{f}k}}\widetilde{\mathbf{x}}_\mathbf{f} + \underbrace{\widehat{\mathbf{V}}^{-1}\mathbf{n}_k}_{\check{\mathbf{n}}_k} \qquad (45)$$

After pre-whitening, the new measurement noise becomes $\check{\mathbf{n}}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_\Lambda)$, where $\mathbf{I}_\Lambda$ is identity matrix. Then, we can follow the same steps in case I with new Jacobians in (45).

B Marginalization of Multi-Poses and Features

We first solve the information matrix with null space operation Σ_{NS} according to (16):

$$\Sigma_{\mathrm{NS}} = \Sigma_1^{(\mathrm{NS})} + \Sigma_2^{(\mathrm{NS})} \qquad (46)$$

where $\Sigma_1^{(\mathrm{NS})}$ and $\Sigma_2^{(\mathrm{NS})}$ denote the information from multiple poses and features, respectively. We know that the inverse of state covariances after K steps of propagation $(\mathbf{P}_\mathbf{x})^{-1} = \Sigma_1^{(\mathrm{NS})}$, and $\mathbf{P}_\mathbf{x}$ is:

$$\mathbf{P}_\mathbf{x} = \begin{bmatrix} \mathbf{P}_1 & \mathbf{P}_1\mathbf{F}_1^\top & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{F}_1\mathbf{P}_1 & \mathbf{P}_{2|1} & \mathbf{P}_{2|1}\mathbf{F}_2^\top & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{F}_2\mathbf{P}_{2|1} & \mathbf{P}_{3|2} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \ddots & \mathbf{P}_{K-1|K-2}\mathbf{F}_{K-1}^\top \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{F}_{K-1}\mathbf{P}_{K-1|K-2} & \mathbf{P}_{K|K-1} \end{bmatrix} \tag{47}$$

where $\mathbf{P}_{k+1|k} = \mathbf{F}_k\mathbf{P}_{k|k-1}\mathbf{F}_k^\top + \mathbf{Q}_k$, where $k = 2 \dots K-1$. The information $\Sigma_1^{(\mathrm{NS})}$ from features $j = 1 \dots N$ can be expressed as follows:

$$\Sigma_2^{(\mathrm{NS})} = \sum_{j=1}^{N} \Sigma_2^{(\mathrm{NS}j)} = \sum_{j=1}^{N} \mathbf{H}_\mathbf{x}^{(j)\top}\mathbf{U}_\mathbf{n}^{(j)}(\mathbf{U}_\mathbf{n}^{(j)\top}\mathbf{R}^{(j)}\mathbf{U}_\mathbf{n}^{(j)})^{-1}\mathbf{U}_\mathbf{n}^{(j)\top}\mathbf{H}_\mathbf{x}^{(j)} \tag{48}$$

where $(\cdot)^{(j)}$ represents the corresponding parameters regarding to the feature $\mathbf{x}_{\mathbf{f}_j}$. Similarly, we can solve the information matrix in MLE formulation with Schur complement operation Σ_{SC} as follows:

$$\Sigma_{\mathrm{SC}} = \Sigma_1^{(\mathrm{SC})} + \Sigma_2^{(\mathrm{SC})} \tag{49}$$

where $\Sigma_1^{(\mathrm{SC})}$ and $\Sigma_2^{(\mathrm{SC})}$ denote the information from poses and features after Schur complement operation, respectively.

$$\Sigma_1^{(\mathrm{SC})} = \begin{bmatrix} \mathbf{P}_1^{-1} + \mathbf{F}_1^\top\mathbf{Q}_1^{-1}\mathbf{F}_1 & -\mathbf{F}_1^\top\mathbf{Q}_1^{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{Q}_1^{-1}\mathbf{F}_1 & \mathbf{Q}_1^{-1} + \mathbf{F}_2^\top\mathbf{Q}_2^{-1}\mathbf{F}_2 & -\mathbf{F}_2^\top\mathbf{Q}_2^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\mathbf{Q}_2^{-1}\mathbf{F}_2 & \mathbf{Q}_2^{-1} + \mathbf{F}_3^\top\mathbf{Q}_3^{-1}\mathbf{F}_3 & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \ddots & -\mathbf{F}_{K-1}^\top\mathbf{Q}_{K-1}^{-1} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & -\mathbf{Q}_{K-1}^{-1}\mathbf{F}_{K-1} & \mathbf{Q}_{K-1}^{-1} \end{bmatrix} \tag{50}$$

If we consider the feature Jacobians for the whole sytem as:

$$\mathbf{H}_\mathbf{f} = \begin{bmatrix} \mathbf{H}_\mathbf{f}^{(1)} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{H}_\mathbf{f}^{(2)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_\mathbf{f}^{(N)} \end{bmatrix} = \begin{bmatrix} \mathbf{U}_e^{(1)} & \mathbf{U}_e^{(2)} & \cdots & \mathbf{U}_e^{(N)} \end{bmatrix} \begin{bmatrix} \mathbf{R}_\Delta^{(1)} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_\Delta^{(2)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{R}_\Delta^{(N)} \end{bmatrix} \tag{51}$$

The matrix $\Sigma_2^{(\mathrm{SC})}$ can be written as:

$$\Sigma_2^{(\mathrm{CS})} = \sum_{j=1}^{N} \Sigma_2^{(\mathrm{CS}j)} = \sum_{j=1}^{N} \left[\mathbf{H}_\mathbf{x}^{(j)\top} \left[(\mathbf{R}^{(j)})^{-1} - (\mathbf{R}^{(j)})^{-1}\mathbf{U}_\mathbf{e}^{(j)}[\mathbf{U}_\mathbf{e}^{(j)\top}(\mathbf{R}^{(j)})^{-1}\mathbf{U}_\mathbf{e}^{(j)}]^{-1}\mathbf{U}_\mathbf{e}^{(j)\top}(\mathbf{R}^{(j)})^{-1} \right] \mathbf{H}_\mathbf{x}^{(j)} \right] \tag{52}$$

From the proof for a single feature $\mathbf{x}_{\mathbf{f}_j}$, we can have $\Sigma_2^{(\mathrm{NS}j)} = \Sigma_2^{(\mathrm{SC}j)}$. (48) and (52) are just the summation of each feature's information. Therefore, they are also equivalent.

At last, $\Sigma_1^{(\mathrm{NS})} = \mathbf{P}_\mathbf{x}^{-1} = \Sigma_1^{(\mathrm{SC})}$ can be proved by induction. Let $\mathbf{P}_{\mathbf{x}_k}$ denotes the propagated covariance at time step k, then we have:

$$\mathbf{P}_{\mathbf{x}_{k+1}} = \begin{bmatrix} \mathbf{P}_{\mathbf{x}_k} & \mathbf{P}_{\mathbf{x}_k}\mathbf{H}_{\mathbf{F}_k}^\top \\ \mathbf{H}_{\mathbf{F}_k}\mathbf{P}_{\mathbf{x}_k} & \mathbf{P}_{k+1|k} \end{bmatrix} \tag{53}$$

where $\mathbf{H}_{\mathbf{F}_k} = \begin{bmatrix} \mathbf{0} & \cdots & \mathbf{0} & \mathbf{F}_k \end{bmatrix}$. We also define $\Sigma_{1_k}^{(\mathrm{NS})}$ and $\Sigma_{1_k}^{(\mathrm{SC})}$, which denote the propagated information and MLE information, respectively. For the induction, when k=1, we have:

$$\Sigma_{1_1}^{(\mathrm{NS})} = \mathbf{P}_{\mathbf{x}_1}^{-1} = \mathbf{P}_1^{-1} = \Sigma_{1_1}^{(\mathrm{SC})} \tag{54}$$

Now, if we have $\Sigma_{1_k}^{(\mathrm{NS})} = \Sigma_{1_k}^{(\mathrm{SC})}$, we need to show that: $\Sigma_{1_{k+1}}^{(\mathrm{NS})} = \Sigma_{1_{k+1}}^{(\mathrm{SC})}$. From matrix inversion Lemma, we have:

$$\Sigma_{1_{k+1}}^{(\mathrm{NS})} = \mathbf{P}_{\mathbf{x}_{k+1}}^{-1} = \begin{bmatrix} \mathbf{P}_{\mathbf{x}_k} & \mathbf{P}_{\mathbf{x}_k}\mathbf{H}_{\mathbf{F}_k}^\top \\ \mathbf{H}_{\mathbf{F}_k}\mathbf{P}_{\mathbf{x}_k} & \mathbf{H}_{\mathbf{F}_k}\mathbf{P}_{\mathbf{x}_k}\mathbf{H}_{\mathbf{F}_k}^\top + \mathbf{Q}_k \end{bmatrix}^{-1} = \begin{bmatrix} \mathbf{P}_{\mathbf{x}_k}^{-1} + \mathbf{H}_{\mathbf{F}_k}^\top \mathbf{Q}_k^{-1}\mathbf{H}_{\mathbf{F}_k} & -\mathbf{H}_{\mathbf{F}_k}^\top \mathbf{Q}_k^{-1} \\ -\mathbf{Q}_k^{-1}\mathbf{H}_{\mathbf{F}_k} & \mathbf{Q}_k^{-1} \end{bmatrix} = \Sigma_{1_{k+1}}^{(\mathrm{SC})} \tag{55}$$

C Running Time Comparison

This section is a detail explanation for the vision factor experiment. From the Description in the paper, we can have the following measurement equations ($\mathbf{n}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$):

$$\mathbf{z}_1 = \mathbf{h}({}^1\mathbf{x}_\mathbf{f}) + \mathbf{n}_1 \tag{56}$$

$$\mathbf{z}_2 = \mathbf{h}({}_2^1\mathbf{x}, {}^1\mathbf{x}_\mathbf{f}) + \mathbf{n}_2 \tag{57}$$

where

$$\mathbf{z}_k = \begin{bmatrix} {}^k u_l \\ {}^k v_l \\ {}^k u_r \end{bmatrix} = \begin{bmatrix} \frac{{}^k x_\mathrm{f}}{{}^k z_\mathrm{f}} \\ \frac{{}^k y_\mathrm{f}}{{}^k z_\mathrm{f}} \\ \frac{{}^k x_\mathrm{f} - b}{{}^k z_\mathrm{f}} \end{bmatrix} \tag{58}$$

$$\mathbf{R}_k = \sigma^2 \mathbf{I}_2 \tag{59}$$

After linearization, we can have:

$$\begin{bmatrix} \tilde{\mathbf{z}}_1 \\ \tilde{\mathbf{z}}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{3\times 6} & \mathbf{H}_{\mathbf{proj}_1} \\ \mathbf{H}_{\mathbf{proj}_2}\mathbf{H}_{12} & \mathbf{H}_{\mathbf{proj}_2}\mathbf{H}_{2\mathbf{f}} \end{bmatrix} \begin{bmatrix} {}_2^1\tilde{\mathbf{x}} \\ {}^1\tilde{\mathbf{x}}_\mathbf{f} \end{bmatrix} + \begin{bmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \end{bmatrix} \tag{60}$$

where

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}_2^1\tilde{\mathbf{x}}} = \mathbf{0} \tag{61}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^1\tilde{\mathbf{x}}_\mathbf{f}} = \mathbf{H}_{\mathbf{proj}_1} \tag{62}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial {}_2^1\tilde{\mathbf{x}}} = \mathbf{H}_{\mathbf{proj}_1}\mathbf{H}_{12} = \mathbf{H}_{\mathbf{proj}_1}\begin{bmatrix} \lfloor {}_1^2\mathbf{R}({}^1\mathbf{P}_\mathbf{f} - {}^1\mathbf{P}_2)\times \rfloor & -{}_1^2\mathbf{R} \end{bmatrix} \tag{63}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^1\tilde{\mathbf{x}}_\mathbf{f}} = \mathbf{H}_{\mathbf{proj}_2}\mathbf{H}_{2\mathbf{f}} = \mathbf{H}_{\mathbf{proj}_2}{}_1^2\mathbf{R} \tag{64}$$

$$\mathbf{H}_{\mathbf{proj}_k} = \begin{bmatrix} \frac{1}{{}^k z_\mathrm{f}} & 0 & -\frac{{}^k x_\mathrm{f}}{{}^k z_\mathrm{f}^2} \\ 0 & \frac{1}{{}^k z_\mathrm{f}} & -\frac{{}^k y_\mathrm{f}}{{}^k z_\mathrm{f}^2} \\ \frac{1}{{}^k z_\mathrm{f}} & 0 & -\frac{{}^k x_\mathrm{f} - b}{{}^k z_\mathrm{f}^2} \end{bmatrix} \tag{65}$$

Then, we can rewrite the linearized equations as:

$$\underbrace{\begin{bmatrix} \tilde{\mathbf{z}}_1 \\ \tilde{\mathbf{z}}_2 \end{bmatrix}}_{\tilde{\mathbf{z}}} = \underbrace{\begin{bmatrix} \mathbf{0}_{3\times 6} \\ \mathbf{H}_{\mathbf{proj}_2}\mathbf{H}_{12} \end{bmatrix}}_{\mathbf{H}_\mathbf{x}} {}_2^1\tilde{\mathbf{x}} + \underbrace{\begin{bmatrix} \mathbf{H}_{\mathbf{proj}_1} \\ \mathbf{H}_{\mathbf{proj}_2}\mathbf{H}_{2\mathbf{f}} \end{bmatrix}}_{\mathbf{H}_\mathbf{f}} {}^1\tilde{\mathbf{x}}_\mathbf{f} + \underbrace{\begin{bmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \end{bmatrix}}_{\mathbf{n}} \tag{66}$$

Note that, for feature $j, j = 1 \dots m$, we can have:

$$\begin{aligned}
\tilde{\mathbf{z}}^{(1)} &= \mathbf{H}_{\mathbf{x}}^{(1)} {}_2^1\tilde{\mathbf{x}} + \mathbf{H}_{\mathbf{f}}^{(1)} {}^1\tilde{\mathbf{x}}_{\mathbf{f}}^{(1)} + \mathbf{n}^{(j)} && (67)\\
\vdots &= \vdots && (68)\\
\tilde{\mathbf{z}}^{(j)} &= \mathbf{H}_{\mathbf{x}}^{(j)} {}_2^1\tilde{\mathbf{x}} + \mathbf{H}_{\mathbf{f}}^{(j)} {}^1\tilde{\mathbf{x}}_{\mathbf{f}}^{(j)} + \mathbf{n}^{(j)} && (69)\\
\vdots &= \vdots && (70)\\
\tilde{\mathbf{z}}^{(m)} &= \mathbf{H}_{\mathbf{x}}^{(m)} {}_2^1\tilde{\mathbf{x}} + \mathbf{H}_{\mathbf{f}}^{(m)} {}^1\tilde{\mathbf{x}}_{\mathbf{f}}^{(m)} + \mathbf{n}^{(m)} && (71)
\end{aligned}$$

Now we will compare the computation running time for feature marginalization.

C.1 Graph SLAM Optimization

For the Marginalization, the first step is to compute the linearizion points (no matter MSCKF or graph SLAM), that is the state estimate ${}_2^1\hat{\mathbf{x}}$ and the $\hat{\mathbf{x}}_{\mathbf{f}}$, the formulation for this Graph SLAM problem is:

$$\sum_{j=1}^{m} \left\| \begin{bmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \end{bmatrix}^{(j)} - \begin{bmatrix} \mathbf{h}({}^1\mathbf{x}_{\mathbf{f}}) \\ \mathbf{h}({}_2^1\mathbf{x}, {}^1\mathbf{x}_{\mathbf{f}}) \end{bmatrix}^{(j)} \right\|^2_{\mathbf{R}_k^{-1}} \tag{72}$$

The Jacobians are shown in (66). After this graph optimization, we can get the linearization points and marginalize the features to keep only the pose information $\Sigma_{\mathbf{x}}$.

Algorithm 1 SLAM with Feature Marginalization

1: Input: $\mathbf{z}_1^{(j)}$,$\mathbf{z}_2^{(j)}$ and stereo baseline b with $j = 1 \dots m$
2: **for** $j \leftarrow 1$, m **do**
3: Compute $\mathbf{H}_{\mathbf{x}}^{(j)}$ and $\mathbf{H}_{\mathbf{f}}^{(j)}$
4: **end for**
5: Solve the ${}_2^1\hat{\mathbf{x}}$ and the $\hat{\mathbf{x}}_{\mathbf{f}}^{(j)} (j = 1 \dots m)$ from (72) with Levenberg Marquardt Method.
6: $\hat{\mathbf{x}}_{\mathbf{f}}^{(j)} (j = 1 \dots m)$ marginalization with one algorithm from Section C.2
7: Return the remaining pose information $\Sigma_{\mathbf{x}}$

C.2 Marginalization Algorithm

C.2.1 Schur Complement Marginalization

We need first construct the Measurement Jacobians:

$$\underbrace{\begin{bmatrix} \tilde{\mathbf{z}}^{(1)} \\ \vdots \\ \tilde{\mathbf{z}}^{(j)} \\ \vdots \\ \tilde{\mathbf{z}}^{(m)} \end{bmatrix}}_{\tilde{\mathbf{z}}^{(all)}} = \underbrace{\begin{bmatrix} \mathbf{H}_{\mathbf{x}}^{(1)} \\ \vdots \\ \mathbf{H}_{\mathbf{x}}^{(j)} \\ \vdots \\ \mathbf{H}_{\mathbf{x}}^{(m)} \end{bmatrix}}_{\mathbf{H}_{\mathbf{x}}^{(all)}} {}_2^1\tilde{\mathbf{x}} + \underbrace{\begin{bmatrix} \mathbf{H}_{\mathbf{f}}^{(1)} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{H}_{\mathbf{f}}^{(j)} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{H}_{\mathbf{f}}^{(m)} \end{bmatrix}}_{\mathbf{H}_{\mathbf{x}}^{(all)}} \begin{bmatrix} {}^1\tilde{\mathbf{x}}_{\mathbf{f}}^{(1)} \\ \vdots \\ {}^1\tilde{\mathbf{x}}_{\mathbf{f}}^{(j)} \\ \vdots \\ {}^1\tilde{\mathbf{x}}_{\mathbf{f}}^{(m)} \end{bmatrix} + \underbrace{\begin{bmatrix} \mathbf{n}^{(1)} \\ \vdots \\ \mathbf{n}^{(j)} \\ \vdots \\ \mathbf{n}^{(m)} \end{bmatrix}}_{\mathbf{n}^{(all)}} \tag{73}$$

And the info after marginalization for the ${}_2^1\mathbf{x}$ is:

$$\Sigma_{\mathbf{x}}^{SCF} = \mathbf{H}_{\mathbf{x}}^{(all)\top} \left(\mathbf{R}^{(all)}\right)^{-1} \mathbf{H}_{\mathbf{x}}^{(all)} - \mathbf{H}_{\mathbf{x}}^{(all)\top} \left(\mathbf{R}^{(all)}\right)^{-1} \mathbf{H}_{\mathbf{f}}^{(all)} \left[\mathbf{H}_{\mathbf{f}}^{(all)\top} \left(\mathbf{R}^{(all)}\right)^{-1} \mathbf{H}_{\mathbf{f}}^{(all)}\right]^{-1} \mathbf{H}_{\mathbf{f}}^{(all)\top} \left(\mathbf{R}^{(all)}\right)^{-1} \mathbf{H}_{\mathbf{x}}^{(all)} \tag{74}$$

Since $\mathbf{R}^{(all)} = \sigma^2\mathbf{I}$, we can simplify the above equation as:

$$\Sigma_{\mathbf{x}}^{SCF} = \frac{1}{\sigma^2}\left(\mathbf{H}_{\mathbf{x}}^{(all)\top}\mathbf{H}_{\mathbf{x}}^{(all)} - \mathbf{H}_{\mathbf{x}}^{(all)\top}\mathbf{H}_{\mathbf{f}}^{(all)}\left[\mathbf{H}_{\mathbf{f}}^{(all)\top}\mathbf{H}_{\mathbf{f}}^{(all)}\right]^{-1}\mathbf{H}_{\mathbf{f}}^{(all)\top}\mathbf{H}_{\mathbf{x}}^{(all)}\right) \tag{75}$$

See the Algorithm 2.

Algorithm 2 Schur Complement Marginalization

1: Input: ${}_2^1\hat{\mathbf{x}}$, ${}^1\mathbf{x}_{\mathbf{f}}^{(j)}$, $\mathbf{n}^{(j)}$, $j = 1 \dots m$
2: Construct $\mathbf{H}_{\mathbf{x}}^{(all)}$ and $\mathbf{H}_{\mathbf{f}}^{(all)}$
3: Compute $\mathbf{H}_{\mathbf{x}}^{(all)\top}\mathbf{H}_{\mathbf{x}}^{(all)}$
4: Compute $\mathbf{H}_{\mathbf{x}}^{(all)\top}\mathbf{H}_{\mathbf{f}}^{(all)}$
5: Compute $\left[\mathbf{H}_{\mathbf{f}}^{(all)\top}\mathbf{H}_{\mathbf{f}}^{(all)}\right]^{-1}$
6: Compute $\Sigma_{\mathbf{x}}^{SCF}$ by Eq. (75)

C.2.2 Schur Complement Marginalization With Sparse Structure

With a careful inspection of the (73), we find that the feature Jacobians are with perfect sparse structure, therefore, we can consider to take advantage of this and design more efficient algorithm. For (75), we can have:

$$\left[\mathbf{H}_{\mathbf{f}}^{(all)\top}\mathbf{H}_{\mathbf{f}}^{(all)}\right]^{-1} = \begin{bmatrix} \left(\mathbf{H}_{\mathbf{f}}^{(1)^\top}\mathbf{H}_{\mathbf{f}}^{(1)}\right)^{-1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \ddots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \left(\mathbf{H}_{\mathbf{f}}^{(j)^\top}\mathbf{H}_{\mathbf{f}}^{(j)}\right)^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \left(\mathbf{H}_{\mathbf{f}}^{(m)^\top}\mathbf{H}_{\mathbf{f}}^{(m)}\right)^{-1} \end{bmatrix} \tag{76}$$

$$\mathbf{H}_{\mathbf{x}}^{(all)\top}\mathbf{H}_{\mathbf{f}}^{(all)} = \begin{bmatrix} \mathbf{H}_{\mathbf{x}}^{(1)\top}\mathbf{H}_{\mathbf{f}}^{(1)} & \cdots & \mathbf{H}_{\mathbf{x}}^{(j)\top}\mathbf{H}_{\mathbf{f}}^{(j)} & \cdots & \mathbf{H}_{\mathbf{x}}^{(m)\top}\mathbf{H}_{\mathbf{f}}^{(m)} \end{bmatrix} \tag{77}$$

Therefore, we can rewrite (75) as:

$$\Sigma_{\mathbf{x}}^{SCS} = \Sigma_{\mathbf{x}}^{SCF} \tag{78}$$

$$= \frac{1}{\sigma^2}\sum_{j=1}^{m}\left(\mathbf{H}_{\mathbf{x}}^{(j)\top}\mathbf{H}_{\mathbf{x}}^{(j)} - \mathbf{H}_{\mathbf{x}}^{(j)\top}\mathbf{H}_{\mathbf{f}}^{(j)}\left[\mathbf{H}_{\mathbf{f}}^{(j)\top}\mathbf{H}_{\mathbf{f}}^{(j)}\right]^{-1}\mathbf{H}_{\mathbf{f}}^{(j)\top}\mathbf{H}_{\mathbf{x}}^{(j)}\right) \tag{79}$$

$$= \frac{1}{\sigma^2}\sum_{j=1}^{m}\left(\Sigma_{\mathbf{x}^{(j)}}^{SC}\right) \tag{80}$$

From the above derivation, we can see that the Schur complement operation is equivalent to the summation of results from individual schur complement for each single feature. We can design the Schur complement algorithm with sparse structure as:

C.2.3 Null Space Marginalization With Householder QR

For null space marginalization, the key is to compute the left null space of the feature Jacobian $\mathbf{H}_{\mathbf{f}}$. One method is to directly compute the QR factorization of $\mathbf{H}_{\mathbf{f}}$ as:

$$\mathbf{H}_{\mathbf{f}} = \begin{bmatrix}\mathbf{U}_{\mathbf{e}} & \mathbf{U}_{\mathbf{n}}\end{bmatrix}\begin{bmatrix}\mathbf{R}_{\Delta} \\ \mathbf{0}\end{bmatrix} \tag{81}$$

Algorithm 3 Schur Complement Marginalization With Sparse Structure

1: Input: ${}_2^1\hat{\mathbf{x}}$, ${}^1\mathbf{x}_\mathbf{f}^{(j)}, \mathbf{n}^{(j)}, j = 1 \dots m$
2: **for** $j \leftarrow 1,\ m$ **do**
3: Compute $\mathbf{H}_\mathbf{x}^{(j)}$ and $\mathbf{H}_\mathbf{f}^{(j)}$
4: Compute $\Sigma_{\mathbf{x}^{(j)}}^{SC}$
5: **end for**
6: Compute $\Sigma_\mathbf{x}^{SCS}$ by Eq. (80)

where $\mathbf{U_n}$ is the null space. Then, the resulting information can be described as:

$$\Sigma_\mathbf{x}^{(NSqr)} = \sum_{j=1}^{m} \left(\left(\mathbf{U}_\mathbf{n}^{(j)\top} \mathbf{H}_\mathbf{x}^{(j)} \right)^\top \mathbf{U}_\mathbf{n}^{(j)\top} \mathbf{H}_\mathbf{x}^{(j)} \right) = \sum_{j=1}^{m} \left(\Sigma_{\mathbf{x}^{(j)}}^{(NSqr)} \right) \tag{82}$$

Therefore, the algorithm can be described as:

Algorithm 4 Null-Space Marginalization With Householder QR

1: Input: ${}_2^1\hat{\mathbf{x}}$, ${}^1\mathbf{x}_\mathbf{f}^{(j)}, \mathbf{n}^{(j)}, j = 1 \dots m$
2: **for** $j \leftarrow 1,\ m$ **do**
3: Compute $\mathbf{H}_\mathbf{x}^{(j)}$ and $\mathbf{H}_\mathbf{f}^{(j)}$
4: Compute QR factorization with Householder (Eigen function) and get $\mathbf{U}_\mathbf{n}^{(j)}$
5: Compute $\mathbf{U}_\mathbf{n}^{(j)\top} \mathbf{H}_\mathbf{x}^{(j)}$
6: Compute $\Sigma_{\mathbf{x}^{(j)}}^{(NSqr)}$
7: **end for**
8: Compute $\Sigma_\mathbf{x}^{NSqr}$ by Eq. (82)

C.2.4 Null Space Marginalization With Projection

An explicit null space expression can be described in the form as:

$$\mathbf{U_p} = \mathbf{I} - \mathbf{H_f} \left(\mathbf{H}_\mathbf{f}^\top \mathbf{H_f} \right)^{-1} \mathbf{H}_\mathbf{f}^\top \tag{83}$$

Following (82), we can write the information with $\mathbf{U_p}$ as:

$$\Sigma_\mathbf{x}^{(NSpr)} = \sum_{j=1}^{m} \left(\left(\mathbf{U}_\mathbf{p}^{(j)\top} \mathbf{H}_\mathbf{x}^{(j)} \right)^\top \mathbf{U}_\mathbf{p}^{(j)\top} \mathbf{H}_\mathbf{x}^{(j)} \right) = \sum_{j=1}^{m} \left(\Sigma_{\mathbf{x}^{(j)}}^{(NSpr)} \right) \tag{84}$$

However, we still need to prove that (84) and (82) are equivalent.

Note that, since $\mathbf{U_p}$ and $\mathbf{U_n}$ are both the null space of $\mathbf{H_f}$, and the columns of $\mathbf{U_n}$ can be seen as a basis for $\mathbf{U_p}$. Therefore, we can write it as:

$$\mathbf{U_p} = \mathbf{U_n S} \tag{85}$$

where $\mathbf{S}$ is the scalar matrix describing the coordinates of the each column of $\mathbf{U_p}$ in the basis of $\mathbf{U_n}$. Since $\mathbf{U_p}$ is a projector, we have:

$$\mathbf{U}_\mathbf{p}^\top \mathbf{U_p} = \mathbf{S}^\top \mathbf{S} \tag{86}$$

$$\mathbf{U_p} \mathbf{U}_\mathbf{p}^\top = \mathbf{U_n S} \tag{87}$$

Therefore

$$\left(\mathbf{U_n} - \mathbf{S}^\top\right)\mathbf{S} = \mathbf{0} \tag{88}$$

We can see that $\mathbf{S} = \mathbf{U_n}^\top$ is a solution, That is:

$$\mathbf{U_p} = \mathbf{U_n}\mathbf{U_n}^\top \tag{89}$$

Plug this solution into (84) we can prove that it is equivalent to (82). Therefore, the algorithm can be described as:

Algorithm 5 Null-Space Marginalization With Projection

1: Input: ${}_2^1\hat{\mathbf{x}}$, ${}^1\mathbf{x}_\mathbf{f}^{(j)}, \mathbf{n}^{(j)}, j = 1 \dots m$
2: **for** $j \leftarrow 1,\ m$ **do**
3: Compute $\mathbf{H}_\mathbf{x}^{(j)}$ and $\mathbf{H}_\mathbf{f}^{(j)}$
4: Compute $\mathbf{U}_\mathbf{p}^{(j)}$
5: Compute $\Sigma_{\mathbf{x}^{(j)}}^{(NSpr)}$
6: **end for**
7: Compute $\Sigma_\mathbf{x}^{NSpr}$ by Eq. (84)

C.2.5 Null Space Marginalization With Given Rotations

The original method for calculating the null space is Given Rotations. So, we also provide a version for this null space calculation.

$$\Sigma_\mathbf{x}^{(NSgn)} = \sum_{j=1}^{m}\left(\left(\mathbf{U}_\mathbf{gn}^{(j)\top}\mathbf{H}_\mathbf{x}^{(j)}\right)^\top \mathbf{U}_\mathbf{gn}^{(j)\top}\mathbf{H}_\mathbf{x}^{(j)}\right) = \sum_{j=1}^{m}\left(\Sigma_{\mathbf{x}^{(j)}}^{(NSgn)}\right) \tag{90}$$

Therefore, the algorithm can be described as:

Algorithm 6 Null-Space Marginalization With Given Rotations

1: Input: ${}_2^1\hat{\mathbf{x}}$, ${}^1\mathbf{x}_\mathbf{f}^{(j)}, \mathbf{n}^{(j)}, j = 1 \dots m$
2: **for** $j \leftarrow 1,\ m$ **do**
3: Compute $\mathbf{H}_\mathbf{x}^{(j)}$ and $\mathbf{H}_\mathbf{f}^{(j)}$
4: Compute $\mathbf{U}_\mathbf{gn}^{(j)\top}\mathbf{H}_\mathbf{x}^{(j)}$ with given rotations
5: Compute $\Sigma_{\mathbf{x}^{(j)}}^{(NSgn)}$
6: **end for**
7: Compute $\Sigma_\mathbf{x}^{NSgn}$ by Eq. (90)

C.3 Marginalization Runtime Results

we use the same setup as experiments before. We have 121 features are going to be marginalized, and compare the running time for the whole optimization and marginalization with 500 times. The results are as following:

For each marginalization method, we also run 500 times and take average to compare each marginalization performances.

Algorithm 7 Given Rotations

Input: $\mathbf{H_f}$, $\mathbf{H_x}$, (r, c): rows and columns of $\mathbf{H_f}$
for $i \leftarrow 1$, c **do**
  for $j \leftarrow 1$, r **do**
    Compute $a_{ii} = \mathbf{H_f}_{(i,i)}$ and $a_{ji} = \mathbf{H_f}_{(j,i)}$
    if $a_{ii} == 0$ **then**
      $c = 0$ and $s = 1$
    else
      $c = \frac{a_{ii}}{\sqrt{a_{ii}^2 + a_{ji}^2}}$ and $s = -\frac{a_{ji}}{\sqrt{a_{ii}^2 + a_{ji}^2}}$
    end if
    Construct $\mathbf{R} = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}$
    $\mathbf{H_f}_{([i,j],1:3)} = \mathbf{R}\mathbf{H_f}_{([i,j],1:3)}$
    $\mathbf{H_x}_{([i,j],:)} = \mathbf{R}\mathbf{H_x}_{([i,j],:)}$
  end for
end for

Table 3: Overall Running Time

Marginalization Algorithm	Average Running Time(s)
Optimization with Schur Complement Marginalization	0.0440574
Optimization with Given Rotations	0.00953969

Table 4: Marginalization Method Comparison

Marginalization Algorithm	Average Running Time(s)	Speed Rank
Full Schur Complement	0.0321067	5
Null Space with Given Rotations	0.0000867469	2
Null Space with HouseHolder QR	0.000116933	3
Null Space with Direct HouseHolder	0.000118446	4
Null Space with Projection Form	0.0000757341	1

References

[1] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *International Journal of Robotics Research*, Dec. 2014.

[2] G. Huang, K. Eckenhoff, and J. Leonard, "Optimal-state-constraint EKF for visual-inertial navigation," in *Proc. of the International Symposium on Robotics Research*, (Sestri Levante, Italy), Sept. 12–15, 2015.

[3] K. Eckenhoff, P. Geneva, and G. Huang, "High-accuracy preintegration for visual-inertial navigation," in *in Proc. of International Workshop on the Algorithmic Foundations of Robotics*, December 18-20 2016.

[4] G. Huang, M. Kaess, and J. Leonard, "Consistent sparsification for graph optimization," in *Proc. of the European Conference on Mobile Robots*, (Barcelona, Spain), pp. 150–157, Sept. 25–27, 2013.

[5] E. Nerurkar, K. Wu, and S. Roumeliotis, "C-KLAM: Constrained keyframe-based localization and mapping," in *Proc. of the International Conference on Robotics and Automation*, (Hong Kong, China), pp. 3638–3643, May 31-June 7 2014.

[6] K. Eckenhoff, L. Paull, and G. Huang, "Decoupled, consistent node removal and edge sparsification for graph-based slam," in *in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, October 9-14 2016.

[7] 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), pp. 3565–3572, Apr. 10–14, 2007.

[8] M. Li and A. I. Mourikis, "Optimization-based estimator design for vision-aided inertial navigation," in *Proc. of the Robotics: Science and Systems Conference*, (Sydney, Australia), July 2012.

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

[10] M. Li and A. I. Mourikis, "Vision-aided inertial navigation with rolling-shutter cameras," *International Journal of Robotics Research*, vol. 33, pp. 1490–1507, Sept. 2014.

[11] M. Li and A. I. Mourikis, "Online temporal calibration for Camera-IMU systems: Theory and algorithms," *International Journal of Robotics Research*, vol. 33, pp. 947–964, June 2014.

[12] C. X. Guo and S. I. Roumeliotis, "Imu-rgbd camera navigation using point and plane features," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Tokyo, Japan), pp. 3164–3171, Nov. 3–7, 2013.

[13] C. Guo, D. Kottas, R. DuToit, A. Ahmed, R. Li, and S. Roumeliotis, "Efficient visual-inertial navigation using a rolling-shutter camera with inaccurate timestamps," in *Proc. of the Robotics: Science and Systems Conference*, (Berkeley, CA), July 13–17, 2014.

[14] D. G. Kottas and S. I. Roumeliotis, "Efficient and consistent vision-aided inertial navigation using line observations," in *Proc. of 2013 IEEE International Conference on Robotics and Automation*, (Karlsruhe, Germany), pp. 1540–1547, May 6-10 2013.

[15] D. G. Kottas and S. I. Roumeliotis, "An iterative kalman smoother for robust 3d localization on mobile and wearable devices," in *Proc. of IEEE International Conference on Robotics and Automation*, (Seattle, WA), pp. 6336–6343, May 26-30 2015.

[16] Y. Yang and G. Huang, "Acoustic-inertial underwater navigation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017.

[17] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, pp. 1–21, Feb 2017.

[18] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, pp. 1365–1378, Dec. 2008.

[19] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[20] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation*. New York: John Willey and Sons, 2001.

[21] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition*, (Providence, RI), June 18-20 2012.