Toward Invariant Visual-Inertial State Estimation using Information Sparsification

Shih-Chieh (Jerry) Hsiung

CMU-RI-TR-18-50

August 2018

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Michael Kaess (Chair)
Simon Lucey
Kumar Shaurya Shankar

*Submitted in partial fulfillment of the requirements
for the degree of Master of Science.*

To My Family

Abstract

In this work, we address two current challenges in real-time visual-inertial odometry (VIO) systems — efficiency and nonlinearity. To this end, we present a novel approach to tightly couple visual and inertial measurements in a fixed-lag VIO framework using information sparsification. To bound computational complexity, fixed-lag smoothers perform marginalization of variables but consequently deteriorate accuracy and especially efficiency. Current state-of-the-art approaches work around this by selectively discarding measurements and marginalizing additional variables. However, such strategies are sub-optimal from an information-theoretic perspective. In contrast, our approach formulates an optimization based on Kullback-Leibler divergence to preserve most of the information. To validate our approach, we conduct extensive real-time drone tests and perform comparisons to current state-of-the-art fixed-lag VIO methods in the EuRoC visual-inertial dataset. The experimental results show that the proposed method achieves competitive and superior accuracy in almost all trials.

In achieving a more efficient and accurate state estimator, the second part of the work presents the on-going progress in formulating an optimization-based VIO system using matrix Lie groups. Inspired by the recently developed Invariant-EKF framework, the proposed framework presents better convergence and addresses the consistency problem commonly seen in EKF-based and fixed-lag frameworks. In particular, we provide detailed derivations of a novel IMU preintegration framework using the group affine properties. Simulation results show our proposed formulation allows the nonlinear optimizer to converge with significantly fewer iterations, as compared to the state-of-the-art IMU preintegration scheme.

Acknowledgments

First and foremost, I am grateful for my advisor, Prof. Michael Kaess, for the inspiration, suggestion, and guidance of my research project. I would also like to thank my committee members, Prof. Simon Lucey and Kumar Shaurya Shankar for providing advice on my thesis. I am sincerely thankful for Ming Hsiao, Eric Westman, and the entire Robot Perception Lab (RPL) for the countless hours of discussion and laughter. I would not have accomplished so much without all your help, and I feel really honored to be a member of RPL.

I would like to thank my family, friends from the Harvey Mudd College, and Bai Xue for the moral support, especially during stressful times. It is through your encouragement that I am able to push myself beyond what I knew I could accomplish.

Finally, to all my friends at CMU: *Thank you!* Each one of you has made my two-year journey at Carnegie Mellon the most unique and enjoyable experience in my life. To name a few: Paloma “best partner” Sodhi, Sanjiban “has a brother” Choudhury, Aditya “HAHAHA” Dhawale, Xuning “feel like knowing you forever” Yang, Alex “champ” Spitzer, Ratnesh “lit” Madaan, Ming “my master” Hsiao, Eric “you are so tall” Westman, Rogerio “ma spanish hombre” Bonatti, Vishal “the good life” Dugar, Puneet “congrats” Puri, Achal “SD bro” Dave, Anirudh “24 challenge” Vemula, Pragna “foooood” Mannan, Nadine “schedule everything” Chang, Tess “Ni-Hao” Hellebrekers, Himanshi “always smiley” Yadav, Kumar Shaurya “yay” Shankar, Ceci “don’t be nervous” Morales, Cara “love whiskey” Craig, Druv “roboman” Saxena, Wennie “stories” Tabib, Roberto “snowboard master” Shu, Cormac “spuuud” Omeadhra, Mike “never drink with me” Lee, Logan “good puppy” Ellis, Lauren “my snack sharer” Liu, Zimo “Sup XXX” Li, Jack “late night labmate” Yang, Bing “where are you” Ho, Sudu “tsao-san” Suresh, Cherie “Pate” Ho, Micah “my good neighbor” Corah, Akshay “goaaallll” Hinduja, Rohit “cam man” Garg, Kevin “I like your doughnut shirt” Pluckter, John “the reason VIO flies” Keller, Geetesh “awesome pilot” Dubey, Silvio “thank you” Maeta.

Contents

Chapter 1

Introduction

1.1 Motivation

Existing and emerging technologies like mobile robotics and artificial intelligence will drastically change how we live our daily lives. For example, autonomous cars will make transportation accessible to everyone; service robots could improve the quality of children and elderly cares; agricultural robots could increase general food production. With various possible applications, however, all mobile robotics need to navigate through a known or unknown environment.

To navigate autonomously in an unknown environment, an intelligent mobile robot requires the abilities to control, trajectory planning, obstacle avoidance, state estimation, and mapping, etc. Of all mentioned, state estimation is the most vital part because it is the core to all the other algorithms. For instances, a controller uses state estimates as feedback systems for feedback controls [46]; a trajectory planner uses state estimates to plan shortest paths [19]; an obstacle avoidance algorithm uses state estimates to calculate the relative positions of the robot and its obstacles [56]; a mapping algorithm uses state estimates to build a consistent map of the surroundings [15]. Knowing the robot status from an accurate and robust state estimation algorithm enables robot operating smoothly and correctly.

At its core, a state estimator provides information about a robot's "states," which definition depends on the applications. In the context of robot navigation such as drone flying or autonomous car driving, the robot states include but not limited to 3D position, orientation, and sometimes linear and angular velocities. To estimate these quantities, a state estimator could utilize external sensors systems such as VICON or OptiTrack [58] to provide accurate position information. However, relying on an external sensor system restrains the operational area because the fields of view of these systems are fixed. On the other hand, a state estimator could fuse information from various onboard sensors such as global positioning system (GPS) receivers, cameras, lidars, inertial measurement units (IMU), etc. For example, when a robot is operating in an outdoor environment, its state estimator combines GPS with IMU to provide the location of a robot [28]. In an indoor and GPS-denied environments, the state estimator utilizes camera and lidar with IMU to calculate the pose of a robot [9]. Often, when the problem of state estimation for robot localization joins with mapping the environment, it is famously known as the simultaneous localization and mapping (SLAM) problem [16]. SLAM is a large and on-going field of research in the robotics community, however, this thesis focuses on the localization part of SLAM.

State estimators utilizing GPS signals are ubiquitous and well-developed. A simple example is when a person uses his or her cellphone and Google Map to navigate through the complex environment. However, the GPS signal easily gets shielded and undergoes multi-path reflections in an urban or indoor environment such as in New York City or inside a tunnel. To account for these problems, researchers have been looking for alternative solutions that utilize different onboard sensors. In the recent years, along with the development of cameras and microelectro-mechanical sensors (MEMS), much attention has been given to methods combining cameras and inertial sensors for the complementary nature of their information. While inertial sensors are responsive in short-term dynamic movements, cameras provide rich exteroceptive information for long-term navigation. In particular, Visual-Inertial Odometry (VIO) has shown effectiveness in challenging scenarios such as indoor and GPS-denied environments compared to previously existing methods. The ability to navigate indoors is particularly important in applications such as search and rescue, damage inspection, indoor exploration and mapping. In densely populated cities and natural environments like forests, VIO could be used to better aid localization when combined with existing methods that utilize GPS and other global reference points.

VIO is now widely used in technologies like virtual reality (VR), augmented reality (AR), autonomous drone flying, an autonomous car driving to name a few. The current state-of-the-art VIO for mobile robotics has demonstrated unprecedented accuracy in determining a robot's position and orientation using just cameras and IMUs. Couple example are OKVIS [48] and VINS-MONO [59] However, these methods are still not perfect in providing the optimal solution from the information theoretical perspective. Often, these methods still fail in situations such as dynamics robot maneuvers [9]. Therefore, there remain challenges ahead in achieving a "perfect" VIO state estimator.

In this thesis, we take an in-depth and theoretical look at the VIO optimization algorithm that fuses sensor information, and provides insights using information theory and differential geometry. We also propose improvements upon the state-of-the-art methods. In the remainder of this chapter, we discuss current challenges in modern VIO systems and summarize the remainder of this thesis which details our contributions to address these challenges.

1.2 Current Challenges and Scope

Designing a real-time and robust VIO state estimator for mobile robots is challenging for the following two reasons:

1. Limitations such as computational resources and energy capacity reduce computing power

available to run the algorithm.

2. Estimating orientation of a robot is complex because the mathematical functions and models are highly nonlinear.

In considering efficiency, robustness, and accuracy with regards to the two challenges, typical VIO systems employ either an extended Kalman filter (EKF) for or a graph-based optimization algorithm to combine inertial information with existing visual odometry methods. Both kinds of methods model the state estimation problem using Bayesian probabilistic theory and information theory to deal with sensor noise. While an EKF is known for its efficiency, it is generally less accurate for nonlinear estimation than an optimization approach [9]. However, an optimization-based method is often expensive in computations. To combine the best of EKF and optimization-based methods, in this thesis we focus on a *fixed-lag smoothing VIO*, which performs optimization using graphical models on a fixed set of variables (optimization window). A fixed-lag smoothing VIO is able to bound computational complexity while achieving better accuracy compared to an EKF, as later shown in Section 4.

1.2.1 About VIO Efficiency

When performing an optimization-based VIO state estimation, like other general SLAM problems, the optimizer exploits the *sparsity* of the mathematical model to solve the complex optimization problem. This enables the state estimator to run in real-time. The sparsity comes from the assumptions of measurement independence (also called the Markov Assumption). The sparsity maintains when the robot operates but the optimization problem gets larger when more sensor measurements arrive.

In order to maintain a constant optimization window, a fixed-lag smoother performs *marginalization* of variables by removing the oldest ones. Imagine at a given time, there is a set of variables we would like to estimate, but estimating every variable is too expensive for the computer. Marginalization is the process to eliminate a subset of variables for estimation to reduce the computation. However, instead of simply discarding the variables, the optimization would consider the effects of marginalized variables in the remaining set of variables. As state estimator runs, one could imagine that the remaining variables accumulate more and more information from marginalizations and as a result, the VIO system is *densified*. A densified system would cause the state estimator to run increasingly slower.

Consecutive marginalization poses a major problem to a fixed-lag smoothing VIO system because it reduces efficiency and can cause catastrophic failures to robot operation without providing timely state estimation. The current state-of-the-art VIO solutions such as OKVIS [48] and VINS-MONO [59] selectively discard measurements in order to avoid such issue. As opposed to existing methods, we propose a novel fixed-lag VIO system that performs *sparsification* in the first part of this thesis. Our system provides a more information theoretically optimal way to reduce computation in order to maintain efficiency.

1.2.2 About VIO Nonlinearity

The marginalization process in a fixed-lag VIO not only reduces efficiency but also causes *inconsistency*. A consistent state estimator does not underestimate the uncertainty of its solution, which means the uncertainty captures the error between the groundtruth path and estimated path. However, current literature has shown that marginalization potentially makes the state estimator to wrongly estimate (underestimate) the uncertainty and therefore degrades the accuracy. Mathematically the problem stems from the nonlinearity of the state estimation problem, and affects both EKF systems and fixed-lag smoothers [36].

When estimating a nonlinear system, iterative methods such as Gradient Descent or Gauss-Newton [9] methods are employed. At each iteration, these methods first linearize the nonlinear problem and evaluate the Jacobians using the linearization of the state variables (also called the linearization points). Therefore, the Jacobian values depend on the state estimates. As the state variables get updated, the linearization points also change. The problem arises when the optimization window combines Jacobians based on previous linearization points and the updated ones. The combination of old and new linearization points was proven to greatly reduce the accuracy of the estimator [36]. To address this issue, we propose an *invariant* optimization framework in the second part of this thesis, following the recent publications on invariant EKF [4, 6]. We show that if we perform state estimation on a different nonlinear space, the consistency issue can be eliminated, which provides better convergence and accuracy. In particular, we explore state estimation on matrix Lie group, which considers the underlying mathematical geometry of the nonlinear problem and connects optimization methods with differential geometry. We present an IMU preintegration method on matrix Lie group based on [29], and show interesting characteristics such as faster optimization convergence comparing to [29] while the Jacobians are independent of the state estimates. However, the complete VIO system on matrix Lie group will be part of the future work.

1.3 Contributions and Organization

In this thesis, we focus on addressing the two challenges listed above by improving upon the state-of-the-art VIO methods. First, we present a *fixed-lag VIO system with information sparsification* that addresses the challenge of efficiency. We then formulate the state estimation problem using *group affine properties on matrix Lie group* [4] and propose a novel IMU preintegration framework using that property. The main contributions of this thesis are listed as follows:

- To the best of our knowledge, this is the first work employing sparsification in the context of fixed-lag VIO to maintain sparsity while minimizing information loss. In this thesis, we detail the derivation and design of our sparsification methodology, which retains the sparsity and nonlinearity of the graphical model in the presence of marginalization.

- To evaluate the proposed VIO system, we conduct real-world experiments with our software pipeline running onboard an Autel X-Star drone, and provide extensive comparisons of our approach to the current state-of-the-art fixed-lag VIO methods on the EuRoC visual-inertial dataset [8] and simulated environment. Moreover, we provide a detailed run-time analysis to demonstrate that the proposed VIO system is suitable for real-time application and suggest ideas for further acceleration of the proposed algorithm.

- To the best of our knowledge, this is the first work employing the invariant framework in an optimization-based VIO system. We formulate the IMU preintegration method on matrix Lie group, based off the work of [29]. This thesis includes the derivations of IMU preintegration measurement models, the jacobians, and covariance propagation using Lie Group theory.

- To evaluate the theoretical contribution, we conduct simulations to show that the proposed IMU preintegration on matrix Lie group has better convergence characteristics compared to the state-of-the-art from the GTSAM library [17].

The thesis is organized as follows. Chapter 2 discusses background and related work. We begin the reviews by introducing existing VIO frameworks and their comparisons. We then delve into more details for existing literature about the two topics covered by this thesis — efficiency and non-linearity in VIO. Chapter 3 provides the math preliminaries for fixed-lag smoothing framework using graphical model and optimization on manifold using Lie Group theory. Chapter 4 details the derivations of our VIO system with information sparsification, including results from public datasets, simulations, and real-time flight tests. In Chapter 5, we introduce the recently developed group affine system and detail the derivation of our IMU preintegration method using group affine system. We then show simulated results and provide comparisons to the current GTSAM implementations. Finally, Chapter 6 summarizes the thesis contributions and presents the on-going future work.

Chapter 2

Background and Related Work

The research of VIO belongs to the general SLAM problems, which has produced a vast amount of literature in the last few decades. While EKF was traditionally used for the SLAM problem, the probabilistic modeling of the SLAM problem has produced a more general graph-based formulation using *factor graph*. Interested readers are referred to [16], which provides a detailed tutorial-like introduction of SLAM using factor graph. In Chapter 3.1 we provide a broad overview of performing state estimation using graphical models. In this chapter, we focus on the related work of VIO specifically.

2.1 Visual Inertial Odometry (VIO)

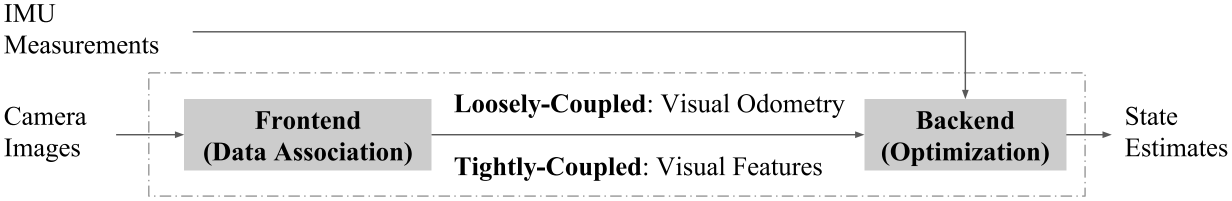


Figure 2.1: A general VIO system consists of a frontend model and a backend module. The frontend module preprocesses images to output either visual features or visual odometry while the backend module performs joint optimization with IMU measurements.

Existing VIO literature introduces various methods for combining visual and inertial data. In general, all VIO algorithms consist of a frontend and a backend module as shown in Fig. 2.1. The frontend component of a state estimator performs data processing of visual information from camera or lidar. Depending on the frontend algorithms, there are both *direct* and *indirect* methods. For example, Direct methods, such as LSD-SLAM [26], solve for camera transformation by minimizing the photometric error based on pixel intensity values. On the other hand, indirect methods like PTAM [44] and ORB-SLAM [53, 55] compute visual features and associate the features across camera frames. Indirect methods then compute camera transformation using these features, hence the name “indirect”.

Once the visual information is processed, either in the form of direct camera transformation or visual features, the processed information is incorporated into the backend module in two ways. A *loosely-coupled* system uses the frontend module such as PTAM [44] or DSO [24] to compute visual data as odometry information [25, 45, 69]. The system then combines the odometry data with inertial data to compute the joint solution. In contrast, a *tightly-coupled* system incorporates computed visual features and inertial data in a single framework [29], [48, 54, 59, 64], which is shown to be the more accurate approach [18, 48].

Either a *loosely-coupled* system or a *tightly-coupled* system can be further categorized into filtering-based [7, 52, 70] and optimization-based methods [41, 48, 59, 64] depending on the backend solver. Filtering-based methods, such as MSCKF [52], are computationally efficient; however, they are known for accumulated linearization errors and inconsistency issues especially in highly nonlinear systems [62]. Huang et al. [37, 38], Li et al. [49], and Hesch et al. [34] propose the First Estimate Jacobian (FEJ) EKF and the Observability Constrained (OC) EKF to resolve such issues by enforcing fixed linearization points. In contrast, optimization-based methods solve for the optimal estimate by iteratively minimizing the measurement residual costs. In general, optimization-based methods formulate the problem with a graphical model such as the *factor graph*, and utilizes graphical model solvers for efficient optimization. They require more computational resources but achieve higher accuracy. However, the sparse nature of VIO allows existing optimization-based methods to utilize sparsity and apply efficient solvers such as iSAM2 [43], g2o [47], Ceres [1], and SLAM++ [40] to achieve real-time performance.

Table 2.1: Categories of existing VIO methods

	Frontend	Backend	Coupling
SVO-MSF (Faessler et al., JFR 2016)	Indirect	Filter-based	Loosely-Coupled
MSCKF (Mourikis et al., ICRA 2007)	Indirect	Filter-based	Tightly-Coupled
ROVIO (Bloesch et al., IROS 2015)	Indirect	Filter-based	Tightly-Coupled
OKVIS (Leutenegger et al., IJRR 2015)	Indirect	Optimization-based	Tightly-Coupled
VINS-MONO (Qin et al., arXiv 2017)	Indirect	Optimization-based	Tightly-Coupled
SVO-GTSAM (Forster et al., TRO 2017)	Indirect	Optimization-based	Tightly-Coupled
Proposed (Hsiung et al., IROS 2018)	**Indirect**	**Optimization-based**	**Tightly-Coupled**

Our method, similar to [48, 59], focuses on the tightly-coupled fixed-lag smoothing framework, which combines advantages from both filtering and batch optimization methods [20]. A fixed-lag smoother maintains a bounded computational complexity by fixing the number of target variables in the optimization window while allowing nonlinear optimization to solve for the optimal solution. Table 2.1 summarizes the categories of current VIO systems and show where our proposed work fit in the existing literature.

For more in-depth comparisons of current state-of-the-art VIO algorithms, readers are referred to [18]. The paper provides a complete benchmark including root mean squared error (RMSE), memory usage, CPU usage, and computation time on a variety of hardware platforms for all the open-source VIO systems (Fig. 2.2). From the result of the paper, it is clear to see that fixed-lag VIO systems achieve the best accuracy of all and are able to run real-time with slightly more computations than filtering-based methods.

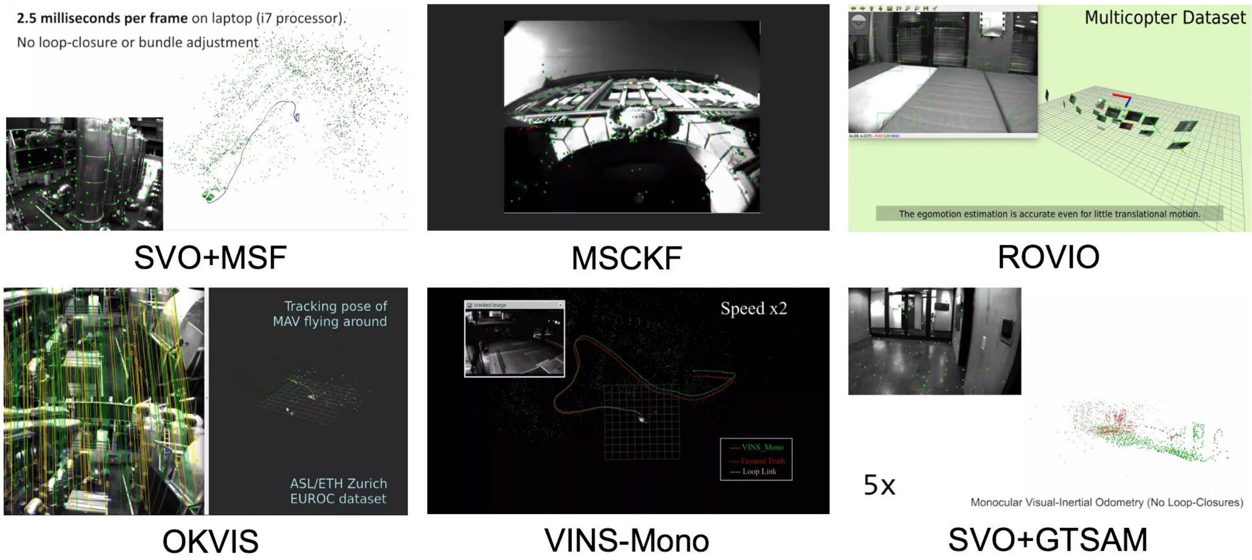

2.5 milliseconds per frame on laptop (i7 processor).
No loop-closure or bundle adjustment
SVO+MSF
MSCKF
Multicopter Dataset
The egomotion estimation is accurate even for little translational motion.
ROVIO
Tracking pose of MAV flying around
ASL/ETH Zurich EUROC dataset
OKVIS
Speed x2
VINS_Mono
Ground Truth
Loop Link
VINS-Mono
5x
Monocular Visual-Inertial Odometry (No Loop-Closures)
SVO+GTSAM

Figure 2.2: This figure lists current available open-source VIO systems [Image credit: [18]].

2.2 Sparsification in Graph-based SLAM Framework

To be able to efficiently solve a graph-based optimization for SLAM, existing methods exploit its sparsity in the information form [27, 30, 67]. Sparsity is an important property of a SLAM system [16, 30], which both filtering-based methods and optimization-based methods benefit from. For instance, Eustice et al. [27] and Thrun et al. [63] exploit the sparse structure and develop information filters to efficiently solve the landmark-based SLAM problem. Existing graph solvers [1, 42, 43, 47] exploit sparsity for efficient optimization. In factor graph SLAM, the information matrix specifies the weights and connectivity between variables [16]. However, as the optimization window grows over time, a fixed-lag smoother needs to marginalize variables to maintain a constant computational complexity [71].

Successive marginalizations create “fill-in”, additional non-zero entries in the otherwise sparse information matrix, which significantly reduces computational efficiency [30]. To avoid such issue, current state-of-the-art methods such as OKVIS [48] and VINS-MONO [59] 1) selectively discard measurements for sparsity and 2) marginalize additional variables. From an information-theoretic perspective, the information content of the optimization window is reduced and the marginalized variables are no longer optimize-able. The solution to the consecutive optimizations will no longer be optimal with respect to the original problem. As opposed to existing methods, our algorithm addresses the aforementioned issues by incorporating information sparsification to minimize the information loss while maintaining sparsity.

Current literature in graph sparsification focuses on the context of large SLAM pose graphs because sparsification originates from reducing the computation for large pose graph SLAM system [14, 22, 35, 39]. Wang et al. [68] formulate the sparsification problem by minimizing Kullback-Leibler divergence (KLD) in a laser-based SLAM application. Carlevaris-Bianco et al. [10] propose a generic linear constraint (GLC) which utilizes the Chow-Liu tree to approximate the information of the Markov blanket. Our work follows Mazuran et al.'s Nonlinear Factor Recovery (NFR) [51], which uses specified nonlinear factors to approximate the dense prior by KLD optimization. To our knowledge, our work is the first to demonstrate online sparsification in a fixed-lag VIO framework. We show that our methodology achieves state-of-the-art performance on a public test dataset and is suitable for real-time state estimation.

2.3 Group Affine Property on Matrix Lie Group

To marginalize out old variables, a fixed-lag VIO system first linearizes state variables and uses Schur-Complement to compute the resultant information. However, when this linearized information is reintroduced back to the VIO graphical model, the combination of both previous linearization points and the updated ones causes inconsistency of the state estimator [36]. Using observability gramian, [36] has mathematically shown that a traditional EKF SLAM or a fixed-lag smoother, when solving a highly nonlinear problem, becomes inconsistent because some Jacobians in the system are dependent on the old linearization points. To account for the issue, First-Estimate Jacobian fixed-lag smoothing was introduced by [38], which evaluates Jacobians with their respective linearization points. Huang et al. from [36] later proposes observability-constrained fixed-lag smoother which evaluate Jacobians at the optimal state estimates that also satisfy observability constraints.

Barrau et al. from [4], however, has provided a different insight on the consistency issue. The authors discovered the Jacobian evaluations are the result rather than the cause of inconsistency issue. They showed that the root cause of inconsistency is the space where the state estimation is performed. Rather than estimate states on the Euclidean space, the *invariant EKF* framework proposed by [3] estimates states on the manifold. Utilizing tools from differential geometry, the same authors propose a general dynamic system in [4] that satisfies the *group affine property*. They have shown the traditional state estimation on Euclidean space also satisfies such property.

The techniques of optimization on manifold have nice mathematical properties to solve rigid body transformations because of its rotation representation on vector space. There are existing literature such as [2, 29, 50] also performs state estimation on manifold. The state space in these works, however, are on the Special Orthogonal Group ($SO(3)$) or the Special Euclidean Group ($SE(3)$). The invariant framework by [12] expands the state space to a larger Lie Group, and apply optimization on manifold to solve the consistency issue. This new insight has recently promoted numerous related work. For example, [32] uses invariant EKF along with contact sensors for bipedal robot estimation. [3, 72] has shown the invariant framework resolves the consistency issue in EKF SLAM without using observability constraints. [33] has used the invariant framework to develop a VIO system using EKF. To the best of our knowledge, the closest related work to ours is by [12]. The author employs the invariant framework to the optimization-based method for factor graph. However, the author provides little evaluations and the results are shown using 2D navigation problems.

Our work is based on [29], where we formulate a collection of IMU measurements as one preintegrated measurement. Different from [29]. we derive the preintegration using the group affine properties and show that it possesses better convergence, by comparison, using simulated datasets.

Chapter 3

Math Preliminaries

3.1 State Estimation using Factor Graph

When a robot travels in a trajectory, the onboard computer would obtain information from the time synchronized sensors at constant time intervals (assuming the sensors are time-calibrated). As shown in Fig. 3.1, the sensor information reflects the snapshots of the robot at that particular time instants.

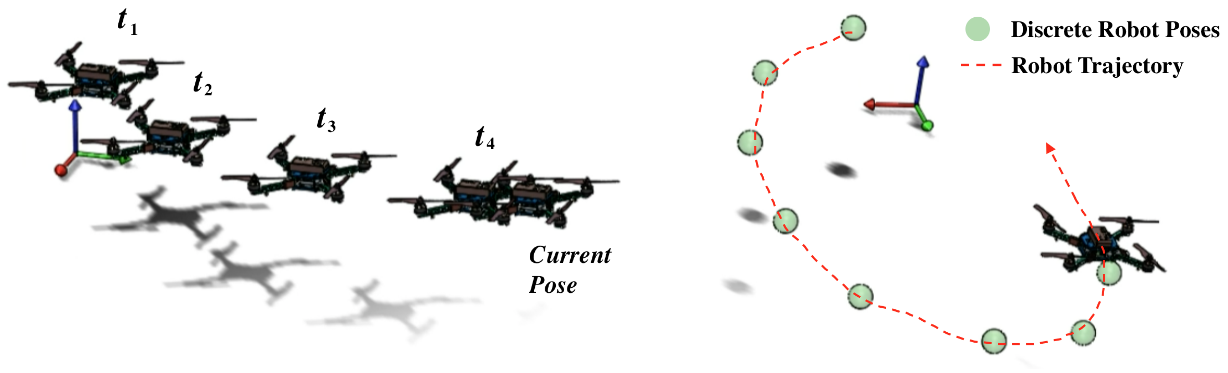
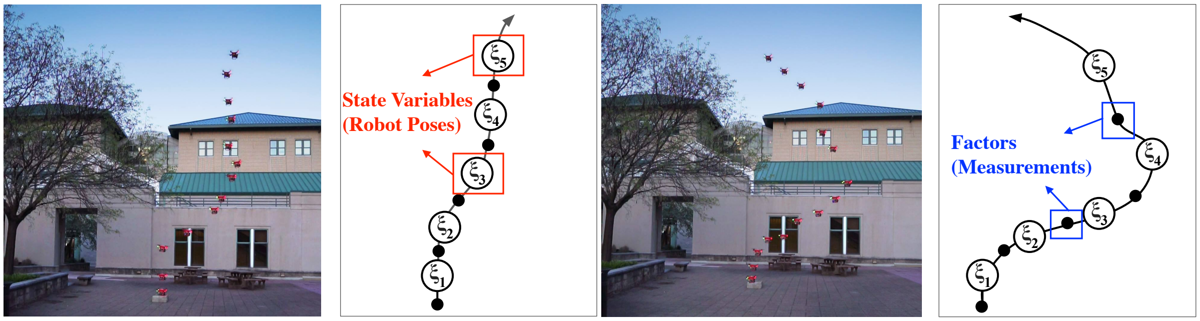


Figure 3.1: **(a)** shows that at constant time intervals, the robot obtains information that reflects the status of the robot at that particular time instant. We could then collect all the sensor measurements in the timely manner and solve for the discrete robot poses as shown in (**b**).

Following [16], we collect all the sensor measurements ordered by time, and utilize factor graph to formulate the VIO problem. A factor graph is a bipartite graph that consists of variable nodes and factors. Each variable node represents a state variable that we would like to estimate, and each factor represents a measurement model that specifies the relationship between variables.

As a simple example, suppose we are interested in solving an autonomous drone taking off using a pose graph (a factor graph with only poses as variables), each variable node is a discrete pose ξ_t

at a time instant, and each factor represents pose-to-pose measurement model. A factor graph provides mathematical representations for the underlying probabilistic inferences, but also provides a visualization for physical robot trajectories. The example pose graph shown in Fig. 3.2 describes the relationship between physical visualization and a factor graph.



(a) (b)

Figure 3.2: These two figures illustrate using factor graphs to represent discrete robot pose. Each pose can be modeled as a variable, and the pose-to-pose measurement such as odomety are factors that connect pair of poses. A prior is given to the first pose to fixate the first pose to the origin.

In the physical world, however, sensor measurements are polluted by noises. To incorporate noise in a measurement, we formulate a general measurement model of the form

$$z = h(x) + v, \quad v \sim \mathcal{N}(0, \Sigma) \tag{3.1}$$

where we assume the noise itself is a zero-mean Gaussian v with the standard deviation Σ. One can perceive a factor graph as a collection of noisy measurements and itself a high-dimensional Gaussian distribution. We can then solve the factor graph by formulating it into a *maximum a posteriori* (MAP) problem commonly known in the statistics and machine learning communities. That is, solving for the best state estimates that matche the observations. Following the previous example in Fig. 3.2, the posterior can be represented by

$$\begin{aligned}
\xi^\star &= \underset{\{\xi_1,\xi_2,\ldots,\xi_k\}}{\arg\max} \; P(\{\xi_1, \xi_2, \ldots, \xi_k\}|Z) \\
&\approx \underset{\{\xi_1,\xi_2,\ldots,\xi_k\}}{\arg\max} \; P(\xi_1)P(Z|\{\xi_1, \xi_2, \ldots, \xi_k\}) \qquad \text{(Bayes rule)} \\
&= \underset{\{\xi_1,\xi_2,\ldots,\xi_k\}}{\arg\max} \; P(\xi_1) \prod_{(i,j)\in Z} P(z_i|\xi_i, \xi_j) \qquad \text{(Markov assumption)}
\end{aligned} \tag{3.2}$$

where $Z = \{z_1, z_2, \ldots, z_i\}$ is the collection of all measurements, $P(\xi_1)$ is the prior probability, and $P(z_i|\xi_i, \xi_j)$ is the pose-to-pose measurement.

Using Gaussian assumption, this can be reduced into a least-squares problem by taking the negative log-likelihood

$$\begin{aligned}\xi^\star &= \underset{\{\xi_1,\xi_2,\ldots,\xi_k\}}{\arg\min} \; -\log\left[P(\xi_1)\prod_{(i,j)\in Z} P(z_i|\xi_i,\xi_j)\right] \\ &= \underset{\{\xi_1,\xi_2,\ldots,\xi_k\}}{\arg\min} \; \|\mathbf{r}_1\|^2_{\Sigma_1} + \sum_{(i,j)\in Z} \|\mathbf{r}_{ij}\|^2_{\Sigma_z}\end{aligned} \tag{3.3}$$

where $\mathbf{r}_i = z_i - h_i(x)$ denotes the difference between the measurement and predicted measurement using the model, and $\|\mathbf{r}\|^2_\Sigma = \mathbf{r}^\top\Sigma^{-1}\mathbf{r}$ denotes the Mahalanobis distance.

Depending on whether the problem is a linear or a nonlinear system, we can use solvers such as sparse Cholesky for a linear system or iterative methods like Gauss-Newton to solve nonlinear systems. Generally, the pipeline of solving a nonlinear factor graph is shown in Fig. 3.3.

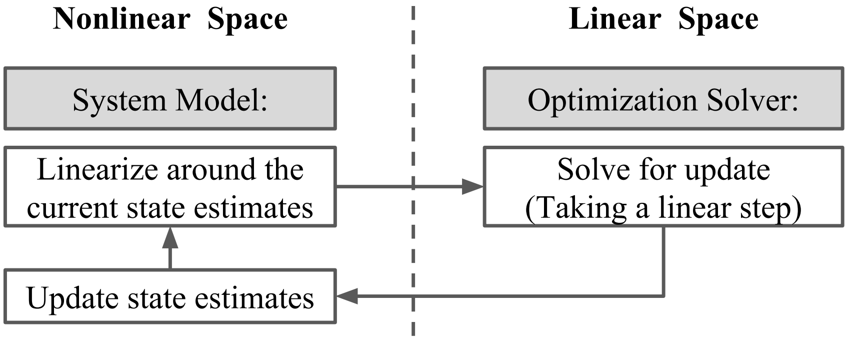

Nonlinear Space
Linear Space
System Model:
Optimization Solver:
Linearize around the current state estimates
Solve for update (Taking a linear step)
Update state estimates

Figure 3.3: The general optimization pipeline for an nonlinear factor graph is during each iteration, the system is linearized around the current state estimates. The solver then solves for an update step. These two steps are repeated until convergence.

3.2 VIO System Formulation

A general VIO system includes both information from inertial sensors and cameras. Below we first introduce the IMU measurement model and camera projection model, and then describe the overall factor graph structure for a VIO system.

3.2.1 Camera and IMU Measurement Models

For visual measurement, we use the standard pinhole camera projection model:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \frac{f_x X_C}{Z_C} + u_0 \\ \frac{f_y X_C}{Z_C} + v_0 \end{bmatrix} \tag{3.4}$$

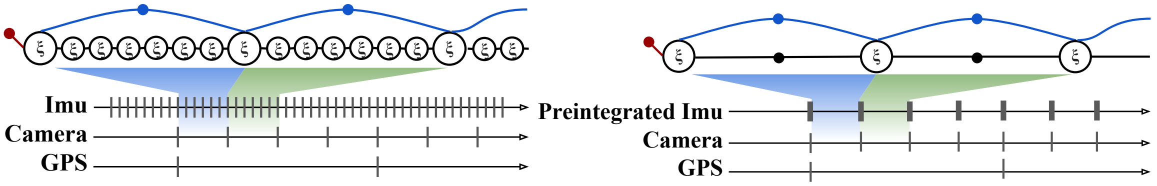
where $l_C = [X_C, Y_C, Z_C]^\top = R_W^C[X_W, Y_W, Z_W]^\top + t_W^C$ is the 3D point of the feature after transformed by the camera pose (R_W, t_W).

For inertial measurements, we adopt the IMU model by [31, 57, 60]. A standard 3-axis IMU measures the acceleration a^B and rotation rate ω^B of the sensor with respect to an inertial frame. The measurements are affected by white noise η and slowly varying bias b:

$$\begin{aligned}\tilde{\omega}^B(t) &= \omega^B(t) + b^g(t) + \eta^g(t)\\ \tilde{a}^B(t) &= R_W^{B\top}(t)(a^W(t) - g^W) + b^a(t) + \eta^a(t)\end{aligned} \tag{3.5}$$

Since the bias changes over time, we will also include them as state variables to be estimated. We then assume the measurement values remains constant between time interval Δt and use the simple Euler method (first-order hold) for integration. In considering real-time implementation, we adopt the *preintegration scheme* by [29], where we accumulate multiple IMU raw measurement into a meta-measurement called the IMU preintegrated measurement.

When fusing multiple sensors, one needs to account for the different operation rate of each sensor. In the visual-inertial case, IMU measurements usually arrive in a much faster rate than camera frames, and therefore the preintegration scheme 1) performs integrations on the local-robot frame and 2) enables integrations of multiple inertial measurements to match the rate of the camera. This allows more efficient optimization without re-integrating all measurements if the state estimates change. The difference between regular IMU scheme and the preintegrated scheme is illustrated in Fig. 3.4.



(a) Normal Imu Formulation

(b) Preintegration Imu Formulation

Figure 3.4: This figure shows IMU takes data much faster than Camera and GPS. The blue factor on the top represent visual odometry connecting two camera frames. **(a)** shows a pose graph incorporate normal IMU measurement as individual measurement. The total number of state variables is large and each IMU measurement needs to be reintegrate if the state estimates change. In contrast, the preintegration scheme in **(b)** proposed by [29] accumulates multiple IMU measurements into one preintegrated measurement in the local frame. This enables the solver to solve the VIO problem efficiently.

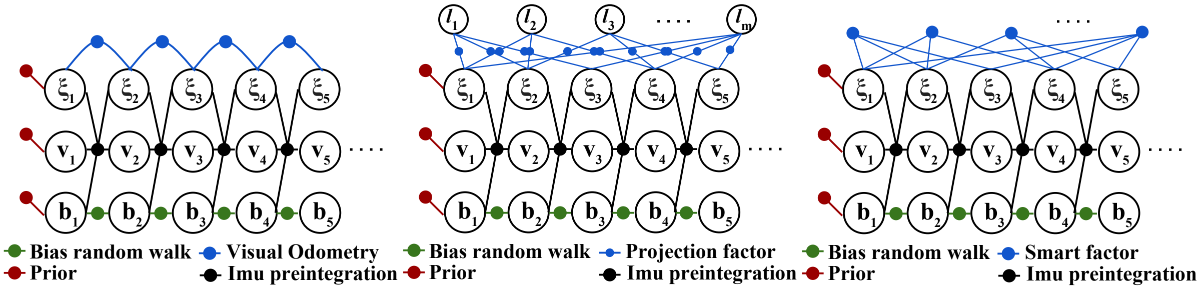
3.2.2 VIO Graph

There are generally two types of VIO factor graph structure depending on the coupling of frontend and backend modules. However both in common are the variables include IMU states (pose ξ, velocity v , IMU biases b), and visual landmarks in 3D.

For a loosely-coupled system, the visual information is pre-processed into pose-to-pose odometry information. Therefore, the graph resembles a pose graph, with only pose-to-pose measurement models from IMU and visual odometry as shown in Fig. 3.5a.

The factor graph for a tightly-coupled VIO system is more complex. The structure of a tightly-couple VIO factor graph is similar to that of a bundle adjustment problem, where landmarks are directly connected to poses by visual measurements. Fig. 3.5b shows a simple VIO system that includes both IMU states and landmarks.

Recently proposed by [29], instead of incorporating each landmark as a variable, the landmarks are marginalized out and become *smart factors* that connect multiple poses. Therefore, each landmark becomes just one factor. This method speeds up the optimization process since landmark variables are eliminated from the state estimation.


l_1 l_2 l_3 .... l_m
....
ξ_1 ξ_2 ξ_3 ξ_4 ξ_5
v_1 v_2 v_3 v_4 v_5 ....
b_1 b_2 b_3 b_4 b_5
Bias random walk
Visual Odometry
Prior
Imu preintegration
Projection factor
Smart factor

(a) Loosely-coupled VIO (b) Tightly-coupled VIO (c) VIO with smart factors

Figure 3.5: The difference between a loosely-coupled VIO **(a)** and a tightly-coupled VIO **(b)** is how the graph incorporate landmarks. While loosely-coupled system incorporates only visual odometry, the tightly-coupled method direcly incorporate landmarks as part of the state variables. **(c)** shows a recent propsed method using smart factors [29], which treats each landmark as one factor rather than a variable.

3.3 Fixed-Lag Smoother Formulation

The difference between a general VIO system and a fixed-lag VIO system is the marginalization step. A fixed-lag smoother bounds computation complexity by ensuring the optimization window being a constant size. To do so, the fixed-lag smoothing VIO system marginalizes old variables when new variables are added. Fig. 3.6a illustrates the concept of an optimization window in a trajectory. A fixed-lag smoother only optimizes the variables in the window so the optimization dimensionality never gets larger.

In addition, our fixed-lag formulation follows the idea of *keyframe*-based VIO, which is also implemented in [29, 48, 59]. Keyframes represent camera frames that retain more information, such as larger baseline or large number of distinct features.

In details, at each time w, our fixed-lag smoother optimizes a window of states :

$$\mathcal{X}_w = \{\mathcal{K}_w, \mathcal{F}_w, \mathcal{L}_w\} \tag{3.6}$$

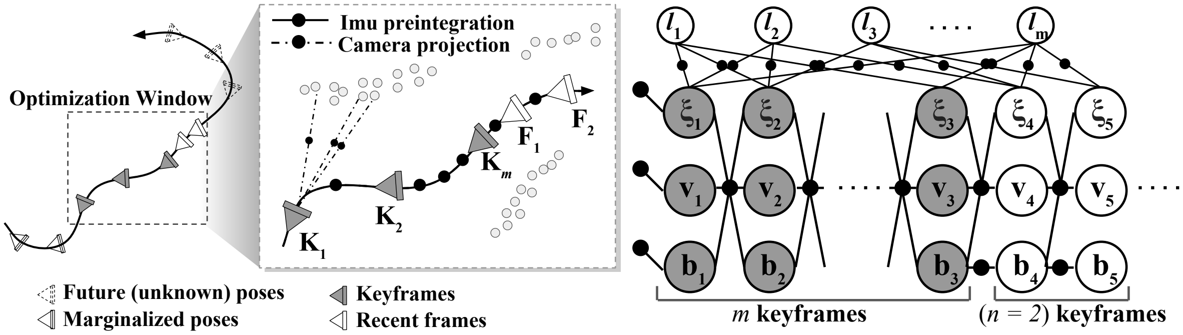
where the set $\mathcal{K} = \{K_1, \ldots, K_m\}$ contains m consecutive keyframes K; $\mathcal{F} = \{F_1, \ldots, F_n\}$ contains n recent frames F; $\mathcal{L} = \{L_1, \ldots, L_p\}$ contains p landmarks L.

For each frame F_i or keyframe K_i, the robot state x_i is defined as:

$$x_i = \left[\xi_i^\top, \mathbf{v}_i^\top, \mathbf{b}_i^\top\right]^\top \tag{3.7}$$

where $\xi \in \mathbb{R}^6$ is the minimum representation of the 3D robot pose, $\mathbf{v} \in \mathbb{R}^3$ the velocity, $\mathbf{b} = \left[\mathbf{b}_a^\top, \mathbf{b}_g^\top\right]^\top \in \mathbb{R}^6$ the IMU accelerometer and gyroscope biases. The measurements $\mathcal{Z}_i$ associated with each F_i or K_i consists of a set of q camera measurements $\mathcal{C}_i = \{c_{i1}, \ldots, c_{iq}\}$ and a *relative* or *marginalized* IMU measurement I_i between two consecutive frames or keyframes respectively. We follow the IMU preintegration method [29] to generate the *relative* IMU measurement. The *marginalized* IMU measurement is detailed in Section 4.2.1. We define each landmark L_j as a 3D point $\mathbf{l} \in \mathbb{R}^3$ in the world frame.

Using the factor graph formulation, we represent each measurement residual $\mathbf{r}$ as a factor in the graph shown in Fig. 3.6.



(a) (b)

Figure 3.6: This figure shows our keyframe-based fixed-lag VIO system. The shaded variable frames represent the keyframes, while the transparent variable nodes are normal frames. Keyframes are more informative and provide larger baselines. They also store distinct features as opposed to adjacent frames.

The two main types of factors are IMU preintegration factors and stereo projection factors. The preintegrated IMU factor between x_i and x_{i+1} allows efficient relinearization during optimization. Its residual consists of three terms:

$$\mathbf{r}_{I_i} = \left[\mathbf{r}_{\Delta\xi_i}^\top \quad \mathbf{r}_{\Delta\mathbf{v}_i}^\top \quad \mathbf{r}_{\Delta\mathbf{b}_i}^\top\right]^\top \tag{3.8}$$

where $\mathbf{r}_{\Delta\xi_{ij}}$ and $\mathbf{r}_{\Delta\mathbf{v}_{ij}}$ and $\mathbf{r}_{\Delta\mathbf{b}_{ij}}$ corresponds to the residuals of pose, velocity, and biases respectively. Interested readers are referred to [29] for detailed derivations for IMU preintegration.

Given the states and the measurement residuals, the optimal solution for the fixed-lag VIO factor graph is

$$\mathcal{X}_w^\star = \arg\min_{\mathcal{X}_w} \; \left\|\mathbf{r}_0\right\|_{\Sigma_0}^2 + \sum_{\mathcal{Z}_i \in \{\mathcal{K}_w, \mathcal{F}_w\}} \left(\|\mathbf{r}_{I_i}\|_{\Sigma_{I_i}}^2 + \sum_{c_{ij} \in \mathcal{C}_i} \|\mathbf{r}_{c_{ij}}\|_{\Sigma_{c_{ij}}}^2 \right) \tag{3.9}$$

where $\mathbf{r}_0$ represent the prior residual, and the corresponding measurement covariances Σ_0, Σ_{I_i}, and $\Sigma_{c_{ij}}$. To solve the nonlinear SLAM problem, optimizers such as Dogleg and Levenberg-Marquart iterate on the linearized cost of (3.9) with respect to $\delta\mathcal{X}_w$. At iteration k, the linearized residual of IMU and camera measurements evaluate at the linearization point $\hat{\mathcal{X}}_w^{(k)}$ are in the forms:

$$\delta\mathcal{X}_w^{(k+1)} = \arg\min_{\delta\mathcal{X}_w} \left\|\mathbf{r}_0^{(k)} + \delta\mathcal{X}_0\right\|_{\Sigma_0}^2 + \sum_{\mathcal{Z}_i\in\{\mathcal{K}_w,\mathcal{F}_w\}} \left(\|\mathbf{r}_{I_i}^{(k)} + H_{I_i}^{(k)}\delta\mathcal{X}_w\|_{\Sigma_{I_i}}^2 + \sum_{c_{ij}\in\mathcal{C}_i} \|\mathbf{r}_{c_{ij}}^{(k)} + H_{\mathcal{C}_{il}}^{(k)}\delta\mathcal{X}_w\|_{\Sigma_{c_{ij}}}^2\right) \tag{3.10}$$

where

$$\mathbf{r_i}^{(k)} = h_i(\mathcal{X}^{(k)}) - z_i \tag{3.11}$$

is the estimation error during k-th iteration, and

$$H_{I_i}^{(k)} = \left.\frac{\partial \mathbf{r}_{I_i}}{\partial \mathcal{X}_\omega}\right|_{\mathcal{X}_\omega = \hat{\mathcal{X}}_\omega^{(k)}}, \quad H_{c_{ij}}^{(k)} = \left.\frac{\partial \mathbf{r}_{c_{ij}}}{\partial \mathcal{X}_\omega}\right|_{\mathcal{X}_\omega = \hat{\mathcal{X}}_\omega^{(k)}} \tag{3.12}$$

are the IMU and camera measurement Jacobians. The optimizer solves for $\delta\mathcal{X}_w^{(k+1)}$ and updates the window iteratively:

$$\hat{\mathcal{X}}_w^{(k+1)} = \hat{\mathcal{X}}_w^{(k)} \oplus \delta\mathcal{X}_w^{(k+1)} \tag{3.13}$$

3.4 Optimization on Manifold

In (3.13), we use the notation $\oplus$. The $\oplus$ operator follows vector addition in $\mathbb{R}^n$ but it has a different meaning if the state variable is on a different space. In this thesis, we focus on matrix Lie group since it captures the underlying representation of 3D rotations. In matrix Lie group, the $\oplus$ operator indicates matrix multiplication such as $SO(3)$ for rotation and $SE(3)$ for poses.

3.4.1 Problem with Rotation

In general, we assume the state vector lies on a some n-dimensional Euclidean space $\mathbb{R}^n$, and therefore the optimization is performed on the space. However, when dealing with 3D rotation, it is more complicated because a rotation doesn't lie on the Euclidean space. Even though a rotation matrix is $3\times3 \in \mathbb{R}^{3\times3}$, but it does not capture nine degrees of freedom (DOF). In fact, a 3D rotation matrix only captures three DOF. Similar in the 2D case, a 2×2 rotation matrix only captures 1 DOF. Therefore, mathematically we say a 3D rotation matrix lies on a 3-dimensional *manifold* commonly known as the Special Orthogonal Group ($SO(3)$):

$$SO(3) = \{\mathbf{R} \in \mathbb{R}^{3\times3} | \mathbf{R}^\top \mathbf{R} = \mathbf{I}, \det(\mathbf{R}) = 1\} \tag{3.14}$$

The $SO(3)$ manifold is the collection of all 3×3 rotation matrix, and its operations are different than those used in the Euclidean space. For example, we cannot "subtract" two rotation as $R_1 -$

$R_2 \notin SO(3)$ and most importantly, rotations are not communitive $R_1R_2 \neq R_2R_1$. Besides $SO(3)$, another group commonly used in robotics [2, 13] is the Special Euclidean Group $SE(3)$, which represent a 4×4 transformation matrix (pose). Without using vector operations, we can, however, perform operations using *matrix Lie group*, which is a set of matrices that lie on a *differential manifold* that has smooth operations [2, 13]. Smooth roughly means a small change in input to a group operation will cause a small change in the output.

Optimization with rotations is tricky as shown in Fig. 3.7. Suppose the surface is the $SO(3)$ space, and the red line follows the true optimization path. If we do not obey the group law and arbitrarily optimize a 3×3 matrix, the solutions will not guarantee to be valid rotation matrices. That is, the solution leaves the manifold surface and no longer satisfies some properties of the rotation matrix (such as orthonormal) shown in the blue dotted line. In traditional methods of optimizing rotation matrix, a projection step is required to bring the blue solution back to the manifold. However, this requires an extra step and exhibits bad convergence properties. Since rotations belong to matrix Lie group, we will derive proper optimization using tools from differential geometry to ensure the optimization follows the manifold path.

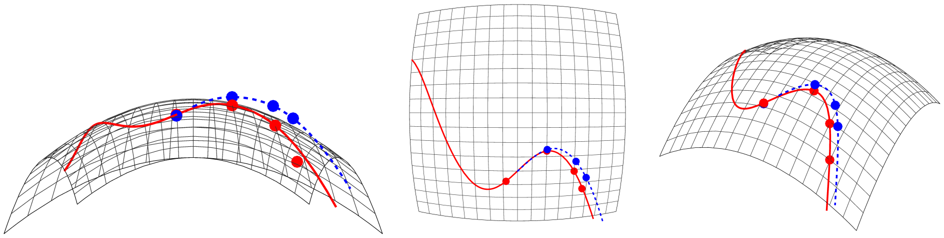


Figure 3.7: When optimization with rotations, it is important to ensure the resultant matrix is a rotation matrix so the solution lies on the manifold (red). However, we we arbitrarily optimize a 3×3 matrix, the solution will not guarantee to be a rotation matrix (blue) and might be an invalid rotation matrix.

3.4.2 Matrix Lie Group - Lie algebra Correspondences

According to [2], a group G is a set of elements that follows a group law (eg. Matrix multiplication), and also satisfy four group axioms, namely closure, associativity, identity, and invertibility. An *Lie* Group is a locally flat differential manifold, and it possesses the *symmetric* property. For example, an earth is locally flat but it is a sphere, and a small rotation of a sphere looks like itself. The most important concept of a Lie group, being locally differentiable, is the correspondence to its *Lie algebra* $\mathfrak{g}$, which is the tangent space at the identity element (in n-dimensional matrix Lie group, the identity is the $n \times n$ identity matrix).

The idea of Lie group-Lie algebra correspondence is a powerful one illustrated in Fig. 3.8

Visually, a Lie group can be seen as the folding of a flat surface, and every coordinate on the manifold (orange) corresponds to exactly one point on the flat tangent space (blue). The origin of the folding (tangent point) is the "identity" element of the manifold. This idea entails that whenever

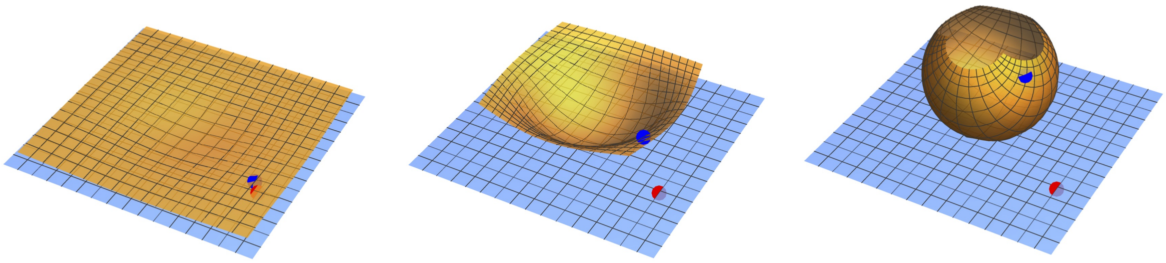


Figure 3.8: This figure shows the Lie group-Lie algebra correspondences. Every point on the orange manifold has a correspondence to a point on its tangent space $\mathcal{T}_G$. The mapping from a manifold element to a tangent element is through the exponential map.

we perform optimization on a Lie group, we can perform the same optimization using the linear tangent space, and therefore convert a nonlinear problem into a linear one. Moreover, a point on the tangent space $\mathfrak{g}$ is a linear combination of the basis called the *generators*, and the coefficient of the combination is called the *chart*. For example in $SO(3)$, we can then represent any point on the tangent space $so(3)$ using the chart ω:

$$\begin{aligned} \omega \in \mathbb{R}^3 \\ \omega_1 G_1 + \omega_2 G_2 + \omega_3 G_3 \in so(3) \end{aligned} \tag{3.15}$$

Therefore, the chart represents the minimum representation of the space (eg. $\mathbb{R}^3$ for 3D rotation), and is the state space we will perform optimization on for rotation.

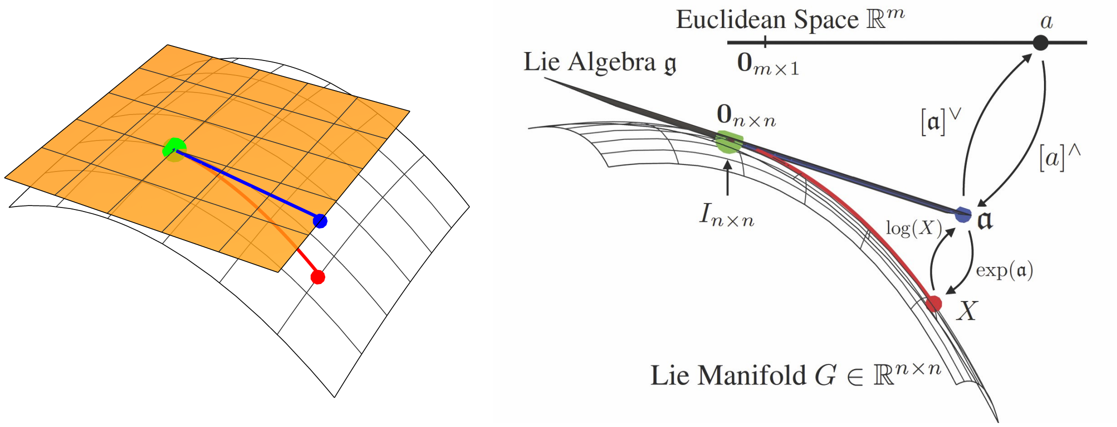


Figure 3.9: Illustration of matrix exponential and matrix logarithm.

The mathematical conversions between chart, Lie algebra, and Lie group shown in Fig. 3.9, where

$$\begin{aligned} [\cdot]^\vee &: \mathfrak{g} \to \mathbb{R}^m \\ [\cdot]^\wedge &: \mathbb{R}^m \to \mathfrak{g} \\ \exp(\mathfrak{a}) &: \mathfrak{g} \to G \\ \log(a) &: G \to \mathfrak{g} \end{aligned} \tag{3.16}$$

where we called $\exp(\cdot)$ the exponential map, which takes an element of the tangent space to the corresponding manifold space; $\log(\cdot)$ the logrithm map, which is the inverse of the exponential map. The "hat" $[\cdot]^\wedge$ operator takes an element of the tangent space to its chart, and the inverse for "ve" operator. For mathematical derivation of these operators, readers are referred to [13, 21].

3.4.3 3D Rotation - Special Orthogonal Group SO(3)

For $SO(3)$ specifically used in our work and by [29], the closed-form for each operation is given by the closed-form [21]:

$$[\omega]^\wedge = \begin{bmatrix}\omega_1\\ \omega_2\\ \omega_3\end{bmatrix} = \begin{bmatrix}0 & -\omega_3 & \omega_2\\ \omega_3 & 0 & -\omega_1\\ -\omega_2 & \omega_1 & 0\end{bmatrix} \in so(3) \tag{3.17}$$

$$\exp([\phi]^\wedge) = \mathbf{I} + \frac{\sin(\|\phi\|)}{[\|\phi\|}[\phi]^\wedge + \frac{1-\cos(\|\phi\|)}{[\|\phi\|^2}([\phi]^\wedge)^2 \tag{3.18}$$

$$\log(R) = \frac{A\cdot(R-R^\top)}{2\sin(A)},\quad A = \cos^{-1}(\frac{\mathrm{Tr}(R)-1}{2}) \tag{3.19}$$

If we use small angle approximation to (3.18), we get

$$\exp([\phi]^\wedge) \approx \mathbf{I} + [\phi]^\wedge \tag{3.20}$$

In general, $\exp([(a+b)]^\wedge) \neq \exp([a]^\wedge)\exp([b]^\wedge)$, but the Baker-Campbell-Hausdorff (BCH) formula can be used [2]. If b is small enough, however, we can use the first-approximation suggested by [29]. Specifically, the first-order approximation can be made to these operations using the idea of *right Jacobian* J_r of exponential map [13]:

$$J_r(\phi) = \mathbf{I} - \frac{1-\cos(\|\phi\|)}{[\|\phi\|^2}[\phi]^\wedge + \frac{\|\phi\| - \sin(\|\phi\|)}{[\|\phi\|^3}([\phi]^\wedge)^2 \tag{3.21}$$

The idea of J_r relates small perturbation in the tangent space to the right multiplicative increments shown in Fig. 3.10 As shown, the first-order approximation for exponential map is

$$\exp([\phi + \delta\phi]^\wedge) \approx \exp([\phi]^\wedge)\exp([J_r(\phi)\delta\phi]^\wedge) \tag{3.22}$$

Likewise, we can utilize the Right Jacobian to perform the first-order approximation on the log map as

$$\log(\exp([\phi]^\wedge)\exp([\delta\phi]^\wedge))^\vee \approx \phi + J_r^{-1}(\phi)\delta\phi \tag{3.23}$$

where

$$J_r^{-1}(\phi) = \mathbf{I} + \frac{1}{2}[\phi]^\wedge + \Big(\frac{1}{\|\phi\|^2} + \frac{1+\cos(\|\phi\|)}{2\|\phi\|\sin(\|\phi\|)}\Big)([\phi]^\wedge)^2 \tag{3.24}$$

Using first-order approximation, our IMU preintegration formulation in Section 5 and the works proposed by [29] is able to solve the mathematical equations in real-time.

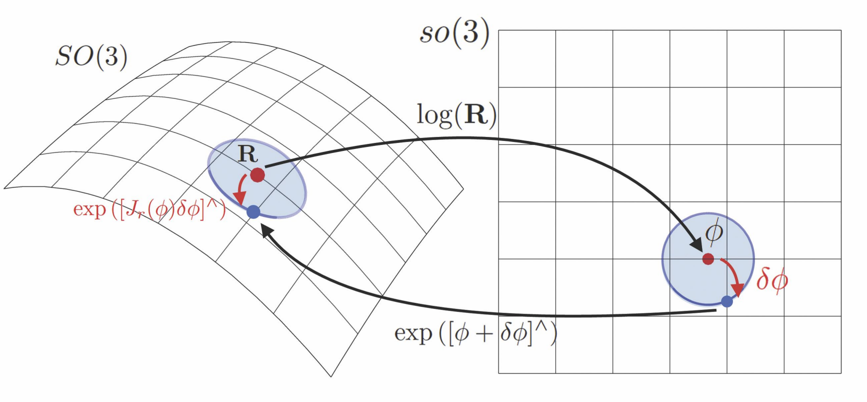
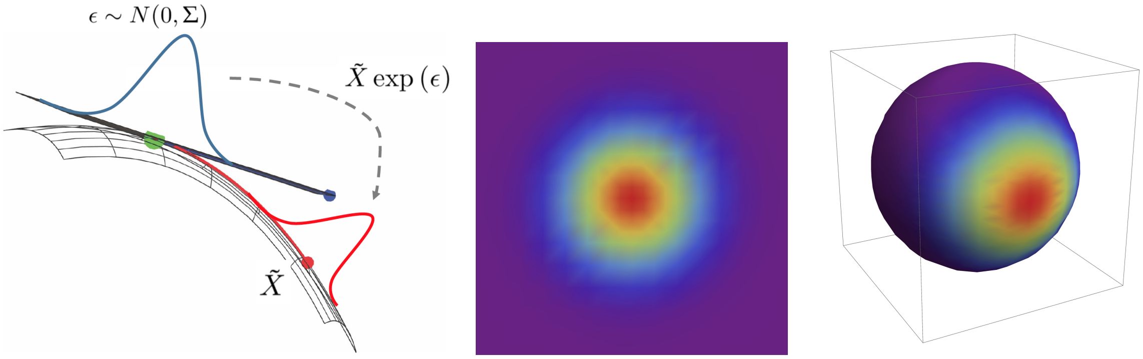

$SO(3)$
$so(3)$
$\log(\mathbf{R})$
$\mathbf{R}$
$\exp\left([J_r(\phi)\delta\phi]^\wedge\right)$
ϕ
$\delta\phi$
$\exp\left([\phi + \delta\phi]^\wedge\right)$

Figure 3.10: The Right Jacobian of the exponential map relates small increment in tangent space to multiplicative increment on the manifold.

3.4.4 Uncertainty on SO(3) and Concentrated Gaussian Distribution

Using differential geometry tools introduced in Section 3.4.2, a Gaussian distribution can be defined directly on the manifold as shown in Fig. 3.11. This allows the state estimation problem to be formulated directly using Lie group, which will be important for the invariant VIO framework presented in Section 5.



(a) (b) (c)

Figure 3.11: We can use the manifold tools to define a Gaussian distribution with mean $\tilde{\mathcal{X}}$ on the manifold, and the variance ϵ on the tangent space **(a)**. This distribution is called the *Concentrated Gaussian Distribution* (CGD) [2]. **(b)** and **(c)** show an example of CGD on the tangent plane and the manifold respectively.

Specifically, in matrix Lie group, there are both *left* multiplication and *right* multiplication depending on whether the noise is multiplied through the left or right of the group element:

$$\text{Left Multiplication}: \hat{X}_L = \tilde{X}\exp(\epsilon), \quad \text{Right Multiplication}: \hat{X}_R = \exp(\epsilon)\tilde{X} \tag{3.25}$$

The two formulations are slightly different and affects the mathematical derivations. Interested readers are referred to [2] or [13] for details. In our formulation in Section 5, we use the left-multiplication framework, which follows [29] but not [2].

Chapter 4

Information Sparsification in Visual Inertial State Estimation

In Section 3.2.2 and Section 3.3, VIO factor graph structures and the concept of a fixed-lag smoother are introduced (Fig. 4.4a). Currently, there are several known drawbacks in a fixed-lag framework (FEJ-FL) [36].

1. In order to bound computational complexity, a fixed-lag smoother marginalizes out variables, which requires linearizing the system by fixing linearization points. As a consequence, it no longer describes the original nonlinear optimization.
2. Marginalization limits the ability for the fixed-lag smoother to converge to the optimal solution in future optimizations because marginalized variables are no longer optimizable.
3. Consecutive marginalizations create a densely connected prior in the graphical model, which significantly decreases computational efficiency.

To address these problems, in this section we detail our first main contribution: a novel information-theoretic approach for a fixed-lag VIO system by online sparsification. The proposed method maintains the original nonlinear VIO optimization while preserves most of the information and its sparsity.

4.1 Problem Formulation

To bound computational complexity, a fixed-lag smoother marginalizes out selected states to maintain a fixed-size optimization window. Marginalization on the Gaussian distribution is typically done by Schur complement on the linearized information matrix Λ, which corresponds to the inverse covariance, $\Lambda = \Sigma^{-1}$.

4.1.1 Information Matrix and Covariance Matrix

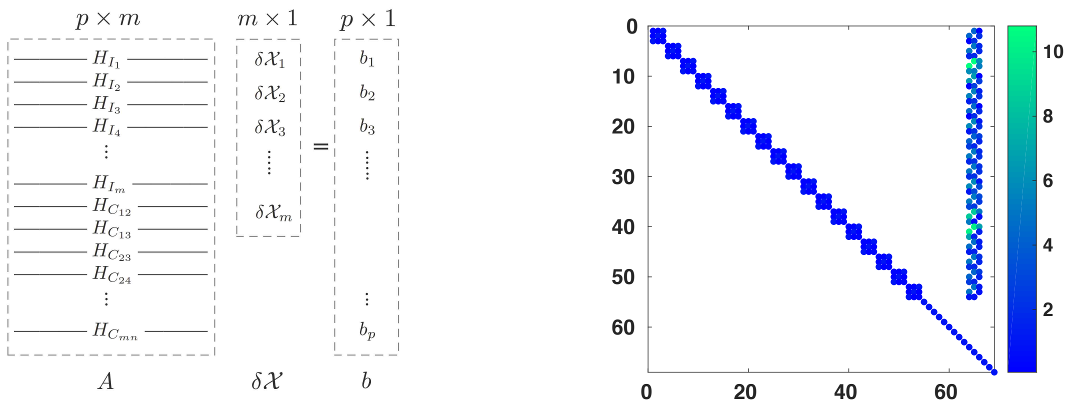
Remember when performing nonlinear optimization, at each iteration we linearized the VIO residuals from (3.9) to below:

$$\delta\mathcal{X}_w^{(k+1)} = \arg\min_{\delta\mathcal{X}_w} \;\left\|\mathbf{r}_0^{(k)} + \delta\mathcal{X}_0\right\|_{\Sigma_0}^2 + \textstyle\sum_{\mathcal{Z}_i\in\{\mathcal{K}_w,\mathcal{F}_w\}} \left(\|\mathbf{r}_{I_i}^{(k)} + H_{I_i}^{(k)}\delta\mathcal{X}_w\|_{\Sigma_{I_i}}^2 + \textstyle\sum_{c_{ij}\in\mathcal{C}_i} \|\mathbf{r}_{c_{ij}}^{(k)} + H_{\mathcal{C}_{il}}^{(k)}\delta\mathcal{X}_w\|_{\Sigma_{c_{ij}}}^2\right) \tag{4.1}$$

Using the definition of the Mahalanobis distance from (3.3) and the symmetric property of measurement covariance matrix, each residual $\|\mathbf{r}^{(k)} + H\delta\mathcal{X}\|_\Sigma^2$ can be simplified to least-squares. This is called the *statistical whitening* step, which corresponds to normalization of multivariate Gaussian.

$$\begin{aligned}
\left\|\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i\right\|_{\Sigma_i}^2 &= (\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i)^\top \Sigma_i^{-1} (\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i) \\
&= (\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i)^\top \Sigma_i^{-\frac{1}{2}} \Sigma_i^{-\frac{1}{2}} (\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i) \\
&= \left(\Sigma_i^{-\frac{1}{2}} (\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i)\right)^\top \left(\Sigma_i^{-\frac{1}{2}} (\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i)\right) \\
&= \|\Sigma_i^{-\frac{1}{2}} (\mathbf{r}_i^{(k)} + H_i\delta\mathcal{X}_i)\|^2 \\
&= \|b_i + A_i\delta\mathcal{X}\|^2
\end{aligned} \tag{4.2}$$

We can then stack each linearized residual from (4.2) into a concatinated form of $\|A\delta\mathcal{X}_w + b\|^2$ shown in Fig. 4.1. Because of the Markov assumption, the stacked Jacobian matrix (and therefore the $A^\top A$ matrix) is inherently *sparse*. Tools such as sparse Cholesky, QR factorization can be utilized to solve the normal equation $(A^\top A)^{-1}\mathcal{X}_w^{(k+1)} = A^\top b$ until convergence. The converged estimates represents a high dimensional multivariate Gaussian distribution. The first moment is the estimates itself, and the second moment is the inverse of linearized $A^\top A$ matrix.



(a)

(b)

Figure 4.1: This figure shows we could concatinate (4.1) into an $A\delta\mathcal{X}_w = b$ form. **(b)** is a real example of a Jacobian matrix of a VIO factor graph.

The $A^\top A$ matrix is called the (Fisher) *Information Matrix*. To see the connection between a Fisher Information matrix and a covariance matrix, we look at factor graph optimization as solving the maximum a posteriori. From statistical theories, the Fisher Information is the second derivative of the log-likelihood with respect to the states, which tells us the curvature of the estimation. The higher the information, the “sharper” the estimation is, which makes the solution more certain. Therefore, the information matrix represents the inverse of covariance matrix.

4.1.2 Marginalization and Sparsity

In a multivariate Gaussian, a marginal distribution of a subset of random variables is the probability distribution contained in this subset of variables. This is typically done by summing or integrating the joint probability distribution over the marginalized variables. The covariance matrix already specifies the marginals of each variable in the diagonal block.

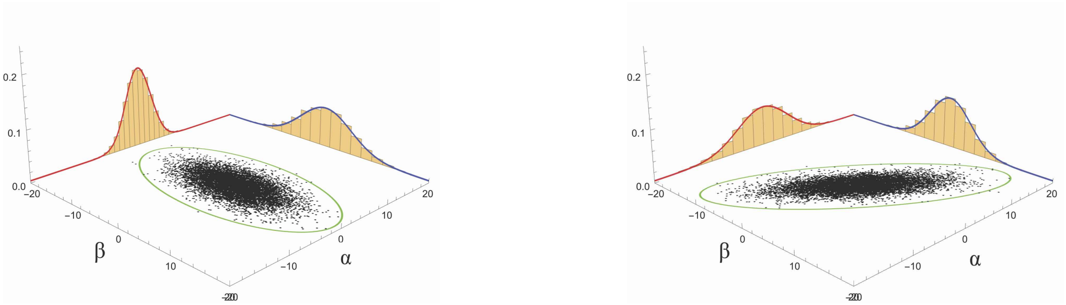
As an example, suppose there is a bivariate Gaussian with the joint mean $\mu_{\alpha,\beta}$ and joint covariance matrix $\Sigma_{\alpha,\beta}$ we can represent the joint distribution as

$$\begin{aligned}&\mu_{\alpha,\beta} = \begin{bmatrix}\mu_\alpha \\ \mu_\beta\end{bmatrix}, \quad \Sigma_{\alpha,\beta} = \begin{bmatrix}\Sigma_{\alpha\alpha} & \Sigma_{\alpha\beta} \\ \Sigma_{\beta\alpha} & \Sigma_{\beta\beta}\end{bmatrix} \\ &\alpha, \beta \sim \mathcal{N}(\mu_{\alpha,\beta}, \Sigma_{\alpha,\beta})\end{aligned} \tag{4.3}$$

Suppose now we would like to marginalize α, the marginal distribution is simply $\beta \sim \mathcal{N}(\mu_\beta, \Sigma_{\beta\beta})$. However, from Section 4.1, the optimization maintains the sparse information matrix $\Lambda = \Sigma^{-1}$. Instead of extracting the main diagonal matrix to the corresponding variable, we perform the *Schur Complement* with the following formula:

$$\begin{aligned}\Sigma_{\alpha,\beta} &= \begin{bmatrix}\Sigma_{\alpha\alpha} & \Sigma_{\alpha\beta} \\ \Sigma_{\beta\alpha} & \Sigma_{\beta\beta}\end{bmatrix} = \begin{bmatrix}\Lambda_{\alpha\alpha} & \Lambda_{\alpha\beta} \\ \Lambda_{\beta\alpha} & \Lambda_{\beta\beta}\end{bmatrix}^{-1} \\ \Sigma_{\beta\beta} &= \Lambda'^{-1}_{\beta\beta} = (\Lambda_{\beta\beta} - \Lambda_{\beta\alpha}\Lambda^{-1}_{\alpha\alpha}\Lambda_{\alpha\beta})^{-1}\end{aligned} \tag{4.4}$$

Marginalization reduces the dimensionality of the problem. Visually in Fig. 4.2, we can treat



(a) Uncorrelated α and β

(b) Correlated α and β

Figure 4.2: This figure shows a bivariate Gaussian distribution and the marginal distribution in each axis. In **(a)**, the two random variables α and β are uncorrelated, meaning that the distribution of one does not depend on the other, as opposed to the correlated case in **(b)**.

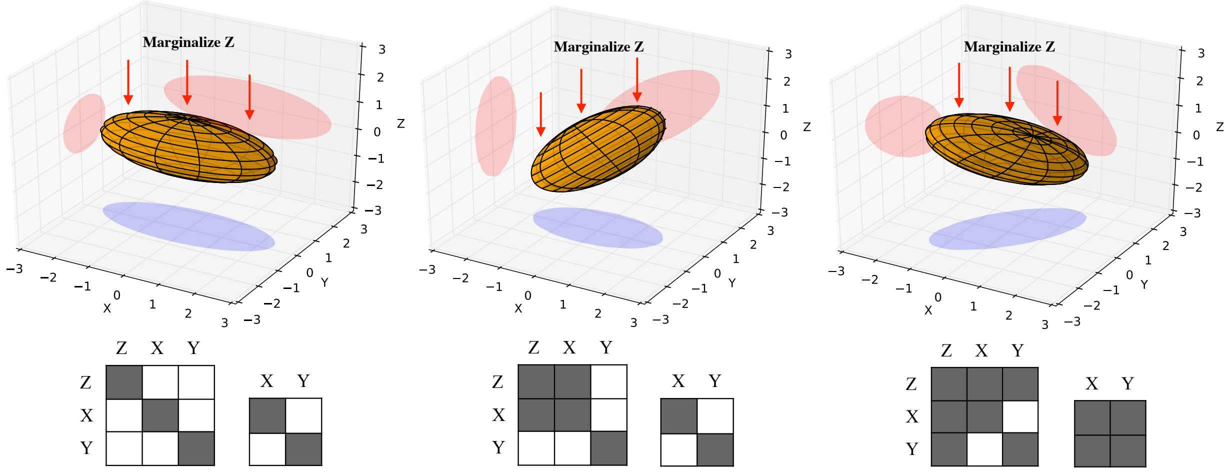
marginalization as a mathematical projection. The dimensionality is reduced by projecting the distribution to the axis of remaining variables. In our example, the distribution is project to the β axis. In fact, the equality between shur-complement marginalization and null-space projection in SLAM is presented in [71].

In a high dimensional Gaussian, the correlation between variables (off diagonal blocks of the information matrix) affects the correlation of the resultant distributions shown in Fig. 4.3. In this 3-variable example, we look at three different cases, with their corresponding information matrix at the bottom of each graph. The marginalization is perform on Z, so we look at the blue resultant distribution in the $X - Y$ plane. In Fig. 4.3a and Fig. 4.3b, because there is no correlation term in the $Z - Y$ block, the resultant distribution shows X and Y are independent. However, in Fig. 4.3c, even if originally there is no correlation between X and Y, but because the $Z - Y$ cross term, the resultant variable is dense.

It means that the cross term relates the entries of marginalized variables into the resultant matrix and transfers the information. Mathematically from (4.4), the cross term affects the sparsity of the resultant information $\Lambda'_{\beta\beta}$. This example explains two very important concepts about marginalization in a factor graph:

1. While the dimensionality is reduced through marginalization, the the resultant information matrix will no longer be sparse. In a factor graph, it means the resultant factor will be connecting to more variables defined by the cross terms.

2. Only variables that are correlated to the marginalized variables get effected by marginalization.



(a) X, Y, Z are independent

(b) $X - Z$ are dependent

(c) $X - Z, Y - Z$ are dependent

Figure 4.3: This figure illustrate the effect of marginalization to the sparsity of the resultant information, which is largely determined by the off-diagonal blocks. In this case, we are marginalizing Z, and the resultant blue $X - Y$ distribution is either correlated or not, depending on the $Z - Y$ off-diagonal term. This tells us that marginalization can potentially result a dense information matrix.

4.1.3 Marginalization in Fixed-Lag VIO

In a VIO factor graph such as Fig. 4.4a, the information matrix is much larger. The cross terms relating the marginalized variables to the remaining variables only has non-zero terms defining by the measurement Jacobians. These are variables nodes that are incident to the marginalized variables. Therefore, when performing marginalization, we only need to consider the specific set of variables called the Markov blanket ($\mathcal{X}_{\text{(MB)}}$). We denote the Markov Blanket information matrix as $\Lambda_{\text{(MB)}}$. In Fig. 4.5a, the red variables and factors show an example of the Markov blanket with respect to the marginalized robot states of keyframe K_{k-m}.

$\Lambda_{\text{(MB)}}$ is constructed by the measurement Jacobian in the Markov blanket:

$$\Lambda_{\text{(MB)}} = H_0^\top \Sigma_0^{-1} H_0^\top + H_{I_i}^\top \Sigma_{I_i}^{-1} H_{I_i} + \sum_{c_{ij} \in \mathcal{C}_i} H_{c_{ij}}^\top \Sigma_{c_{ij}}^{-1} H_{c_{ij}} \tag{4.5}$$

Note that $\Lambda_{\text{(MB)}}$ is sparse and its entries correspond to the connectivity in the graph. Define $\mathcal{X}_R \in \mathcal{X}_{\text{(MB)}}$ the remaining states, and $\mathcal{X}_M \in \mathcal{X}_{\text{(MB)}}$ the marginalized states, we can perform Schur-Complements on $\mathcal{X}_M$:

$$\begin{aligned} \Lambda_{\text{(MB)}} &= \begin{bmatrix} \Lambda_{\mathcal{X}_R \mathcal{X}_R} & \Lambda_{\mathcal{X}_M \mathcal{X}_R} \\ \Lambda_{\mathcal{X}_R \mathcal{X}_M} & \Lambda_{\mathcal{X}_M \mathcal{X}_M} \end{bmatrix} \\ \Lambda_{\text{t}} &= \Lambda_{\mathcal{X}_R \mathcal{X}_R} - \Lambda_{\mathcal{X}_R \mathcal{X}_M} \Lambda_{\mathcal{X}_M \mathcal{X}_M}^{-1} \Lambda_{\mathcal{X}_M \mathcal{X}_R} \end{aligned} \tag{4.6}$$

where Λ_{t} is the *target information* corresponding to the dense resultant matrix (prior). Marginalization degrades the algorithm efficiency as the factor graph loses its sparse structure. To cope with such issue, keyframe-based VIO methods such as OKVIS [48] and VINS-MONO [59] selectively discard measurements to maintain sparsity during marginalization. For landmarks that are not observed by the recent frames, they are marginalized altogether with the marginalized robot states. It is important to note that such marginalize strategies while maintaining efficiency, potentially lose the capabilities re-estimating the positions of the landmarks and therefore become less accurate. This motivates the main contributions of our work in minimizing information loss during marginalization.

4.2 Sparsification in VIO

4.2.1 Marginalization Strategy

Shown in Fig. 4.4a, the proposed method maintains n recent frames, and m keyframes. When a new frame F_i enters the window, we check whether frame F_{i-n} is a keyframe to select the following *midframe marginalization* or *keyframe marginalization* strategy. In order to enforce consistency, we adopt the First-Jacobian Estimate method from Dong-Si et al. [20] by using the prior linearization points when corresponding measurement Jacobians are first evaluated.

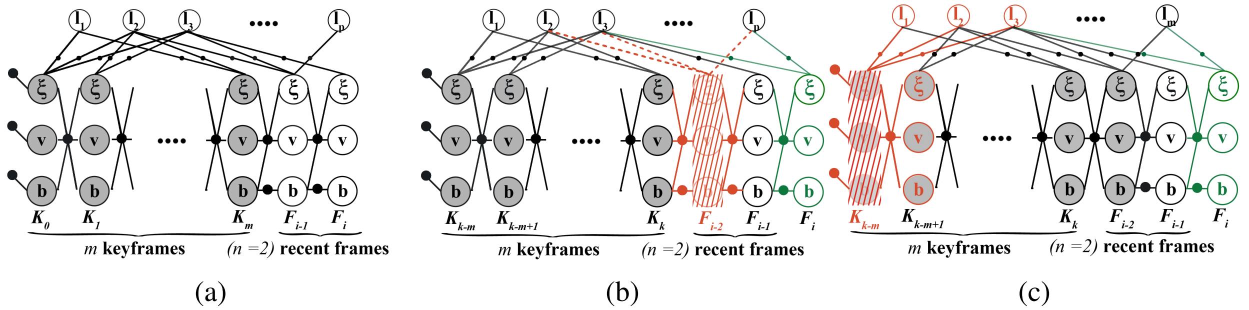


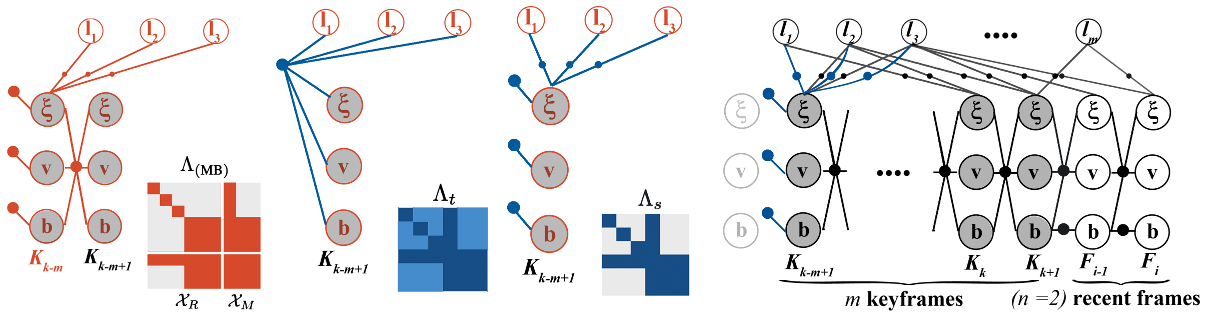
Figure 4.4: **(a)**: The proposed fixed-lag VIO factor graph. Each camera frame corresponds to a robot state is $x = [\xi^\top, v^\top, b^\top]^\top$, where $\xi \in \mathbb{R}^6$ is the minimum representation of the robot pose. $v \in \mathbb{R}^3$ is the velocity. $b \in \mathbb{R}^6$ is the IMU bias. The l's are the visual landmark variables. Measurement factors are represented by solid black circles, including prior factors, IMU factors, and stereo projection factors. **(b)**: Suppose we define the recent frame window size $n = 2$. When a new frame F_i arrives (green), the proposed algorithm looks at frame F_{i-2} being a keyframe (shaded circle), or a regular frame (transparent circle). In this case, F_{i-2} is not a keyframe, the proposed method employs *midframe marginalization* including only the IMU constraints while discarding all visual measurements. **(c)**: If F_{i-2} frame is a keyframe, the proposed method employs *keyframe marginalization* with sparsification. It includes all variables and measurement information in the Markov blanket (in red).

Midframe Marginalization

Fig. 4.4b shows an example of the midframe marginalization strategy. Follow both OKVIS and VINS-MONO on midframe marginalization, we discard all projection factors but only include inertial constraints. This is to keep sparsity but also avoid repeated observations on the landmarks when the robot is stationary. The resulting factor is a *marginalized* IMU measurement that connects to the two corresponding robot states.

Keyframe Marginalization

Fig. 4.4c shows an example of the keyframe marginalization strategy. If frame F_{i-n} is a keyframe, we perform marginalization on the oldest keyframe at K_{k-m} and landmarks that are only connected to the frame. Unlike existing methods, the rest of the landmarks are preserved during the marginalization step, so that they remain in the optimization window for further nonlinear updates. The result is a dense prior connecting to the next state and all the landmarks defined by the Markov blanket. The blue prior factor in Fig. 4.5b shows an example connectivity of this prior. In the real system, the associated information Λ_t can be large as shown in Fig. 4.7a, which significantly reduces computational efficiency. However, keeping landmarks as variables in the optimization window allows further nonlinear updates for subsequent optimizations to reach the optimal solution. To reintroduce sparsity to the graph, our method applies sparsification to the dense prior information.



(a) (b) (c) (d)

Figure 4.5: (**a**): The proposed method first calculates the Markov blanket information $\Lambda_{(\mathrm{MB})}$ from the oldest keyframe. (**b**): The new target information is then calculated by methods such as Schur-Complement. The resultant matrix corresponds to a dense prior factor that connects to every variables in the Markov blanket. (**c**): Given Λ_{t}, we employ sparsification with the designed nonlinear factor topology, which we will recover the corresponding information Λ_{r} for each measurement. (**d**): The proposed method re-insert the sparsified topology back to the original fixed-lag window, which retains sparsity and structural similarity.

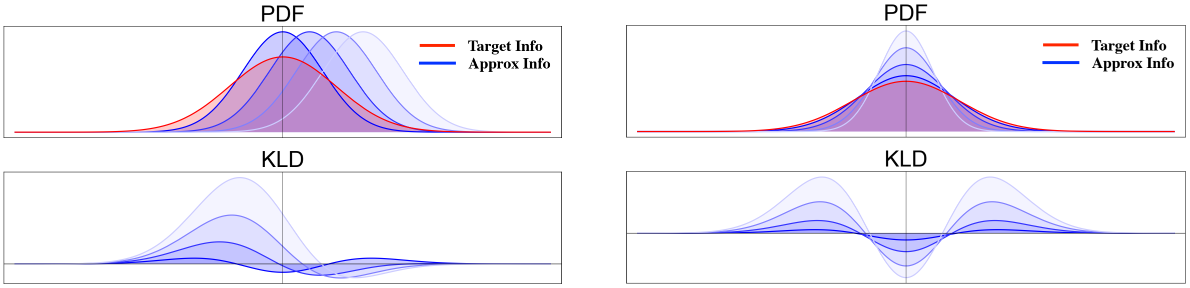
4.2.2 Information Sparsification

The dense prior information Λ_{t} defines a multivariate Gaussian $p(\mathcal{X}_{\mathrm{t}}) \sim \mathcal{N}(\mu_{\mathrm{t}}, \Lambda_{\mathrm{t}})$, with the mean equals to the current linearization point $\mathcal{X}_{\mathrm{t}}$ of the Markov blanket. We use the global linearization point for the Markov blanket since global priors are included in the marginalization.

Our method first specifies a factor graph topology $\mathcal{T}$ for the Markov blanket, which induces a sparsified distribution $p_{\mathrm{s}}(\mathcal{X}_{\mathrm{t}}) \sim \mathcal{N}(\mu_{\mathrm{s}}, \Lambda_{\mathrm{s}})$. We follow NFR [51] to recover the approximate distribution such that the KLD from $p_{\mathrm{s}}(\mathcal{X}_{\mathrm{t}})$ to $p(\mathcal{X}_{\mathrm{t}})$ is minimized:

$$D_{KL}(p(\mathcal{X}_{\mathrm{t}}) \| p_{\mathrm{s}}(\mathcal{X}_{\mathrm{t}})) = \frac{1}{2}\Big(\langle \Lambda_{\mathrm{s}}, \Sigma_{\mathrm{t}} \rangle - \log\det(\Lambda_{\mathrm{s}}) + \|\Lambda_{\mathrm{s}}^{\frac{1}{2}}(\mu_{\mathrm{s}} - \mu_{\mathrm{t}})\|_2^2 - d\Big) \tag{4.7}$$

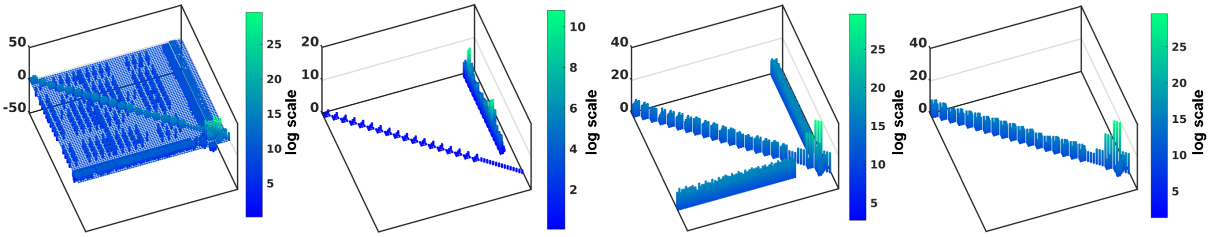
where $\Sigma_{\mathrm{t}} = \Lambda_{\mathrm{t}}^{-1}$.



(a) (b)

Figure 4.6: Illustration of KL-Divergence minimization in both mean ((**a**)) and covariance ((**b**)). The goal is to minimize the integral curve (KLD) to match the sparsified distribution in blue to the original one in red.

For each factor in $\mathcal{T}$, one must define the measurement z_s, the measurement model h_r and the measurement covariance $\Sigma_s = \Lambda_{\mathrm{s}}^{-1}$, such that $z_s = h_r(\mu_s) + v$, $v \sim \mathcal{N}(0, \Sigma_s)$. First, we set the measurements z_s of each factor to be the expected measurement considering the current state estimate $z_s = h_r(\mu_{\mathrm{t}})$ since it induces approximate distribution $\mu_s = \mu_t$ which minimizes (4.7). Next we look explain the definition of the measurement models. Section 4.2.3 details the method to recover Λ_{r} for every measurement.



(a) (b) (c) (d)

Figure 4.7: The diagram illustrates the sparsity of the corresponding matrix in pairs of images. **(a)** is the target information from the Markov blanket. **(b)** is the measurement Jacobian matrix corresponding to (4.10). **(c)** is the sparsified information corresponding to (4.11). **(d)** is the recovered measurement information corresponding to (4.10). In each image pair, the left image, the height in the 3D bar graph represent the magnitude of the log absolute value. The corresponding right image shows the informative entries above 10^{-5} threshold.

To design the topology, we consider that 1) Λ_{s} should best approximate Λ_{t} and 2) $\mathcal{T}$ maintains the sparsity of the graph for future optimizations and 3) $\mathcal{T}$ retains structural similarity to the original graphs. Because of the structure of the Markov blanket in our VIO formulation, the most informative entries of Λ_{t} (see Fig. 4.7a) are located at the main diagonal blocks and off-diagonal entries corresponding to robot state and landmarks. In Fig. 4.4 where we marginalize keyframe K_{k-m} in the example, the corresponding topology is shown in Fig. 4.5b. The topology consists of independent unary prior factors and binary relative measurement factors. The dense prior information always include the remaining robot state x_{R} corresponding to frame K_{k-m+1} and all the landmarks $\mathcal{L}_{\mathrm{R}} = \{\mathbf{l}_p \in \mathcal{X}_{\mathrm{t}}\}$.

Denote R_{R} the rotation, and $\mathbf{p}_{\mathrm{R}}$ the translation of the pose represented by ξ_{R}. We design two types of nonlinear topological measurement models to encapsulate the most informative entries in Λ_{t}. The first is the individual priors for the robot state x_{R},

$$h_{\mathrm{r}}(\xi_{\mathrm{R}}) = \xi_{\mathrm{R}}, \quad h_{\mathrm{r}}(\mathbf{v}_{\mathrm{R}}) = \mathbf{v}_{\mathrm{R}}, \quad h_{\mathrm{r}}(\mathbf{b}_{\mathrm{R}}) = \mathbf{b}_{\mathrm{R}} \tag{4.8}$$

and the second is the relative pose-to-landmark measurement model

$$h_{\mathrm{r}}(\xi_{\mathrm{R}}, \mathbf{l}_p) = R_{\mathrm{R}}^{-1}(\mathbf{l}_p - \mathbf{p}_{\mathrm{R}}), \quad \forall \mathbf{l}_p \in \mathcal{X}_{\mathrm{t}} \tag{4.9}$$

To construct the sparse information Λ_{s} of the topology using (4.8) and (4.9), we first define H_{r} and Λ_{r} as

$$H_{\mathrm{r}} = \begin{bmatrix} \vdots \\ H_{\mathrm{r}}^{(j)} \\ \vdots \end{bmatrix}, \quad \Lambda_{\mathrm{r}} = \begin{bmatrix} \ddots & & \mathbf{0} \\ & \Lambda_{\mathrm{r}}^{(j)} & \\ \mathbf{0} & & \ddots \end{bmatrix} \tag{4.10}$$

where $H_{\mathrm{r}}^{(j)}$ and $\Lambda_{\mathrm{r}}^{(j)}$ are the Jacobian and the unknown information matrix of the j-th nonlinear topological measurement model. An example of H_{r} is shown in Fig. 4.7b. Then Λ_{s} can be written as,

$$\Lambda_{\mathrm{s}} = H_{\mathrm{r}}^{\top} \Lambda_{\mathrm{r}} H_{\mathrm{r}} \tag{4.11}$$

The independent nonlinear topological measurements ensure Λ_{r} to be block-diagonal, which can be recovered as described in the following section.

4.2.3 Topology Measurement Covariances Recovery

With Λ_{t} and H_{r} provided, one can formulate a convex optimization based on KLD to recover the information Λ_{r} from (4.10) [51][22]:

$$\begin{aligned} \min \quad & C_{KL} = \langle H_{\mathrm{r}}^{\top} \Lambda_{\mathrm{r}} H_{\mathrm{r}}, \Sigma_{\mathrm{t}} \rangle - \log\det(H_{\mathrm{r}}^{\top} \Lambda_{\mathrm{r}} H_{\mathrm{r}}) \\ & \text{s.t.} \quad \Lambda_{\mathrm{r}} \succeq 0, \ \Lambda_{\mathrm{r}} \text{ is block diagonal} \end{aligned} \tag{4.12}$$

Typically this constrained optimization requires either Interior Point methods (IP) or limited-memory Projected Quasi-Newton (PQN) [61] and the recently proposed Factor Descent Algorithm [65]; however, PQN is only super linear convergence while IP method requires Hessian. Both methods are costly in terms of computational resources. Because our measurement model always provide a full-rank and invertible Jacobian matrix H_{r}, we are able to solve for (4.12) in closed-form:

$$\Lambda_{\mathrm{r}}^{(i)} = \left(\{H_{\mathrm{r}} \Sigma_{\mathrm{t}} H_{\mathrm{r}}^{\top}\}^{(i)}\right)^{-1} \tag{4.13}$$

where $(\cdot)^{(i)}$ denotes the i-th matrix block. The solution from (4.13) is unique and optimal by the convexity of (4.12). The proof is shown in [51] by examining the gradient of (4.12):

$$\begin{aligned} \frac{\partial C_{KL}}{\partial \Lambda_{\mathrm{r}}^{(i)}} &= \left\{H_{\mathrm{r}}[\Sigma_{\mathrm{t}} - (H_{\mathrm{r}}^{\top} \Lambda_{\mathrm{r}} H_{\mathrm{r}})^{-1})] H_{\mathrm{r}}^{T}\right\}^{(i)} \\ &= \left\{H_{\mathrm{r}} \Sigma_{\mathrm{t}} H_{\mathrm{r}}^{T} - H_{\mathrm{r}} H_{\mathrm{r}}^{-1} \Lambda_{\mathrm{r}}^{-1} H_{\mathrm{r}}^{-\top} H_{\mathrm{r}}^{T}\right\}^{(i)} \\ &= \left\{H_{\mathrm{r}} \Sigma_{\mathrm{t}} H_{\mathrm{r}}^{\top} - \Lambda_{\mathrm{r}}^{-1}\right\}^{(i)} \end{aligned} \tag{4.14}$$

Since (4.12) is an instance of MAXDET problem [66], the optimal solution is given by the sufficient and necessary condition for (4.14) to be $\mathbf{0}$.

Fig. 4.7c shows the recovered sparse information Λ_{s}, where each block corresponds to a nonlinear topological measurement. Our method then replaces the original dense prior Λ_{t} with sparsified topology $\mathcal{T}$ shown in Fig. 4.5d. The updated smoothing window from (3.9) now includes the sparsified measurement residuals $\mathbf{r}_{\mathrm{s}}$ with the corresponding covariance $\Sigma_{\mathrm{r}}^{(i)} = \Lambda_{\mathrm{r}}^{(i)-1}$.

4.3 Experimental Results

4.3.1 Implementation

We implement our method in a complete VIO pipeline that includes a visual frontend that matches stereo features, and a backend optimizer. The visual frontend implementation using OpenCV follows the typical pipeline of Shi-Tomashi Corner detector and KLT optical flow tracking for both temporal and stereo images. We implemented a Levenberg-Marquart optimizer and a factor graph based fixed-lag smoother using the GTSAM library [17]. All experiments are run on an Ubuntu desktop with Intel i7-6700 @3.40GHz CPU.

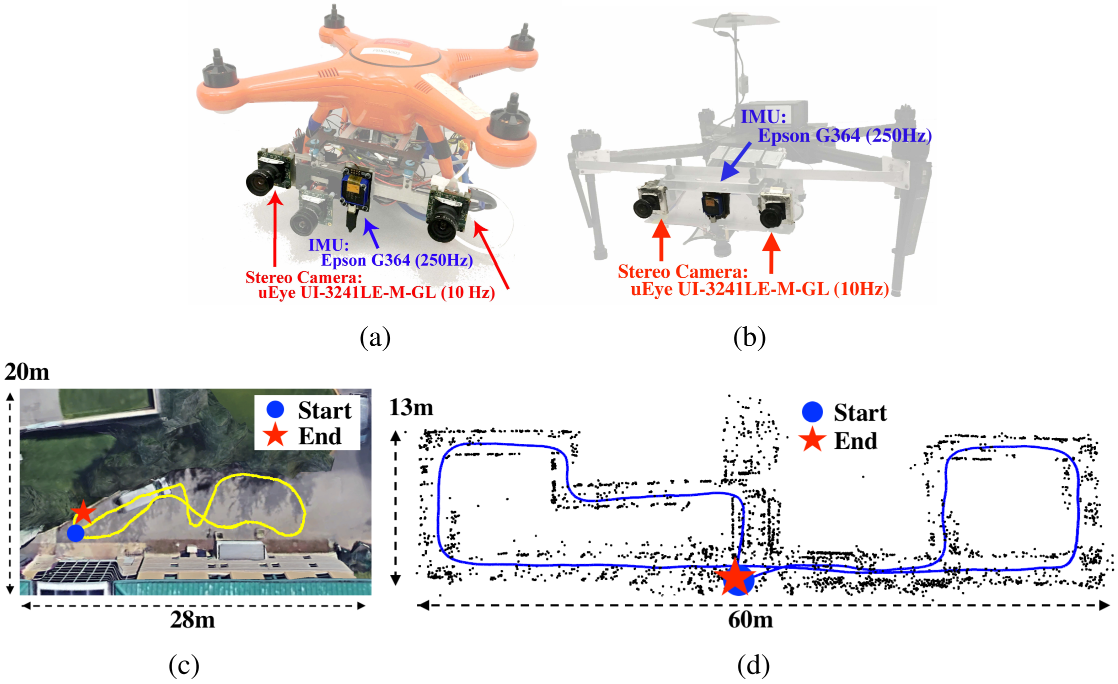


Figure 4.8: **(a)** Our custom built Autel X-Star Premium drone and DJI M100 **(b)** with the visual-inertial payload. It consists of two uEye UI-3241LE-M-GL cameras running at 10Hz and a synchronized Epson G364 IMU running at 250Hz. **(c)** An outdoor data sequence with the proposed algorithm running onboard. The corresponding stereo images are shown above the trajectory. The total distance is approximately 48m with the final drift 0.55m (1.11% error). **(d)** A walk through of 4th floor Newel Simon Hall of Carnegie Mellon University. The total distance is approximately 170m, with the final drift about 0.3m (0.17% error)

4.3.2 Real-time Hardware Test

We have demonstrated our algorithm running real-time onboard using a custom built Autel X-Star Premium drone as shown in Fig. 4.8a. The visual-inertial payload includes two uEye UI-3241LE-M-GL cameras recording at 10Hz and a synchronized Epson G364 IMU recording at

250Hz. Fig. 4.8d shows a trajectory walking through the Newell Simon Hall of Carnegie Mellon University. The total length of the sequence is ≈170 meter with the final position drift of 0.3 meter (0.17% error). We have also tested the proposed algorithm outdoor. One such example is shown in Fig. 4.8c. The total distance traveled is ≈48 meter with the final drift of 0.55 meter (1.11% error). For visualizations, we have included a link to a video showing the proposed algorithm running onboard with dynamic motions and trajectory plots.

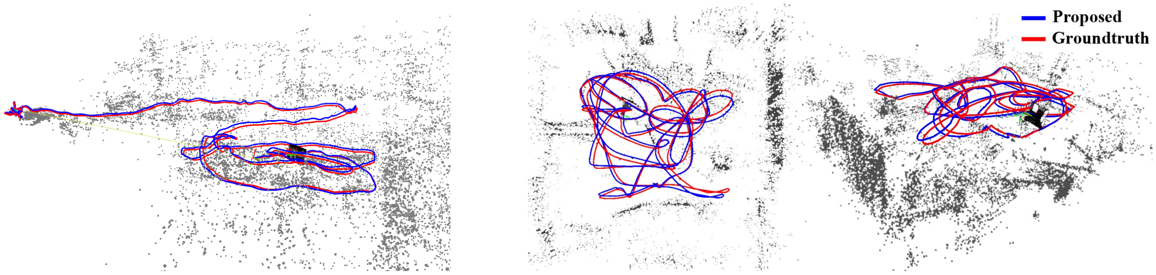


Figure 4.9: The figure shows the trajectories of the proposed method and the groundtruth on EuRoC Vicon Room 2 dataset [8]. The result shows the proposed algorithm achieves highly accurate state estimation in real-time.

4.3.3 Public Test Dataset

We evaluate the proposed method using the EuRoC visual-inertial dataset [8] by the metric of the Absolute Trajectory Error (ATE). ATE indicates the global consistency of the estimated trajectory by comparing the absolute distance to the ground truth.

$$ATE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|trans(X_{gt}^{i} - X_{est}^{i})\|^2}$$

The EuRoC dataset is recorded by a VI sensor with synchronized 20Hz stereo images and 200Hz IMU data. The dataset consists of three major sets of trajectories, Machine Hall (MH), Vicon Room 1 (V1), and Vicon Room 2 (V2), which vary in smooth and aggressive motions in large and small indoor environment. The V1 and V2 dataset present motion blur and lighting change that produce challenges to the state estimator.

We compare our method against stereo OKVIS [48] and monocular VINS-MONO [59], which are the state-of-the-art fixed-lag VIO systems. The loop-closure and online calibration of VINS-MONO functionality are deactivated to compare pure odometry performance. We have included ROVIO [7] to compare fixed-lag VIO approaches with a filtering-based approach. Lastly, to our best knowledge, we implemented OKVIS's marginalization strategy using our GTSAM framework. This is to directly compare the proposed algorithm with OKVIS's marginalization strategy by standardizing the frontend visual module, since OKVIS's frontend module utilizes its backend information for robustness. All results are generated offline in order to ensure the results are the comparison of pure accuracy.

Table 4.1: Root-Mean-Square ATE (meter) on the EuRoC Dataset

	MH					V1			V2	
	01_easy	02_easy	03_medium	04_difficult	05_difficult	01_easy	02_medium	03_difficult	01_easy	02_med.
Proposed	**0.059**	**0.060**	**0.099**	0.238	**0.187**	0.060	0.094	0.257	0.080	0.212
OKVIS	0.160	0.106	0.176	**0.208**	0.292	**0.050**	**0.061**	**0.127**	**0.055**	**0.081**
OKVIS (ours)	0.182	0.144	0.278	0.310	0.401	0.272	0.292	0.353	0.153	0.270
VINS-MONO	0.284	0.237	0.171	0.416	0.308	0.072	0.120	0.159	0.058	0.097
ROVIO	0.354	0.362	0.452	0.919	1.106	0.125	0.160	0.170	0.220	0.392

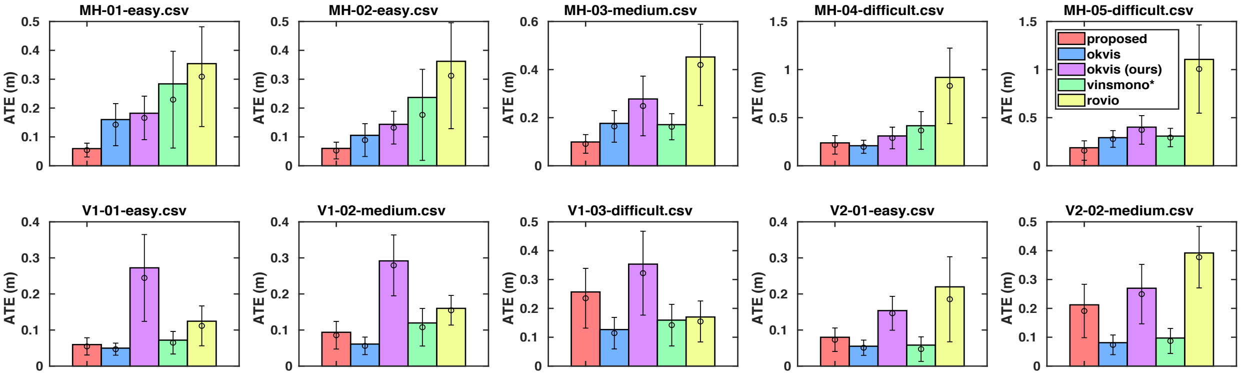


Figure 4.10: The diagram shows the comparison of the proposed method against other state-of-the-art algorithms on the EuRoC datasets [8]. Each color represents the result of an algorithm specified by the legend on the top right. The bar value specifies the Root-Mean-Square Error (RMS) on the Absolute Trajectory Error (ATE) matric in meter. Overlaying on each bar, there is an error bar that shows the mean and the standard deviation (Std) of the ATE.

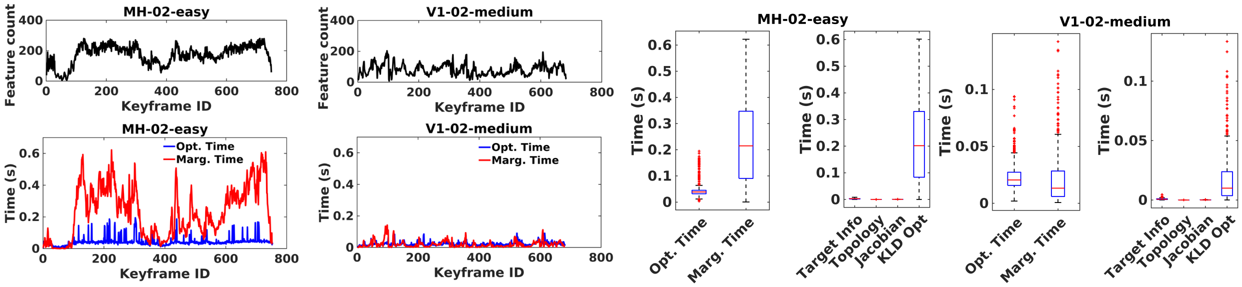
The ATE result is shown in Table. 4.1, and to better visualize we include a bar graph in Fig. 4.10. The result illustrates our proposed method outperforms the existing methods in four out of five trials in the MH dataset, and achieves comparable results in most of the V1 and V2 datasets. However, in V1_03_difficult and V2_02_medium datasets, the proposed method results in errors significantly larger than those from OKVIS and VINS-MONO. In both cases, the dynamic lighting change has caused the stereo camera images vary in grayscale. Consequently it significantly decreases the performance of our frontend matching algorithm. The sudden loss of visual information has caused a discrepancy in the state estimate. However, prior to the loss of features, our method out performs the existing method. It is important to note that both OKVIS and our proposed algorithm fail in running the V2_03_difficult dataset because of the motion blur. Both our frontend modules have failed to matched stereo features and the state estimate eventually diverges. One idea is to compute both sparse features and direct photometric odometry to handle the blurry images.

4.3.4 Run-time Analysis

To demonstrate that our algorithm is appropriate for real-time application, we conduct time profile on the proposed method on the EuRoC dataset with 300 feature cap. The result is shown in Fig. 4.11 and Table. 4.2. The statistics illustrate that the factor graph optimization maintains around 0.02 to 0.05 second and is independent of the difficulty of the dataset. This is expected as

Table 4.2: Run Time Analysis on the EuRoC Dataset

		Optimization			Marginalization		
		Mean (s)	RMSE (s)	Std (s)	Mean (s)	RMSE (s)	Std (s)
MH	01_easy	0.054	0.066	0.039	0.248	0.308	0.182
	02_easy	0.043	0.051	0.027	0.230	0.279	0.158
	03_medium	0.053	0.064	0.037	0.151	0.211	0.147
	04_difficult	0.034	0.042	0.024	0.088	0.129	0.094
	05_difficult	0.040	0.048	0.026	0.115	0.163	0.115
V1	01_easy	0.021	0.025	0.016	0.018	0.031	0.026
	02_medium	0.023	0.026	0.012	0.020	0.029	0.021
	03_difficult	0.021	0.026	0.015	0.015	0.034	0.031
V2	01_easy	0.027	0.033	0.018	0.039	0.062	0.049
	02_medium	0.016	0.017	0.006	0.009	0.013	0.010
	03_difficult	X	X	X	X	X	X



(a)

(b)

Figure 4.11: **(a)**: The execution time of proposed algorithm over keyframe ID. The optimization time is in blue, and the marginalization time is in red. **(Left)** MH_02_easy dataset. **(Right)** V1_02_medium dataset. **(b)**: **(Left pair)** The run-time analysis boxplot of the proposed algorithm on MH_02_easy dataset. On the left is the breakdown of total optimization and marginalization time. On the right is the detail time spent for marginalization only. As shown, the marginalization time is mostly spent on recovering the measurement covariances. **(Right pair)** The same run-time analysis on the V1_02_medium dataset.

the fixed-lag smoother retains a constant size optimization window. The time spent on sparsification, however, varies across datasets but remain bounded through the sequence. From Fig. 4.11a, it is shown that the total marginalization time, including sparsification, per frame is correlated with the number of features being optimized. In fact, in the MH_02_easy dataset, our method spends more time in marginalization than in the harder V1_02_medium dataset.

We have also time profiled each step in the sparsification pipeline shown in Fig. 4.11. It is important to note that the majority of time is spent on recovering the measurement information using (4.13). However, there is a significant room for improvement in the implementation, as (4.13) is clearly highly parallelizable, although we do not take advantage of that here. Furthermore, one can also postpone sparsification step depending on computational resources as shown in [22].

Chapter 5

Toward Invariant VIO on Matrix Lie Group

IIn this chapter, we examine another challenge of modern state estimation algorithms - Nonlinearity. With the presence of 3D rotations, current VIO optimization no longer lies on the Euclidean space but on a manifold (eg. $SO(3)$ explained in Chapter 3.4). An iterative solver is required to solve the nonlinear problem by linearizing the cost function in each iteration. In addition, an EKF and a fixed-lag smoother marginalize out state variables to reduce the dimensionality of the state space. The general pipeline is shown in Fig. 3.3

Not only does marginalization decrease system efficiency as shown in Chapter 4, but also create the well-known *consistency* issue if not dealt with correctly. An inconsistent state estimator is proven to present poor accuracy and fail to fuse measurements correctly. This could lead to divergence as shown in Fig. 5.1.

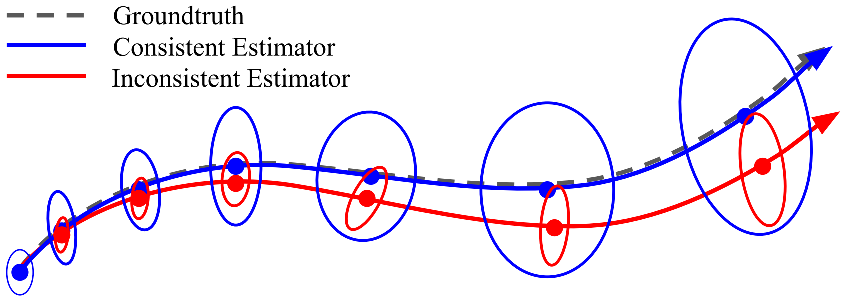


Figure 5.1: This figure shows an example of a consistent estimator and an inconsistent estimator. In the consistent case, the covariance estimate captures the error between the estimated path and the groundtruth. In the inconsistent case, the covariance ellipse is underestimated (overconfident). As a result, the accuracy is deteriorated.

The current state-of-the-art VIO systems require artificial remedies to cope with inconsistency. Our formulation, OKVIS [48] and VINS-MONO [59] employ the First-Estimate-Jacobian (FEJ) method. Another option for a better result is the Observability-Constrained (OC) Fixed-lag smoother [36]. These methods have proven by [62], [37, 38], [49], and [34] to outperform standard EKF and fixed-lag smoother. First-Estimate-Jacobian is an easy and quick fix, however, it has major disadvantages:

1. The linearization points used for Jacobian calculations are suboptimal, which can potentially create large linearization error.
2. The state variables affected by fixed linearization points are no longer optimizable.

Furthermore, FEJ also affects the previously proposed method using sparsification because the reinserted nonlinear factors are evaluated at the suboptimal linearization points. In the following sections, we investigate the recently developed *group affine property* used in the Invariant EKF [4], which has shown to solve the consistency issue without additional steps. We also present our formulation for a novel IMU preintegration model based on the theory. As part of the on-going future work, we would like to derive a complete fixed-lag VIO based on the group property. To our knowledge, this is the first work to formulate IMU preintegration and a fixed-lag VIO system using such property.

5.1 Problem Formulation

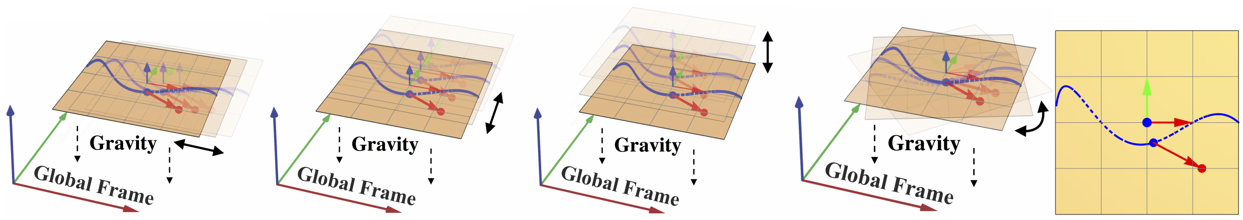
After a marginalization step, a fixed-lag VIO factor graph would include both nonlinear factors and a linearized prior (dense or the sparsified ones) as shown in Fig. 4.5b and Fig. 4.5d. The Jacobian for the prior factors is created based on the old linearization points while the states are continuously updated. The combination of "old" and "new" Jacobians causes inconsistency because some unobserved state space is wrongly updated [36]. The *observability* property of the state estimator is violated.

5.1.1 Local and Global Measurements

To understand system observability, we first look at two types of measurements in a usual SLAM system – local and global measurements. A local measurement is with respect to the local robot frame, such as visual and odometry measurements. This type of measurement does not provide any global (usually the earth) information. In contrast, GPS and absolute pressure measurements are considered global because the measurement provides information about the global status. A typical SLAM system can include both local and global measurements. A VIO system in particular directly obtains global roll and pitch angles because of IMU and gravity alignment.

5.1.2 Consistency and Observability

Mathematically, [37, 38], [49] have shown there are four unobserved spaces (nullspace) using Observability Gramian, which provides a system's observability characteristics. It means that a transformation in any of the four spaces does not affect the cost function and the state estimates.



(a) x shift (b) y shift (c) z shift (d) yaw shift (e) Local frame

Figure 5.2: There are four unobserved spaces in a 3D VIO as shown in this figure. It means that from local information, a VIO system cannot determine the global x, y, z and yaw direction. The local frame remains the same as shown in the right-most figure. However, global roll and pitch can be determined because of IMU and gravity alignment.

These four spaces are global yaw and 3D global translation as shown in Fig. 5.2. In the figure, a robot (blue dot) is moving along a path (blue line), and at some point observes a landmark (red). All four transformations are perceived the same in the local frame because the z axis always aligned with gravity. However, if the roll or pitch changes, the cost increases since the gravity is not aligned with IMU anymore.

However, when traditional EKF-based methods and fixed-lag smoothers propagate and marginalize variables, the Huang et al.'s notice that the nullspace of Observability Gramian decreases. The global yaw direction, even if it is unobservable, is somehow updated and its uncertainty is underestimated. This leads to an inconsistent state estimator which significantly affects its accuracy.

As an example in Fig. 5.3, suppose we are estimating a nonlinear function (such as a SLAM problem) with nonlinear nullspaces shown as the yellow dotted line in the left-most graph. Any solution along the nullspace is an optimal solution since the curve represents the minimum. However, we look at the case when we solve the function with Jacobians evaluated at two linearization points (cross and circle). The middle two graphics show during one optimization step, if we linearize the function at two different points, the cost functions looks different. If we solve the problem individually, this creates no problem because iteratively we can reach the minimum. However, if we combine the Jacobians evaluated at two linearization points, the combined cost function looks something like the right-most graph. Notice the nonlinear surface becomes very different from the original (left-most). More importantly, the nonlinear nullspace has disappeared and we become more certain about our solutions being in the middle region.

Fig. 5.3 illustrates one important characteristic about marginalization and a fixed-lag smoother: after marginalization, a fixed-lag VIO combines factors that evaluated at different linearization points. This lead to solution space being reduced and affect system outputs. From the perspective of *observability* in a nonlinear state estimation problem, combining different linearization points in the presence of nonlinear nullspace reduces its nullspace dimension. It means that through the propagation of the linearized equation, we artificially gain information on unobserved space such as the global yaw direction in the case of traditional EKF and a fixed-lag smoother.

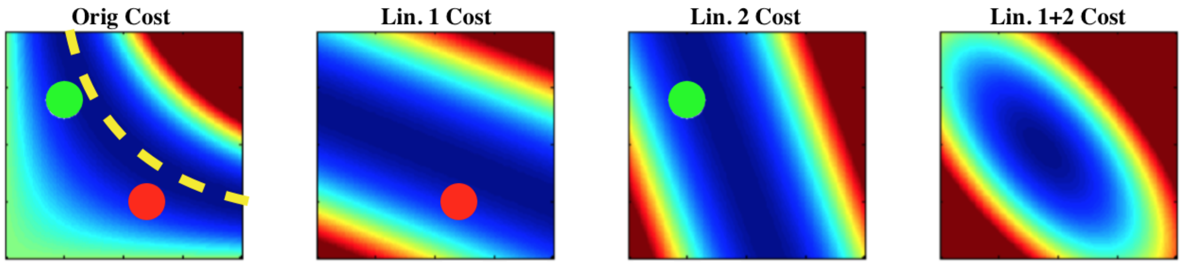


Figure 5.3: This figure illustrates an example showing when combining two different linearization points (green and red), the nonlinear nullspace (yellow dotted line) could disappear. This leads to inconsistency for an EKF-based and fixed-lag smoothing-based state estimator.

5.1.3 Existing Methods for Consistency

Current solutions, proposed by [37, 38, 49], look at the Jacobians (thus information) matrix in a fixed-lag VIO system shown in Fig. 4.1. Specifically, we split all measurement factors $\mathcal{Z}$ into ones that are adjacent to the marginalized variables $\mathcal{Z}_M$ and those that do not $\mathcal{Z}_R$, such that $\mathcal{Z}_M \cup \mathcal{Z}_R \in \mathcal{Z}$. At time step k', we look at the information matrix of a batch MAP estimator and a fixed-lag smoother which is detailed in [36].

In the batch MAP estimator, we sum up all the factors according to (3.3), the information matrix denoted by A_b at time k' is constructed in the similar way as (4.5) and (4.6):

$$\mathbf{A}_b(k') = \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_M} \mathbf{H}_{ij}^\top(k')\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}(k') + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_A} \mathbf{H}_{ij}^\top(k')\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}(k') + \sum_{\kappa=1}^{k'} \mathbf{\Phi}_\kappa^\top(k')_{ij}\mathbf{Q}_\kappa'^{-1}\mathbf{\Phi}_\kappa(k') \tag{5.1}$$

which can be partitioned into blocks

$$\mathbf{A}_b(k') = \begin{bmatrix} \mathbf{A}_{mm}(k') & \mathbf{A}_{mr}(k') & \mathbf{0} \\ \mathbf{A}_{rm}(k') & \mathbf{A}_{rr}(k') & \mathbf{A}_{rn}(k') \\ \mathbf{0} & \mathbf{A}_{nr}(k') & \mathbf{A}_{nn}(k') \end{bmatrix} \tag{5.2}$$

notice that every Jacobian H's are evaluated at the latest state estimate k'. In contrast, if we look at a fixed-lag case, where $\mathcal{Z}_M$ are evaluated at the state estimates from $0:k$:

$$\begin{aligned} \mathbf{A}_b(k') = \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_M} \mathbf{H}_{ij}^\top(k)\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}(k) + \sum_{\mathbf{z}_{ij} \in \mathcal{Z}_A} \mathbf{H}_{ij}^\top(k')\mathbf{R}_{ij}^{-1}\mathbf{H}_{ij}(k') + \\ \sum_{\kappa=1}^{m-1} \mathbf{\Phi}_\kappa^\top(k)_{ij}\mathbf{Q}_\kappa'^{-1}\mathbf{\Phi}_\kappa(k) + \sum_{\kappa=m}^{k'} \mathbf{\Phi}_\kappa^\top(k')_{ij}\mathbf{Q}_\kappa'^{-1}\mathbf{\Phi}_\kappa(k') \end{aligned} \tag{5.3}$$

which can be partitioned into blocks

$$\begin{aligned}\mathbf{A}_b(k') &= \begin{bmatrix}\mathbf{A}_{mm}(k) & \mathbf{A}_{mr}(k) & \mathbf{0}\\ \mathbf{A}_{rm}(k) & \mathbf{A}_{rr}(k) & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0}\end{bmatrix} + \begin{bmatrix}\mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{A}_{rr}(k') & \mathbf{A}_{rn}(k')\\ \mathbf{0} & \mathbf{A}_{nr}(k') & \mathbf{A}_{nn}(k')\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{A}_{mm}(k') & \mathbf{A}_{mr}(k') & \mathbf{0}\\ \mathbf{A}_{rm}(k') & \boxed{\mathbf{A}_{rr}(k) + \mathbf{A}_{rr}(k')} & \mathbf{A}_{rn}(k')\\ \mathbf{0} & \mathbf{A}_{nr}(k') & \mathbf{A}_{nn}(k')\end{bmatrix}\end{aligned} \tag{5.4}$$

The problem is the boxed block, where two different linearization points are used to evaluate the same nonlinear function (similar to Fig. 5.3.

The current literature suggests two methods to tackle the problem:

1. The First-Estimate-Jacobian requires the boxed term $A_{rr}(k')$ to $A_{rr}(k)$, which forces factors connected to variables adjacent to the marginalized variables to use the "old" state estimate.

2. As opposed to using the oldest linearization points, the Observability-Constrainted method performs an extra optimization, which selects the best linearization points that simultaneously satisfy the observability constraints:

$$\begin{aligned}&\underset{\chi_R^\star, \chi_N^\star}{\arg\min}\, \|\chi_R^\star - \hat{\chi_R}(k')\|^2 + \|\chi_N^\star - \hat{\chi_N}(k')\|^2\\ &\text{subject to} \quad \mathbf{A}(k')\mathbf{N}(k') = \mathbf{0}\end{aligned} \tag{5.5}$$

   where $\mathbf{N}$ is a designed choise that defines the desired nullspace with correct dimension.

5.2 Group Affine Property and Invariant Smoothing

As shown in Section 5.1.2, combining Jacobians with different linearization points causes inconsistency and as a consequence, a nullspace disappears. Recently, however, [4] has suggested that while Jacobians are the problems of inconsistency, state space representations and error function definitions have similar effects. Bonnabel et al. propose an *Invariant EKF* framework that fixes inconsistency by defining state estimation on a different state space.

For conventional EKF and fixed-lag methods, the optimization state space is often in the Euclidean space with rotation being in the $SO(3)$ manifold. The nonlinear factors and their error functions, therefore, are defined on the Euclidean space and are dependent on the linearization points. Recently, [3] have shown that 1) if the state space and its uncertainty are represented on a general manifold (eg. matrix Lie group), and 2) the system satisfies the *group affine property*, then the consistency problem can be fixed. As a key result from [4], they have shown that the error propagation is *exact* linear to the first-order by utilizing the Lie group-Lie algebra correspondence (Section 3.4.2). The invariant framework presents the following benefits:

1. The Jacobians no longer depend on the state estimates (linearization points), but rather, some linear function of the measurements themselves. In others words, after marginalization steps, the Jacobians automatically account for the nonlinear space change, and the original nullspaces are preserved.
2. Error propagation is linear, and thus linearization points do not affect the cost function such as in the case of Fig. 5.3.

While the group affine property is shown in both continuous and discrete systems, in this thesis we focus on the discrete system only. The property for continuous systems is detailed in [4]. For the following sub-sections we start by introducing general dynamic systems using abstract groups, and specifically in the case of matrix Lie group and the group affined property itself.

5.2.1 Linear Systems on Groups

In general, a diecrete linear systems on $\mathbb{R}^n$ is defined as

$$\begin{aligned} x_{n+1} &= F_n x_n + a_n \\ y_n &= H_n x_n \end{aligned} \tag{5.6}$$

The estimation error between measured state x and estimated $\hat{x}$ is

$$e_{n+1|n} = x_{n+1} - \hat{x}_{n+1|n} = F_n x_n - F_n \hat{x}_n | n = F_n e_n \tag{5.7}$$

and because the linear matrix F_n, the error propagation is *autonomous*, meaning that it does not depend on state estimate but only the past errors. We would like to utilize matrix Lie group-Lie algebra to achieve similar property in a nonlinear system. First, following [5], we define a general system dynamics by substiting Euclidean operations with group operations:

$$\begin{aligned} x_{n+1} &= Aut(x_n) \cdot a_n \\ y_n &= x_n \star b_n \end{aligned} \tag{5.8}$$

where

1. ”$\cdot$” is a group law satisfies Associativity, Identity and Inversion.
2. $Aut(x_n)$ is a *group automorphism* such that $Aut(a \cdot b) = Aut(a) \cdot Aut(b)$, and $Aut(a^{-1}) = Aut(a)^{-1}$
3. ”$\star$” represents *group action* such that $\forall a, b, \in G, \forall y \in Y$, $a \star (b \star y) = (a \cdot b) \star y$.

In our case when we use matrix Lie group, the group law is matrix multiplication, and group action is matrix-vector product.

Now with group operations, we define the error function in group using *left*-invariant error $\hat{x}^{-1}x$ as opposed to the *right*- invariant error $x\hat{x}^{-1}$. Both are shown to be equivalent but math derivations will be different. With left invariant error definition, we can achieve an autonomous error function following:

$$\begin{aligned}
e_{n+1} &= \hat{x}_{n+1}^{-1} \cdot x_{n+1} \\
&= [Aut_n(\hat{x_n}) \cdot a_n]^{-1} \cdot [Aut_n(x_n) \cdot a_n] \\
&= a_n^{-1} \cdot Aut_n(\hat{x_n})^{-1} \cdot Aut_n(x_n) \cdot a_n \\
&= a_n^{-1} \cdot Aut_n(\hat{x_n}^{-1} x_n) \cdot a_n \\
&= a_n^{-1} \cdot Aut_n(e_n) \cdot a_n \\
&= In_{a_n}^{-1}(\phi_n(e_n))
\end{aligned} \tag{5.9}$$

where $In: \; x \mapsto g \cdot x \cdot g^{-1}$ is a *group inner automorphism.*

5.2.2 Group Affine Dynamics and Properties

The key to exhibit autonomous error propagation is the factorization of the linear map, such as F_n in (5.6) and $Aut(x_n)$ in (5.8). To summarize the group dynamics, we can treat the propagation of state as some combined automorphism ψ:

$$x_{n+1} = \psi_n(x_n) \tag{5.10}$$

For an autonomous error propagation, it is required that

$$\hat{x}_{n+1}^{-1} \cdot x_{n+1} = \psi(\hat{x}_n)^{-1} \cdot \psi(x_n) = \mu(\hat{x}_n^{-1} \cdot x_n) \tag{5.11}$$

where μ is also an automorphism of a group.

[4] have shown that having autonomous error properties ensure the specific dynamics system called the group affine dynamics, which follow below definitions.
Definition 1:

$$\forall x_1, x_2 \in G : \quad \psi(x_1 \cdot x_2) = \psi(x_1)\psi(\mathbf{Id})^{-1}\psi(x_2) \tag{5.12}$$

Definition 2:

$$\forall x_1, x_2 \in G : \quad \mu(x_1^{-1} x_2) = \psi(x_1)^{-1} \cdot \psi(x_2) \tag{5.13}$$

5.2.3 Application to Matrix Lie Group

So far, the derivation of the general linear system has been using an arbitrary abstract group. In our application, we focus on the matrix Lie group and utilize the Lie group - Lie algebra correspondence. First, we can represent error e_n using a linear error ξ_n on the tangent space:

$$e_n = \exp(\xi_n)) \tag{5.14}$$

Substituting e_n with $\exp(\xi_n))$ and applying to (5.14) and (5.9), the Lie group-Algebra correspondence allows us to represent μ with an Adjoint map of Lie group:

$$\mu(\exp(\xi)) = \exp(A\xi) \tag{5.15}$$

This property ensures that by formulating the state space using matrix Lie group with group affine dynamics, the new error becomes a linear function of the previous error.

5.2.4 Invariant Smoothing

When performing nonlinear optimization on factor graph, we could utilize the group affine dynamics to achieve autonomous error propagation. Because of the Lie group - Lie algebra correspondences, we can find a mapping between the nonlinear and linear space by utilizing exponential and log map, as opposed to Fig. 3.3. This enables the optimizer to perform linear updates in a nonlinear space, but the Jacobian will be independent of the state estimates and thus eliminate the consistency issue. The new pipeline for invariant smoothing can be shown in Fig. 5.4:

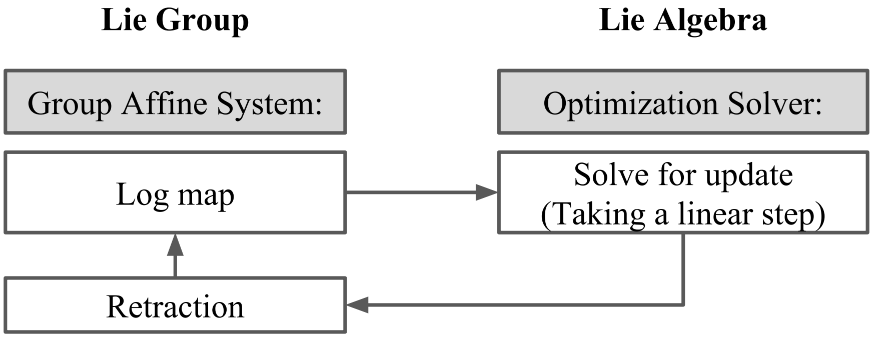


Figure 5.4: The Lie group - Lie algebra correspondence enables the conversion between linear and nonlinear space exactly. Therefore, the Jacobian automatically takes into account the geometry of the nonlinear space, and thus solve the consistency issue.

5.3 IMU Preintegration using Group Affine Property

A VIO system also follows group affine dynamics as shown in [5]. Instead of employing First-Estimate Jacobian to solve the consistency issue, we would like to formulate the VIO system using the group affine property. In this thesis, we propose a novel IMU preintegration method following [29] but formulate the error function using such property. We then compare our formulation against [29] to show that our formulation exhibit faster convergence.

5.3.1 IMU Preintegration Dynamics

Following the IMU propagation model from [29], our proposed method utilizes discrete integration as opposed to the continous one by [23]. We assume that between time t and $t+\Delta t$, the acceleration a^W and rotation rate ω^W measurements remain constant. This is called the zero-order hold for discrete-time approximation:

$$\begin{aligned}
\mathbf{R}(t+\Delta t) &= \mathbf{R}(t)\exp(\omega^W \Delta t) \\
\mathbf{v}(t+\Delta t) &= \mathbf{v}(t) + a^W(t)\Delta t \\
\mathbf{p}(t+\Delta t) &= \mathbf{p}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}a^W(t)\Delta t^2
\end{aligned} \tag{5.16}$$

Now substitute the raw IMU measurements, $\tilde{a}^B(t)$ and $\tilde{\omega}^B$, for $\omega^W(t)$ and $a^W(t)$ from (3.5), we obtain:

$$\begin{aligned}
\mathbf{R}(t+\Delta t) &= \mathbf{R}(t)\exp((\tilde{\omega}(t) - \mathbf{b}^g(t) - \eta^g(t))\Delta t) \\
\mathbf{v}(t+\Delta t) &= \mathbf{v}(t) + g\Delta t + \mathbf{R}(t)(\tilde{a}(t) - \mathbf{b}^a(t) - \eta^a(t))\Delta t \\
\mathbf{p}(t+\Delta t) &= \mathbf{p}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}g\Delta t^2 + \frac{1}{2}\mathbf{R}(t)((\tilde{a}(t) - \mathbf{b}^a(t) - \eta^a(t))\Delta t^2 \\
\mathbf{b}^a(t+\Delta t) &= \mathbf{b}^a(t) + \eta^{ba} \\
\mathbf{b}^g(t+\Delta t) &= \mathbf{b}^g(t) + \eta^{bg}
\end{aligned} \tag{5.17}$$

However, as shown in Section 3.2.1, conventional IMU integration used in factor graphs are slow and inefficient. Instead, using IMU preintegration, we accumulate all IMU measurements between two visual measurements of time $k = i$ and $k = j$. We also assume the IMU is synchronized with the camera and every two IMU measurement are Δt apart. The preintegration scheme is modeled as

$$\begin{aligned}
\mathbf{R}_j &= \mathbf{R}_i \prod_{k=i}^{j-1} \exp((\tilde{w} - \mathbf{b}_k^g - \eta_k^{gd})\Delta t) \\
\mathbf{v}_j &= \mathbf{v}_i + g\Delta t_{ij} + \sum_{k=i}^{j-1} R_k(\tilde{a_k} - \mathbf{b}_k^a - \eta_k^{ad})\Delta t \\
\mathbf{p}_j &= \mathbf{p}_i + \sum_{k=i}^{j-1} v_k \Delta t + \frac{1}{2}g\Delta t\Delta t_{ij} + \frac{1}{2}\sum_{k=i}^{j-1} R_k(\tilde{a_k} - \mathbf{b}_k^a - \eta_k^{ad})\Delta t^2 \\
\mathbf{b}_j^a &= \mathbf{b}_i^a + \Delta t \eta^{ba} \\
\mathbf{b}_j^g &= \mathbf{b}_i^g + \Delta t \eta^{bg}
\end{aligned} \tag{5.18}$$

To avoid recomputing the preintegrated measurement when state estimates change, we also adopt the local preintegration method by eliminating gravity and constant velocity effect. This is to keep a relative IMU measurement $\Delta\mathcal{X}_{ij}$ constant in the presense of optimization. However, with the presense of time-varying bias, IMU measurements also vary. To account for bias, [29] suggested the use of first-order-approximation for measurement bias correction. Instead of recomputing the integral, we have $\Delta\mathcal{X}_{ij} \oplus \frac{\partial \Delta\mathcal{X}_{ij}}{\partial b}$ when the bias is updated.

5.3.2 State Representation

Using the IMU preintegration dynamics introduced in Section 5.3.1, we will now formulate the model using group affine property as shown in (5.10). We first represent each IMU state $\mathcal{X}_i$ from (3.7) into an element of matrix Lie group $\mathcal{G}$ in the form of

$$\mathcal{X}_i = \left[\begin{array}{ccc|ccc} \mathbf{R}_i & \mathbf{v}_i & \mathbf{p}_i & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & I_{3\times3} & \mathbf{b}_i^a & \mathbf{b}_i^g \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{c} \mathbf{R}_i \\ \mathbf{v}_i \\ \mathbf{p}_i \\ \hline \mathbf{b}_i^a \\ \mathbf{b}_i^g \end{array}\right]_{\mathcal{G}} \tag{5.19}$$

With group affine property, the idea is to compute the propagated IMU state $\mathcal{X}_j$ by some automorphism of $\mathcal{X}_i$ and an IMU delta measurement $\Delta\mathcal{X}'_{ij}$. This is to follow the equation form in (5.8). However, to eliminate the effect of gravity and constant velocity, we also introduce an group element g_i and an automorephism ϕ'_i. Using the definition of inner automorphism, we finally get the IMU state propagation with:

$$\begin{aligned} \mathcal{X}_j &= \psi_i(\mathcal{X}_i) \\ &= \phi_i(\mathcal{X}_i) \cdot \Delta\mathcal{X}_{ij} \\ &= [g_i \cdot \phi'_i(\mathcal{X}_i) \cdot g_i^{-1}] \cdot [g_i \cdot \Delta\mathcal{X}'_{ij}] \end{aligned} \tag{5.20}$$

where g_i and ϕ'_i are:

$$g_i = \left[\begin{array}{c} I \\ g\Delta t_{ij} \\ \frac{1}{2}g\Delta t_{ij}^2 \\ \hline 0 \\ 0 \end{array}\right]_{\mathcal{G}}, \quad \phi'_i(\mathcal{X}_i) = \left[\begin{array}{c} \mathbf{R}_i \\ \mathbf{v}_i \\ \mathbf{p}_i + \mathbf{v}_i\Delta t_{ij} \\ \hline \mathbf{b}_i^a \\ \mathbf{b}_i^g \end{array}\right]_{\mathcal{G}} \tag{5.21}$$

With (5.20) and (5.21), it is easily proven that the IMU dynamics on matrix Lie group agrees with (5.18):

$$\begin{aligned} &\mathcal{X}_j = \phi_i(\mathcal{X}_i) \cdot \Delta\mathcal{X}_{ij} \\ &= \left[\begin{array}{c} \mathbf{R}_i \\ \mathbf{v}_i + \Delta t_{ij}(I - \mathbf{R}_i)g \\ \mathbf{p}_i + \mathbf{v}_i\Delta t_{ij} + \frac{1}{2}\Delta t_{ij}^2(I - \mathbf{R}_i)g \\ \hline \mathbf{b}_i^a \\ \mathbf{b}_i^g \end{array}\right]_{\mathcal{G}} \cdot \left[\begin{array}{c} \displaystyle\prod_{k=i}^{j-1} \exp((\tilde{w} - \mathbf{b}_k^g - \eta_k^{gd})\Delta t) \\ \displaystyle\sum_{k=i}^{j-1} \Delta R_{ik}(\tilde{a}_k - \mathbf{b}_k^a - \eta_k^{ad})\Delta t + g\Delta t_{ij} \\ \displaystyle\sum_{k=i}^{j-1} \left[\Delta v_{ik}\Delta t + \frac{1}{2}\Delta R_{ik}(\tilde{a}_k - \mathbf{b}_k^a - \eta_k^{ad})\Delta t^2\right] + \frac{1}{2}g\Delta t_{ij}^2 \\ \hline 0 + \eta^{bad} \\ 0 + \eta^{bgd} \end{array}\right]_{\mathcal{G}} \end{aligned} \tag{5.22}$$

5.3.3 IMU Preintegration Measurement Model

Before defining an IMU preintegration factor used in a factor graph, it is necessary to extract the accurate statistics of zero-mean Gaussian noise parameters from (5.22). The noise model is important for factor graph optimization because the uncertainty decides the weights of a measurement. In our proposed work, we use the left-invariant error model and represent the noise parameters on the chart (and therefore, Lie algebra). We also use the approximation for the exponential map as [29] instead of the definition for efficiency. This enables us to use the representation with Concentrated Gaussian distribution introduced in Section 3.4.4.

To extract the noise $\delta\mathcal{X}_{ij}$ for a preintegrated measurement $\Delta\tilde{\mathcal{X}}_{ij}$, we define the left invariant error as:

$$\exp(\delta\mathcal{X}_{ij}) = \Delta\mathcal{X}_{ij}'^{-1}\Delta\tilde{\mathcal{X}}_{ij}' = \left[\begin{array}{c} \Delta R_{ij}' \exp(\delta\phi_{ij}) \\ \Delta v_{ij}' + \Delta R_{ij}'\delta v_{ij} \\ \Delta p_{ij}' + \Delta R_{ij}'\delta p_{ij} \\ \hline \eta^{bad} \\ \eta^{bgd} \end{array}\right]_{\mathcal{G}}, \quad \delta\mathcal{X}_{ij} = \left[\begin{array}{c} \delta\phi_{ij} \\ \delta v_{ij} \\ \delta p_{ij} \\ \hline \eta^{bad} \\ \eta^{bgd} \end{array}\right]_{\mathcal{G}} \approx \mathcal{N}(\mathbf{0}, \Sigma_{ij}) \tag{5.23}$$

The detail derivations of $\delta\phi_{ij}, \delta v_{ij}, \delta p_{ij}$ is given in Appendix A.1.

5.3.4 IMU Preintegration Factor

Combine the group affine system in (5.20) and the noise model in (5.23), we can now define IMU preintegration factor using group affine property and matrix Lie group:

$$\begin{aligned} \mathcal{X}_j &= \psi_i(\mathcal{X}_i) \\ &= \phi_i(\mathcal{X}_i) \cdot \Delta\mathcal{X}_{ij} \\ &= \left[\phi_i(\mathcal{X}_i) \cdot \Delta\tilde{\mathcal{X}}_{ij}\right] \exp(\delta\mathcal{X}_{ij})^{-1} \\ &= \tilde{\psi}_i(\mathcal{X}_i) \exp(\delta\mathcal{X}_{ij})^{-1} \end{aligned} \tag{5.24}$$

Notice that in our formulation, the measurement is $\tilde{\psi}_i(\mathcal{X}_i)$ and the prediction is simply $\mathcal{X}_j$ as opposed to the pure relative IMU measurement from [29]. The factor residual $\mathbf{r_I}$ is therefore

$$\begin{aligned} \mathbf{r_I} &= \tilde{\psi}_i(\mathcal{X}_i)^{-1}\mathcal{X}_j \\ &= \left[\begin{array}{c} (\mathbf{R}_i\Delta\tilde{R}_{ij})^\top \mathbf{R}_j \\ (\mathbf{R}_i\Delta\tilde{R}_{ij})^\top (\mathbf{v}_i - \mathbf{v}_i) \\ (\mathbf{R}_i\Delta\tilde{R}_{ij})^\top (\mathbf{p}_j - \frac{1}{2}\mathbf{v}_i\Delta t)\delta p_{ij} \\ \hline 0 \\ 0 \end{array}\right]_{\mathcal{G}} \end{aligned} \tag{5.25}$$

In order to solve for the state update ξ_i and ξ_j, we linearize the residual at each iteration by exploiting the Lie group - Lie algebra correspondence. Moreover, we use the group affine property (5.12) and (5.15) to obtain an autonomous Jacobian:

$$
\begin{aligned}
\mathcal{X}_j &= \tilde{\psi}_i(\mathcal{X}_i)\exp(\delta\mathcal{X}_{ij})^{-1} \\
\exp(-\delta\mathcal{X}_{ij}) &= \tilde{\psi}_i(\mathcal{X}_i)^{-1}\mathcal{X}_j \\
&= \tilde{\psi}_i(\hat{\mathcal{X}}_i\exp(\xi_i))^{-1}(\hat{\mathcal{X}}_j\exp(\xi_j)) \\
&\overset{(5.12)}{=} \left[\tilde{\psi}_i(\hat{\mathcal{X}}_i)\tilde{\psi}_i(I)^{-1}\tilde{\psi}_i(\exp(\xi_i))\right]^{-1}(\hat{\mathcal{X}}_j\exp(\xi_j)) \\
&\overset{(5.13)}{=} \mu_l(\exp(\xi_i))^{-1}\left[\tilde{\psi}_i(\hat{\mathcal{X}}_i)^{-1}\hat{\mathcal{X}}_j\right]\exp(\xi_j)
\end{aligned}
\tag{5.26}
$$

Using (5.11) and (5.9), we define the automorphism μ for IMU preintegration as:

$$
\begin{aligned}
\mu_l(\mathcal{X}_i^{-1}\cdot\mathcal{X}_j) &= \psi_i(\mathcal{X}_i)^{-1}\cdot\psi_i(\mathcal{X}_j) \\
\mu_l(\mathcal{X}_i) &= \psi_i(I)^{-1}\cdot\psi_i(\mathcal{X}_i) \\
&= (\phi_i(I)\cdot\Delta\tilde{\mathcal{X}}_{ij}))^{-1}\cdot\phi_i(\mathcal{X}_i)\cdot\Delta\tilde{\mathcal{X}}_{ij}) \\
&= \Delta\tilde{\mathcal{X}}_{ij}{}^{-1}\cdot\phi_i(\mathcal{X}_i)\cdot\Delta\tilde{\mathcal{X}}_{ij} \\
&= (g_i\cdot\Delta\tilde{\mathcal{X}}_{ij}{}')^{-1}\cdot\phi_i(\mathcal{X}_i)\cdot(g_i\cdot\Delta\tilde{\mathcal{X}}_{ij}{}')
\end{aligned}
\tag{5.27}
$$

Finally, with (5.27), we could obtain the linearized error expressed as:

$$
\begin{aligned}
\log(\exp(-\delta\mathcal{X}_{ij})) &= \log(\mu_l(\exp(\xi_i))^{-1}\left[\tilde{\psi}_i(\hat{\mathcal{X}}_i)^{-1}\hat{\mathcal{X}}_j\right]\exp(\xi_j)) \\
&\overset{(BCH)}{\simeq} \epsilon + \xi_j - Adj_{\Delta\tilde{\mathcal{X}}_{ij}{}^{-1}}(A\xi_i)
\end{aligned}
\tag{5.28}
$$

where $\epsilon = \hat{\mathcal{X}}_j{}^{-1}\tilde{\psi}_i(\hat{\mathcal{X}}_i)$ is the current estimation error, the adjoint map Adj derived from the automorphism from (5.27), and A from the log map of $\phi_i(\mathcal{X}_i)$ from the property (5.15). The closed form A and $Adj_{\mathcal{X}_i}$ is given by

$$
A = \left[\begin{array}{ccccc}
\mathbf{I} & 0 & 0 & 0 & 0 \\
[g]_\times\Delta t_{ij} & \mathbf{I} & 0 & 0 & 0 \\
\frac{1}{2}[g]_\times\Delta t_{ij}^2 & \mathbf{I}\Delta t_{ij} & \mathbf{I} & 0 & 0 \\
\hline
0 & 0 & 0 & \mathbf{I} & 0 \\
0 & 0 & 0 & 0 & \mathbf{I}
\end{array}\right], \quad
Adj_{\mathcal{X}_i} = \left[\begin{array}{ccccc}
\mathbf{R}_i & 0 & 0 & 0 & 0 \\
[\mathbf{v}_i]_\times\mathbf{R}_i & \mathbf{R}_i & 0 & 0 & 0 \\
[\mathbf{p}_i]_\times\mathbf{R}_i & 0 & \mathbf{R}_i & 0 & 0 \\
\hline
0 & 0 & 0 & \mathbf{I} & 0 \\
0 & 0 & 0 & 0 & \mathbf{I}
\end{array}\right]
\tag{5.29}
$$

We can easily rearrange (5.28) into $Ax - b$ form for nonlinear optimization. However, as opposed to [29], the Jacobians are now autonomous from the state estimates but only depends on the measurements.

$$
J(\xi_i) = \left[\begin{array}{ccccc}
\Delta R_{ij}^\top & 0 & 0 & 0 & 0 \\
\Delta R_{ij}^\top[\Delta v_{ij}]_\times + \Delta R_{ij}^\top[g]_\times\Delta t_{ij} & \Delta R_{ij}^\top & 0 & 0 & 0 \\
\Delta R_{ij}^\top[\Delta p_{ij}]_\times + \frac{1}{2}\Delta R_{ij}^\top[g]_\times\Delta t_{ij}^2 & \Delta t_{ij}\Delta R_{ij}^\top & \Delta R_{ij}^\top & 0 & 0 \\
\hline
0 & 0 & 0 & I & 0 \\
0 & 0 & 0 & 0 & I
\end{array}\right]_{\mathcal{G}}, \;
J(\xi_j) = \left[\begin{array}{ccccc}
\mathbf{I} & 0 & 0 & 0 & 0 \\
0 & \mathbf{I} & 0 & 0 & 0 \\
0 & 0 & \mathbf{I} & 0 & 0 \\
\hline
0 & 0 & 0 & \mathbf{I} & 0 \\
0 & 0 & 0 & 0 & \mathbf{I}
\end{array}\right]_{\mathcal{G}}
\tag{5.30}
$$

5.3.5 Incorporating Bias Update

From the IMU dynamics (3.5), we could see the bias terms directly affects the measurements. When the bias updates, we would need to re-evaluate the preintegrated measurements $\Delta\tilde{\mathcal{X}}_{ij}$. Instead of reintegrating the preintegrated IMU measurements, we adopt the first-order update by [29]. That is, we incrementally update the measurement by incorporate a small update to the bias correction: $\tilde{\psi}_i(\mathcal{X}_i) \oplus (\frac{\partial\tilde{\psi}_i(\mathcal{X}_i)}{\partial b})b$. Since both preintegrated measurement $\tilde{\psi}_i$ and the partial derivatives $(\frac{\partial\tilde{\psi}_i(\mathcal{X}_i)}{\partial b})$ remains constant, the measurement update to (5.24) is simply adding the extra bias updates:

$$\begin{aligned}\mathcal{X}_j =& \Big(\psi_i(\tilde{\mathcal{X}}_i)(\bar{\mathbf{b}}_i^a, \bar{\mathbf{b}}_i^g) \oplus (\frac{\partial\psi_i(\mathcal{X}_i)}{\partial b})\delta b\Big) \exp(\delta\mathcal{X}_{ij})^{-1} \\ =& \boxed{\Big(\phi_i(\mathcal{X}_i) \cdot \Delta\tilde{\mathcal{X}}_{ij}(\bar{\mathbf{b}}_i^a, \bar{\mathbf{b}}_i^g) \cdot \exp((\frac{\partial\psi_i(\mathcal{X}_i)}{\partial b})\delta b\Big)} \exp(\delta\mathcal{X}_{ij})^{-1}\end{aligned} \tag{5.31}$$

where the updated measurement in the box from (5.31) looks like:

$$\tilde{\psi}_i(\mathcal{X}_i) = \left[\begin{array}{c} \mathbf{R}_i \\ \mathbf{v}_i + \Delta t_{ij}(I - \mathbf{R}_i)g \\ \mathbf{p}_i + \mathbf{v}_i\Delta t_{ij} + \frac{1}{2}\Delta t_{ij}^2(I - \mathbf{R}_i)g \\ \hline \mathbf{b}_i^a \\ \mathbf{b}_i^g \end{array}\right]_{\mathcal{G}} \cdot \left[\begin{array}{c} [\prod_{k=i}^{j-1} \exp((\tilde{w} - \bar{\mathbf{b}}_k^g)\Delta t)] \exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g) \\ [\sum_{k=i}^{j-1} \Delta\bar{R}_{ik}(\tilde{a}_k - \bar{\mathbf{b}}_k^a)\Delta t] + g\Delta t_{ij} + \frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g + \frac{\partial\Delta\bar{v}_{ij}}{\partial b^a}\delta\mathbf{b}_i^a \\ \sum_{k=i}^{j-1} [\Delta\bar{v}_{ik}\Delta t + \frac{1}{2}\Delta\bar{R}_{ik}(\tilde{a}_k - \bar{\mathbf{b}}_k^a)\Delta t^2] + \frac{1}{2}g\Delta t_{ij}^2 + \frac{\partial\Delta\bar{p}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g + \frac{\partial\Delta\bar{p}_{ij}}{\partial b^a}\delta\mathbf{b}_i^a \\ \hline 0 \\ 0 \end{array}\right]_{\mathcal{G}} \tag{5.32}$$

The detailed derivation of each partial derivative term is in Appendix A.1.

With bias updates, we would need to account the bias effects to the Jacobian of ξ_i in (5.30). The updated Jacobians for ξ_i, $J(\xi_i)$, and the Jacobian for ξ_j are:

$$J(\xi_i) = \left[\begin{array}{ccccc} \Delta R_{ij}^{\top} & 0 & 0 & 0 & \frac{\partial\Delta\bar{v}_{ij}}{\partial b^g} \\ \Delta R_{ij}^{\top}[\Delta v_{ij}]_{\times} + \Delta R_{ij}^{\top}[g]_{\times}\Delta t_{ij} & \Delta R_{ij}^{\top} & 0 & \frac{\partial\Delta\bar{v}_{ij}}{\partial b^a} & \frac{\partial\Delta\bar{v}_{ij}}{\partial b^g} \\ \Delta R_{ij}^{\top}[\Delta p_{ij}]_{\times} + \frac{1}{2}\Delta R_{ij}^{\top}[g]_{\times}\Delta t_{ij}^2 & \Delta t_{ij}\Delta R_{ij}^{\top} & \Delta R_{ij}^{\top} & \frac{\partial\Delta\bar{p}_{ij}}{\partial b^a} & \frac{\partial\Delta\bar{p}_{ij}}{\partial b^g} \\ \hline 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & 0 & I \end{array}\right]_{\mathcal{G}}, \; J(\xi_j) = \left[\begin{array}{ccccc} \mathbf{I} & 0 & 0 & 0 & 0 \\ 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & \mathbf{I} & 0 & 0 \\ \hline 0 & 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} \end{array}\right]_{\mathcal{G}} \tag{5.33}$$

5.3.6 Incremental Update for IMU Preintegration Measurement

During the preintegration process, we could incrementally update the preintegrated measurement mean and covariance. This is beneficial because it allows efficient implementation for real-time applications. Using tools from matrix Lie group, incremental update is shown below. Suppose we are adding an IMU measurement $\exp(\xi_{jj+1})$ to the preintegrated measurement $\Delta\mathcal{X}_{ij}$ such that

$\Delta X_{ij+1} = \Delta\mathcal{X}_{ij} \oplus \xi_{jj+1}$. We can separate the new IMU noise $\delta\xi_{jj+1}$ and preintegrated noise $\delta\mathcal{X}_{ij}$. By rearranging the terms, we obtain the updated IMU preintegration measurement:

$$\begin{aligned}
\Delta \tilde{X}_{ij+1} \exp(\delta X_{ij+1}) &= \Delta\tilde{\mathcal{X}}_{ij} \exp(\delta\mathcal{X}_{ij}) \exp(\tilde{\xi}_{jj+1} + \delta\xi_{jj+1}) \\
&\overset{(3.22)}{\simeq} \Delta\tilde{\mathcal{X}}_{ij} \exp(\delta\mathcal{X}_{ij}) \exp(\tilde{\xi}_{jj+1}) \exp(Jr(\xi_{jj+1})\delta\xi_{jj+1}) \\
&\overset{\text{(Def. Adj)}}{=} \left[\Delta\tilde{\mathcal{X}}_{ij} \exp(\tilde{\xi}_{jj+1})\right] \left[\exp([Adj^{-1}_{\tilde{\xi}_{jj+1}}]\delta\mathcal{X}_{ij}) \exp(Jr(\xi_{jj+1})\delta\xi_{jj+1})\right]
\end{aligned} \tag{5.34}$$

Where the new mean and covariance on the chart can be extract using the Baker-Campbell-Hausdorff (BCH) formula:

$$\begin{aligned}
\Delta \tilde{X}_{ij+1} &= \Delta\tilde{\mathcal{X}}_{ij} \exp(\tilde{\xi}_{jj+1}) \\
\delta X_{ij+1} &\overset{\text{(BCH)}}{\simeq} [Adj^{-1}_{\tilde{\xi}_{jj+1}}]\delta\mathcal{X}_{ij} + [Jr(\xi_{jj+1})]\delta\xi_{jj+1} \\
&= [Adj^{-1}_{\tilde{\xi}_{jj+1}}]\delta\mathcal{X}_{ij} + \frac{\partial \exp(\xi_{jj+1})}{\partial \xi}\frac{\partial \xi}{\partial u} \begin{bmatrix} \eta^{a}_{j+1} \\ \eta^{w}_{j+1} \\ \eta^{ba}_{j+1} \\ \eta^{bw}_{j+1} \end{bmatrix} \\
&= A\delta\mathcal{X}_{ij} + B\Sigma_{IMU}
\end{aligned} \tag{5.35}$$

The closed-form for $Adj^{-1}_{\tilde{\xi}_{jj+1}}$ and $\frac{\partial \exp(\xi_{jj+1})}{\partial \xi}\frac{\partial \xi}{\partial u}$ is given in Appendix B.2. With the ability to incrementally update, we could incrementally update the preintegration noise in (5.23) by similar first-order update in EKF:

$$\Sigma_{ij+1} = A\Sigma_{ij}A^{\top} + B\Sigma_{IMU}B^{\top} \tag{5.36}$$

5.4 Experimental Results

5.4.1 Implementation

We implemented the proposed algorithm using MATLAB. Simulated trajectoriess are generated based on the IMU models shown in (3.5) with various noise levels and bias noise parameters. To test the IMU preintegration models, we simulate IMU measurements and noisy position priors into a pose graph shown in Fig. 5.5.

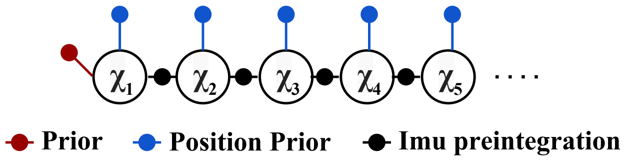
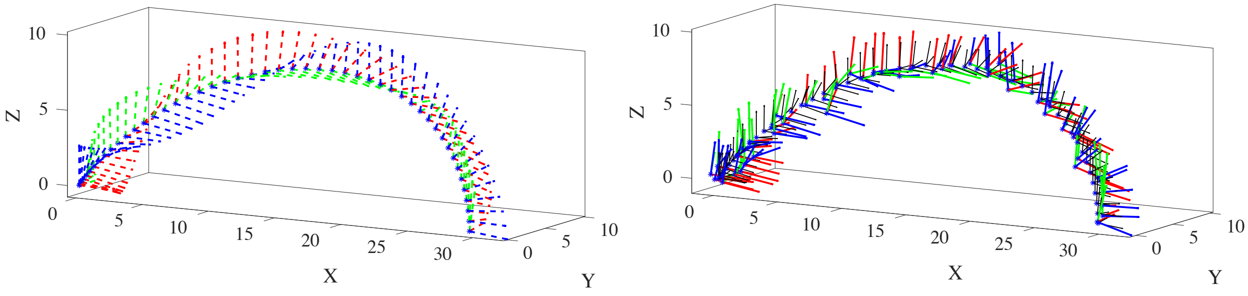


Figure 5.5: To test IMU preintegration formulation, we create a simple pose graph that only includes position priors and new IMU factors. Each group element $\mathcal{X}_i = [\mathbf{R}_i, \mathbf{v}_i, \mathbf{p}_i, \mathbf{b}^a_i, \mathbf{b}^g_i]_{\mathcal{G}}$ as shown in (5.19).

We compare our formulation against the GTSAM implementation by [29] (denote GTSAM-PREINT). To ensure fair comparisons, both our implementation in Matlab and GTSAM-PREINT receives the same trajectories file and use Levenberg-Marquart (LM) optimizer with the same stopping criterion for the factor graph. Here we perform multiple Monti-Carlo simulations by varying acceleration and rotation noise. For all experiments shown, we construct the full factor graph with randomly initialized poses (Fig. 5.6) and solve the entire graph with batch optimization. We have also tested incremental update and the results are almost identical. Fig. 5.6 shows a sample trajectory generated by our simulator.



(a) Simulated groundtruth trajectory based on (3.5).

(b) Randomly initialized poses.

Figure 5.6: This figure shows the groundtruth simulated trajectory **(a)** and the randomly initialized poses **(b)**. We input the exact same data into both our algorithm and GTSAM-PREINT by [29] to directly compare the accuracy and convergence.

5.4.2 Simulation

For our simulation experiments, we have fixed the parameters for bias accelerometer noise η^{bg} and bias gyro noise η^{ba}. With total 50 different noise values, we run 20 different initializations for each value in order to compute the mean and variance for the statistics. The summary of our parameter setting is shown in Table 5.1.

Table 5.1: Parameters for simulation experiments

	LM Solver Parameters			Noise Parameters			
	Rel. Err.	Abs. Err.	Max Iteration	η^a (m/s^2)	η^g (rad/s)	η^{ba} (m/s^3)	η^{bg} (rad/s^2)
Accel Noise Test	$1e^{-5}$	$1e^{-5}$	20	0.05 to 1	0.01	$\mathcal{N}(1e^{-3}, 1e^{-4})$	$\mathcal{N}(1e^{-3}, 1e^{-4})$
Gyro Noise Test	$1e^{-5}$	$1e^{-5}$	20	0.01	0.05 to 1	$\mathcal{N}(1e^{-3}, 1e^{-4})$	$\mathcal{N}(1e^{-3}, 1e^{-4})$
Accel Bias Test	$1e^{-5}$	$1e^{-5}$	20	0.01	0.01	0.05 to 1	$\mathcal{N}(1e^{-3}, 1e^{-4})$
Gyro Bias Test	$1e^{-5}$	$1e^{-5}$	20	0.01	0.01	$\mathcal{N}(1e^{-3}, 1e^{-4})$	0.05 to 1

In the first test we compare the accuracy and numer of iterations until convergence with respect to the accelerometer noise η^a from 0.05 to 1 m/s^2. The results are shown in Fig. 5.7. Similarly, we performed Monti-Carlo runs with varying gyroscope noise η^g, accelerometer bias noise η^{ba} and gyroscope bias noise η^{bg} shown in Fig. 5.8, Fig. 5.9, and Fig. 5.10.

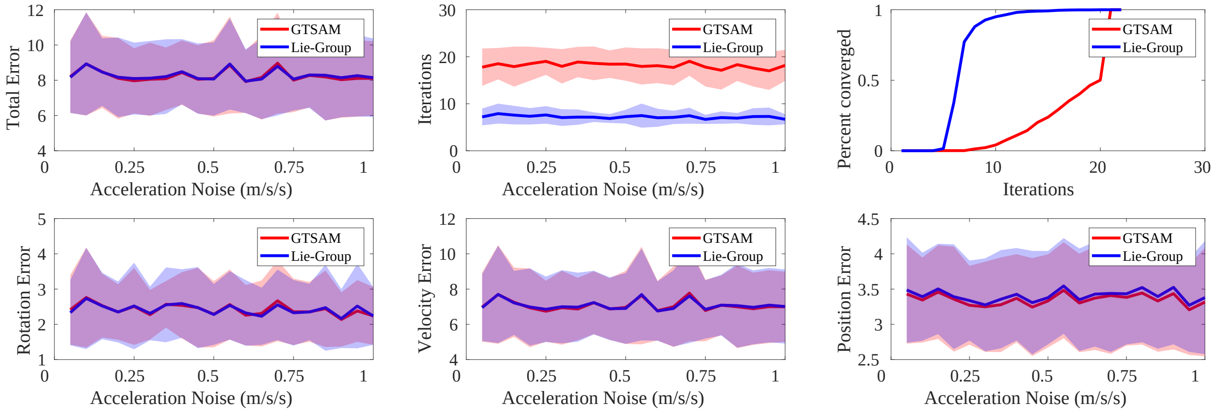


Figure 5.7: Accelerometer noise test.

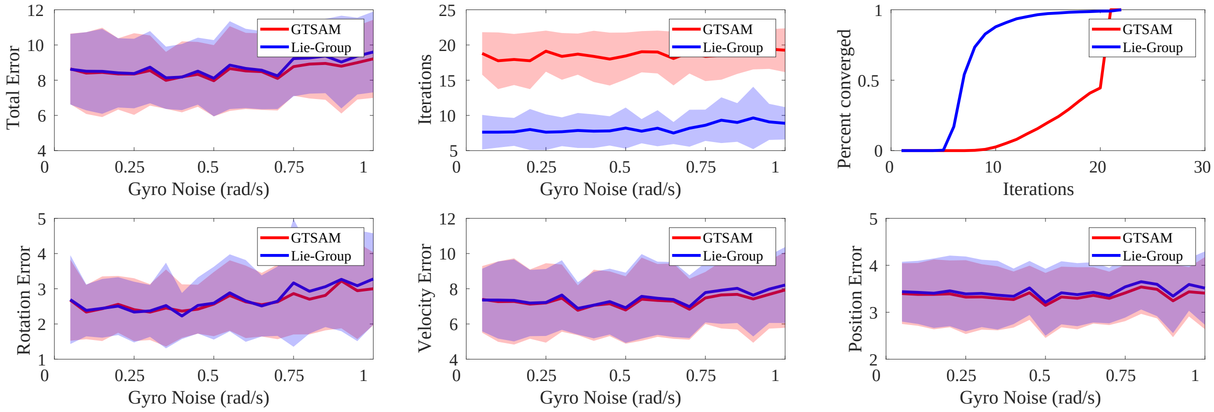


Figure 5.8: Gyroscope noise test.

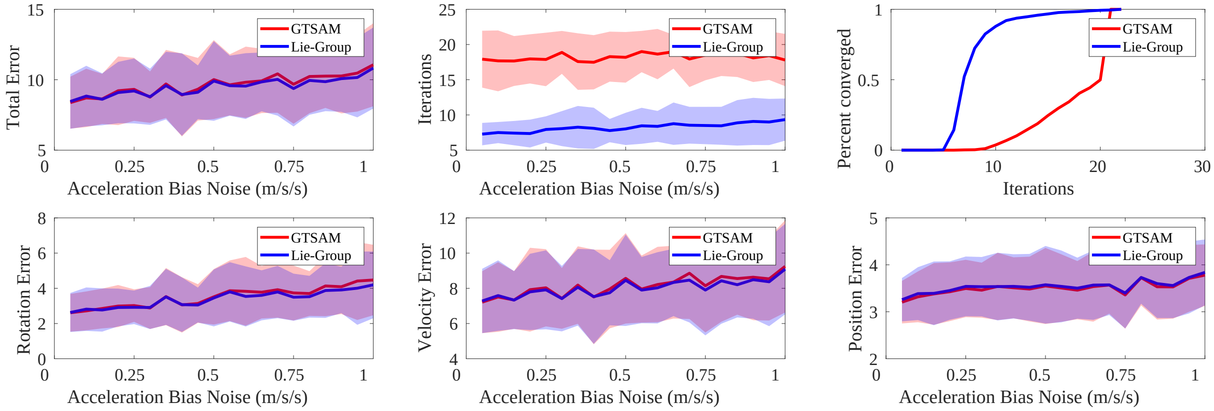


Figure 5.9: Accelerometer bias noise test.

In all four experiments, our formulation and GTSAM-PREINT achieves almost identical accuracy with very slight variations because both our methods perform similar first-order approximations. However, due to numerical or the differences between MATLAB and C++, the converged values

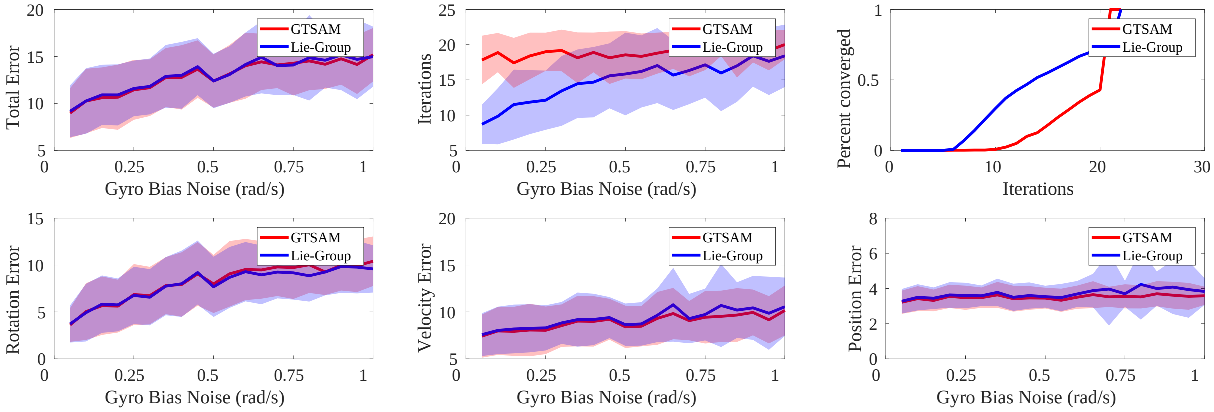


Figure 5.10: Gyroscope bias noise test.

do not perfectly match. However, from both experiments, our formulation requires much fewer iterations to converge. We think this is because the matrix Lie group formulation takes into account the nonlinear space of the problem, and therefore the Jacobians reflect better descent directions. Though remains as future works, we would like to investigate the idea of *natural gradient descent* in connection to our IMU preintegration formulation.

Chapter 6

Conclusions

6.1 Contributions

In this thesis, we have introduced two current challenges for existing fixed-lag visual-inertial odometry algorithms. With these two challenges in mind, we design and implement a VIO system and present an on-going work to address the effects of these challenges upon the accuracy and efficiency of a real-time VIO system.

First, we investigate the densification of information matrix due to marginalization steps in a fixed-lag smoothing framework. Marginalization creates “fill-in” and significantly reduces the efficiency of a fixed-lag system. As opposed to the existing methods, we propose a novel fixed-lag smoothing VIO framework with online information sparsification. Comparing to the existing methods, the proposed algorithm not only retains the sparsity and nonlinearity of the original optimization but also minimizes information loss in the presence of marginalization. Furthermore, we propose a factor graph topology that retains the structural similarity of the original fixed-lag window. This allows continuous operation of our algorithm, which is essential for navigating in exploration applications. The proposed method is compared to the existing fixed-lag VIO systems and achieves competitive results. From the time analysis has shown the potential of real-time implementation.

In the second part of the thesis, we address the issue of the well-known consistency problem in traditional linearized nonlinear systems such as an EKF-based method and a fixed-lag smoother. While the current solutions such as First-Estimate-Jacobian provides simple fixes, it disables the abilities to re-estimate specific state variables. Instead, we apply the recently developed group affine dynamics using Matrix Lie Group, and utilize tools from Lie Group - Lie Algebra correspondence to improve the state-of-the-art VIO formulation. Toward invariant VIO, in this thesis, we present our preliminary works on a novel IMU preintegration scheme using group affine dynamics. Our method is different from the IMU preintegration method presented by [29], which takes into account the manifold structure but does not solve the consistency problem. To validate our approach, we perform extensive simulations and directly compared to the original implementation. We have shown that our method achieves similar accuracy while converges with much fewer iterations.

The proposed algorithms aim to improve the existing VIO systems by providing alternative formulation to the current VIO methods. Toward a more robust, accurate and efficient state estimator, this thesis also seeks to provide insights and understandings of the underlying “structure” of the factor graph optimization framework.

6.2 Discussion and Future Work

The proposed IMU preintegration using group affine property is part of the on-going work toward a full invariant VIO system. The next steps involve deriving visual measurements and incorporating information sparsification using the property on Matrix Lie Group. We believe that the Lie Group formulation of VIO system will not only eliminate consistency problem but also further enhance information sparsification presented in the first part of the thesis. By utilizing the duality between Lie Group and Lie Algebra, the reinserted nonlinear factors can be further optimized under the invariant framework.

In regarding the proposed sparsification framework, there are many possibilities for further investigation and potentially better solutions. First, we would like to explore other factor graph topologies. For example, a possible direction is utilizing Chow-Liu Tree by the measure of mutual information between variable nodes in the factor graph. [51] also proposes other possible topologies such as a cliquey-graph to be investigated. More quantitative comparisons between topologies could be an interesting direction to pursue.

Another possibility is to investigate different cost function for the sparsification in the equation (4.7). Since Kullback-Leibler Divergence is not a symmetric measure, other divergence metrics such as Cauchy-Schwarz divergence will be possible substitutions for the cost function.

Finally, there is existing literature performing a different form of sparsification on a VIO system. For instance, instead of sparsifying the backend factor graph, [11] sparsifies the frontend features by formulating a submodular optimization for selecting a better subset of features based on some information matrics. It will be interesting to investigate other sparsification methods in terms of the trade-off between accuracy and efficiency in comparing to the VIO systems proposed in this thesis.

The idea of using Matrix Lie Group and representing covariances on the chart of a Lie Group has been investigated in this thesis but also existing robotics literature such as [2]. However, to the best of our knowledge, there is no work connecting the topics of natural gradient descent and information geometry for probabilistic inference in a factor graph. The mathematical ideas of probability, information and geometry could provide a further understanding of representing uncertainty and state estimation in nonlinear or alternative spaces, which could lead to derivations of more accurate and efficient solvers.

Bibliography

[1] Sameer Agarwal, Keir Mierle, and Others. Ceres solver. http://ceres-solver.org. 2.1, 2.2

[2] Timothy D. Barfoot. *State Estimation for Robotics*. Cambridge University Press, New York, NY, USA, 1st edition, 2017. ISBN 1107159393, 9781107159396. 2.3, 3.4.1, 3.4.2, 3.4.3, 3.11, 3.4.4, 6.2

[3] Axel Barrau and Silvere Bonnabel. An EKF-SLAM algorithm with consistency properties. 2015. URL http://arxiv.org/abs/1510.06263. 2.3, 5.2

[4] Axel Barrau and Silvere Bonnabel. The invariant extended Kalman filter as a stable observer. *IEEE Transactions on Automatic Control*, 62(4):1797–1812, 2017. ISSN 00189286. doi: 10.1109/TAC.2016.2594085. 1.2.2, 1.3, 2.3, 5, 5.2, 5.2, 5.2.2

[5] Axel Barrau and Silvere Bonnabel. Linear observation systems on groups (I). working paper or preprint, December 2017. URL https://hal-mines-paristech.archives-ouvertes.fr/hal-01671724. 5.2.1, 5.3

[6] Axel Barrau and Andsil Ere Bonnabel. Invariant Kalman Filtering. *Annu. Rev. Control Robot. Auton. Syst*, 1(521):1–5, 2018. doi: 10.1146/annurev-control-060117. URL https://doi.org/10.1146/annurev-control-060117-105010{%}0Awww.annualreviews.org. 1.2.2

[7] Michael Bloesch, Sammy Omari, Marco Hutter, and Roland Siegwart. Robust visual inertial odometry using a direct EKF-based approach. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015. ISBN 8610828378018. doi: 10.3929/ethz-a-010782581. 2.1, 4.3.3

[8] M Burri, J Nikolic, P Gohl, T Schneider, J Rehder, S Omari, M Achtelik, and R Siegwart. The EuRoC MAV datasets. *Intl. J. of Robotics Research (IJRR)*, 2015. URL http://projects.asl.ethz.ch/datasets/doku.php?id=kmavvisualinertialdatasets. 1.3, 4.9, 4.3.3, 4.10

[9] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. on Robotics (TRO)*, 32(6):1309–1332, 2016. ISSN 15523098. doi: 10.1109/TRO.2016.2624754. 1.1, 1.2, 1.2.2

[10] N. Carlevaris-Bianco, M. Kaess, and R.M. Eustice. Generic factor-based node removal: En-

abling long-term SLAM. *IEEE Trans. on Robotics (TRO)*, 30(6):1371–1385, December 2014. 2.2

[11] Luca Carlone and Sertac Karaman. Attention and anticipation in fast visual-inertial navigation. *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3886–3893, 2017. ISSN 10504729. doi: 10.1109/ICRA.2017.7989448. 6.2

[12] Paul Chauchat, Axel Barrau Andsil, and Ere Bonnabel. Invariant smoothing on Lie Groups. 2018. URL `https://arxiv.org/pdf/1803.02076.pdf`. 2.3

[13] Gregory Chirikjian. Stochastic models, information theory, and lie groups, volume 2, 01 2012. 3.4.1, 3.4.2, 3.4.3, 3.4.4

[14] Siddharth Choudhary, Vadim Indelman, Henrik I. Christensen, and Frank Dellaert. Information-based reduced landmark SLAM. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, volume 2015-June, pages 4620–4627, 2015. ISBN 9781479969227. doi: 10.1109/ICRA.2015.7139839. 2.2

[15] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Re-integration. 36(3), 2016. ISSN 07300301. doi: 10.1145/3054739. URL `http://arxiv.org/abs/1604.01093`. 1.1

[16] F. Dellaert and M. Kaess. Factor graphs for robot perception. *Foundations and Trends in Robotics*, 6(1-2):1–139, August 2017. doi: 10.1561/2300000043. 1.1, 2, 2.2, 3.1

[17] Frank Dellaert. Factor graphs and GTSAM: A hands-on introduction. Technical report, Georgia Tech, September 2012. URL `https://research.cc.gatech.edu/borg/sites/edu.borg/files/downloads/gtsam.pdf`. 1.3, 4.3.1

[18] Jeffrey Delmerico and Davide Scaramuzza. A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots. *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 2502–2509, 2018. 2.1, 2.1, 2.2

[19] Jing Dong, Mustafa Mukadam, Frank Dellaert, and Byron Boots. Motion Planning as Probabilistic Inference using Gaussian Processes and Factor Graphs. *Robotics: Science and Systems (RSS)*. doi: 10.15607/RSS.2016.XII.001. URL `http://www.roboticsproceedings.org/rss12/p01.pdf`. 1.1

[20] Tue Cuong Dong-Si and Anastasios I. Mourikis. Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 5655–5662, 2011. ISBN 9781612843865. doi: 10.1109/ICRA.2011.5980267. 2.1, 4.2.1

[21] Ethan Eade. Lie groups for 2d and 3d transformations. 2017. 3.4.2, 3.4.3

[22] K. Eckenhoff, L. Paull, and G. Huang. Decoupled, consistent node removal and edge sparsification for graph-based SLAM. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 3275–3282, Oct 2016. doi: 10.1109/IROS.2016.7759505. 2.2, 4.2.3, 4.3.4

[23] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. Continuous Preintegration Theory for Graph-based Visual-Inertial Navigation. 2018. URL `http://arxiv.org/abs/`

1805.02774. 5.3.1

[24] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, March 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2658577. 2.1

[25] Jakob Engel, Jurgen Sturm, and Daniel Cremers. Camera-based navigation of a low-cost quadrocopter. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2815–2821, 2012. ISBN 978-1-4673-1736-8. doi: 10.1109/IROS.2012.6385458. URL http://ieeexplore.ieee.org/document/6385458/. 2.1

[26] Jakob Engel, Thomas Sch, and Daniel Cremers. LSD-SLAM: Large-Scale Direct Monocular SLAM. *Eur. Conf. on Computer Vision (ECCV)*, pages 834–849, 2014. ISSN 16113349. doi: 10.1007/978-3-319-10605-2_54. 2.1

[27] R. Eustice, M. Walter, and J. Leonard. Sparse extended information filters: insights into sparsification. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 3281–3288, August 2005. doi: 10.1109/IROS.2005.1545053. 2.2

[28] Jay Farrell. *Aided Navigation: GPS with High Rate Sensors*. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 2008. ISBN 0071493298, 9780071493291. 1.1

[29] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual-inertial odometry. *IEEE Trans. on Robotics (TRO)*, 33(1):1–21, 2017. ISSN 15523098. doi: 10.1109/TRO.2016.2597321. 1.2.2, 1.3, 2.1, 2.3, 3.2.1, 3.4, 3.2.2, 3.5, 3.3, 3.3, 3.3, 3.4.3, 3.4.3, 3.4.3, 3.4.4, 5.3, 5.3.1, 5.3.1, 5.3.3, 5.3.4, 5.3.4, 5.3.5, 5.4.1, 5.6, 6.1

[30] Udo Frese. A proof for the approximate sparsity of SLAM information matrices. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, number April, pages 329–335, 2005. ISBN 078038914X. doi: 10.1109/ROBOT.2005.1570140. 2.2

[31] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1280–1286, Nov 2013. doi: 10.1109/IROS.2013.6696514. 3.2.1

[32] Ross Hartley, Maani Ghaffari Jadidi, Jessy W. Grizzle, and Ryan M. Eustice. Contact-Aided Invariant Extended Kalman Filtering for Legged Robot State Estimation. *Robotics: Science and Systems (RSS)*, 2018. URL http://arxiv.org/abs/1805.10410. 2.3

[33] Sejong Heo and Chan Gook Park. Consistent EKF-Based Visual-Inertial Odometry on Matrix Lie Group. *IEEE Sensors Journal*, 18(9):3780–3788, 2018. ISSN 1530437X. doi: 10.1109/JSEN.2018.2808330. 2.3

[34] Joel A. Hesch, Dimitrios G. Kottas, Sean L. Bowman, and Stergios I. Roumeliotis. Consistency analysis and improvement of vision-aided inertial navigation. *IEEE Trans. on Robotics (TRO)*, 30(1):158–176, 2014. ISSN 15523098. doi: 10.1109/TRO.2013.2277549. 2.1, 5

[35] G. Huang, M. Kaess, and J.J. Leonard. Consistent sparsification for graph optimization. In *European Conference on Mobile Robots (ECMR)*, pages 150–157, Barcelona, Spain, September 2013. 2.2

[36] G.P. Huang, A.I. Mourikis, and S.I. Roumeliotis. An observability-constrained sliding window filter for SLAM. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 65–72, 2011. 1.2.2, 2.3, 4, 5, 5.1, 5.1.3

[37] Guoquan P. Huang and Stergios I. Roumeliotis. On filter consistency of discrete-time nonlinear systems with partial-state measurements. *Proc. American Control Conference (ACC)*, pages 5468–5475, 2013. ISSN 07431619. doi: 10.1109/ACC.2013.6580693. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6580693`. 2.1, 5, 5.1.2, 5.1.3

[38] Guoquan P. Huang, Anastasios I. Mourikis, and Stergios I. Roumeliotis. Observability-based rules for designing consistent EKF SLAM estimators. *Intl. J. of Robotics Research (IJRR)*, 29(5):502–528, 2010. ISSN 02783649. doi: 10.1177/0278364909353640. 2.1, 2.3, 5, 5.1.2, 5.1.3

[39] V Ila, J M Porta, and J Andrade-Cetto. Information-based compact pose SLAM. *IEEE Trans. on Robotics (TRO)*, 26(1):78–93, 2010. 2.2

[40] Viorela Ila, Lukas Polok, Marek Solony, and Pavel Svoboda. Slam++-a highly efficient and temporally scalable incremental slam framework. *Intl. J. of Robotics Research (IJRR)*, 36(2):210–230, 2017. doi: 10.1177/0278364917691110. URL `https://doi.org/10.1177/0278364917691110`. 2.1

[41] V. Indelman, S. Williams, M. Kaess, and F. Dellaert. Information fusion in navigation systems via factor graph based incremental smoothing. *Journal of Robotics and Autonomous Systems (RAS)*, 61(8):721–738, August 2013. 2.1

[42] M. Kaess, A. Ranganathan, and F. Dellaert. iSAM: Incremental smoothing and mapping. *IEEE Trans. on Robotics (TRO)*, 24(6):1365–1378, December 2008. 2.2

[43] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *Intl. J. of Robotics Research (IJRR)*, 31(2):216–235, February 2012. 2.1, 2.2

[44] Georg Klein and David Murray. Parallel tracking and mapping on a camera phone. *International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 83–86, 2009. ISSN 15534014. doi: 10.1109/ISMAR.2009.5336495. 2.1

[45] K Konolige, M Agrawal, and J Sola. Large scale visual odometry for rough terrain. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 201–212, 2007. ISBN 9783642147425. doi: 10.1007/978-3-642-14743-2_18. 2.1

[46] Vijay Kumar and Nathan Michael. Robotics Research. 100:41–58, 2017. doi: 10.1007/978-3-319-29363-9. URL `http://link.springer.com/10.1007/978-3-319-29363-9`. 1.1

[47] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011. 2.1, 2.2

[48] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Fur-

gale. Keyframe-based visualinertial odometry using nonlinear optimization. *Intl. J. of Robotics Research (IJRR)*, 34(3):314–334, 2015. ISSN 0278-3649. doi: 10.1177/0278364914554813. URL `http://journals.sagepub.com/doi/10.1177/0278364914554813`. 1.1, 1.2.1, 2.1, 2.1, 2.2, 3.3, 4.1.3, 4.3.3, 5

[49] M. Li and A.I. Mourikis. High-precision, consistent EKF-based visual-inertial odometry. *Intl. J. of Robotics Research (IJRR)*, 32(6):690–711, 2013. 2.1, 5, 5.1.2, 5.1.3

[50] A.W. Long, K.C. Wolfe, M. Mashner, and Gregory S. Chirikjian. The Banana Distribution is Gaussian: A Localization Study with Exponential Coordinates. *Robotics: Science and Systems (RSS)*, page 8, 2012. ISSN 2330765X. doi: 10.15607/RSS.2012.VIII.034. URL `http://roboticsproceedings.org/rss08/p34.pdf`. 2.3

[51] Mladen Mazuran, Wolfram Burgard, and Gian Diego Tipaldi. Nonlinear factor recovery for long-term SLAM. *Intl. J. of Robotics Research (IJRR)*, 35(1-3):50–72, 2016. ISSN 0278-3649. doi: 10.1177/0278364915581629. URL `http://journals.sagepub.com/doi/10.1177/0278364915581629`. 2.2, 4.2.2, 4.2.3, 4.2.3, 6.2

[52] A I Mourikis and S I Roumeliotis. A multi-state Kalman filter for vision-aided inertial navigation. *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (April):10–14, 2007. 2.1

[53] Raul Mur-Artal and Juan D. Tardos. Visual-Inertial Monocular SLAM with Map Reuse. 2016. ISSN 2377-3766. doi: 10.1109/LRA.2017.2653359. URL `http://arxiv.org/abs/1610.05949{%}0Ahttp://dx.doi.org/10.1109/LRA.2017.2653359`. 2.1

[54] Raúl Mur-Artal and Juan D Tardós. Visual-inertial monocular SLAM with map reuse. *IEEE Robotics and Automation Letters (RA-L)*, 2(2):796–803, 2017. 2.1

[55] Raul Mur-Artal, J M M Montiel, and Juan D Tardós. Orb-slam: a versatile and accurate monocular slam system. *IEEE Trans. on Robotics (TRO)*, 31(5):1147–1163, 2015. 2.1

[56] Marcin Odelga, Paolo Stegagno, Nicholas Kochanek, and Heinrich H Bülthoff. A Self-contained Teleoperated Quadrotor : On-board State-Estimation and Indoor Obstacle Avoidance. pages 7840–7847, 2018. 1.1

[57] L. Oth, P. Furgale, L. Kneip, and R. Siegwart. Rolling shutter camera calibration. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1360–1367, June 2013. doi: 10.1109/CVPR.2013.179. 3.2.1

[58] James A. Preiss, Wolfgang Honig, Gaurav S. Sukhatme, and Nora Ayanian. Crazyswarm: A large nano-quadcopter swarm. *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3299–3304, 2017. ISSN 10504729. doi: 10.1109/ICRA.2017.7989376. 1.1

[59] Tong Qin, Peiliang Li, and Shaojie Shen. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *arXiv preprint arXiv:1708.03852*, 2017. 1.1, 1.2.1, 2.1, 2.1, 2.2, 3.3, 4.1.3, 4.3.3, 5

[60] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart. Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4304–4311, May 2016. doi: 10.1109/ICRA.2016.7487628.

3.2.1

[61] M. Schmidt, E. van den Berg, M. P. Friedlander, and K. Murphy. Optimizing costly functions with simple constraints: A limited-memory projected quasi-newton algorithm. In *Proceedings of The Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 456–463, Clearwater Beach, Florida, April 2009. 4.2.3

[62] M. A. Skoglund, G. Hendeby, and D. Axehill. Extended Kalman filter modifications based on an optimization view point. In *Intl. Conf. on Information Fusion (FUSION)*, pages 1856–1861, July 2015. 2.1, 5

[63] Sebastian Thrun, Yufeng Liu, Daphne Koller, Andrew Y. Ng, Zoubin Ghahramani, and Hugh Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *Intl. J. of Robotics Research (IJRR)*, 23(7-8):693–716, 2004. doi: 10.1177/0278364904045479. URL `https://doi.org/10.1177/0278364904045479`. 2.2

[64] V. Usenko, J. Engel, J. Stückler, and D. Cremers. Direct visual-inertial odometry with stereo cameras. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 1885–1892, May 2016. doi: 10.1109/ICRA.2016.7487335. 2.1

[65] J. Vallv, J. Sol, and J. Andrade-Cetto. Graph slam sparsification with populated topologies using factor descent optimization. *IEEE Robotics and Automation Letters (RA-L)*, 3(2):1322–1329, April 2018. doi: 10.1109/LRA.2018.2798283. 4.2.3

[66] Lieven Vandenberghe, Stephen Boyd, and Shao-Po Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 19 (2):499–533, 1998. doi: 10.1137/S0895479896303430. URL `https://doi.org/10.1137/S0895479896303430`. 4.2.3

[67] Matthew R. Walter, Ryan M. Eustice, and John J. Leonard. Exactly sparse extended information filters for feature-based SLAM. *Intl. J. of Robotics Research (IJRR)*, 26(4):335–359, 2007. ISSN 0278-3649. doi: 10.1177/0278364906075026. 2.2

[68] Y. Wang, R. Xiong, Q. Li, and S. Huang. Kullback-Leibler divergence based graph pruning in robotic feature mapping. In *European Conference on Mobile Robots (ECMR)*, pages 32–37, Sept 2013. doi: 10.1109/ECMR.2013.6698816. 2.2

[69] S. Weiss, M. W. Achtelik, S. Lynen, M. Chli, and R. Siegwart. Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 957–964, May 2012. doi: 10.1109/ICRA.2012.6225147. 2.1

[70] Kejian J. Wu, Ahmed M. Ahmed, Georgios A. Georgiou, and Stergios I. Roumeliotis. A square root inverse filter for efficient vision-aided inertial navigation on mobile devices. *Robotics: Science and Systems (RSS)*, 2015. ISSN 2330765X. doi: 10.15607/RSS.2015.XI.008. 2.1

[71] Yulin Yang, James Maley, and Guoquan Huang. Null-space-based marginalization: Analysis and algorithm. *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, (September): 6749–6755, 2017. doi: 10.1109/IROS.2017.8206592. URL `http://ieeexplore.`

ieee.org/document/8206592/. 2.2, 4.1.2

[72] Teng Zhang, Kanzhi Wu, Jingwei Song, Shoudong Huang, and Gamini Dissanayake. Convergence and Consistency Analysis for A 3D Invariant-EKF SLAM. pages 1–8, 2017. ISSN 2377-3766. doi: 10.1109/LRA.2017.2651376. URL http://arxiv.org/abs/1702.06680. 2.3

A.1 Derivation of IMU Noise Model

This section explains how we could extract the noise corresponding to the noise model shown in (5.23). We utilize the first order approximation of SO(3) explained in Section 3.4.3.
Rotation $\Delta R'_{ij}$:

$$\begin{aligned}
\Delta R'_{ij} =& \prod_{k=i}^{j-1} \exp((\tilde{w}_k - \mathbf{b}_k^g - \eta_k^{gd})\Delta t) \\
\simeq& \prod_{k=i}^{j-1} \left[\exp((\tilde{w}_k - \mathbf{b}_k^g)\Delta t) \exp(-J_r^k \eta_k^{gd} \Delta t) \right] \\
=& \Delta\tilde{R}'_{ij} \prod_{k=i}^{j-1} \exp(-\Delta R_{k+1j} \top J_r^k \eta_k^{gd} \Delta t) \\
=& \boxed{\Delta\tilde{R}'_{ij} \exp(-\delta\phi_{ij})}
\end{aligned} \tag{1}$$

Velocity $\Delta v'_{ij}$:

$$\begin{aligned}
\Delta v'_{ij} =& \sum_{k=i}^{j-1} \Delta R_{ik}(\tilde{a}_k - \mathbf{b}_k^a - \eta_k^{ad})\Delta t \\
\simeq& \sum_{k=i}^{j-1} \left[\Delta\tilde{R}'_{ik}(I - \delta\phi_{ik})^\wedge(\tilde{a}_k - \mathbf{b}_i^a)\Delta t - \Delta\tilde{R}_{ik}\eta_k^{ad}\Delta t \right] \\
=& \Delta\tilde{v}'_{ij} + \sum_{k=i}^{j-1} \left[\Delta\tilde{R}'_{ik}(\tilde{a}_k - \mathbf{b}_i^a)^\wedge \delta\phi_{ik}\Delta t - \Delta\tilde{R}_{ik}\eta_k^{ad}\Delta t \right] \\
=& \Delta\tilde{v}'_{ij} + \\
& (\Delta R'_{ij} \Delta R'_{ij} \top) \sum_{k=i}^{j-1} \left[\Delta\tilde{R}'_{ik}(\tilde{a}_k - \mathbf{b}_i^a)^\wedge \delta\phi_{ik}\Delta t - \Delta\tilde{R}_{ik}\eta_k^{ad}\Delta t \right] \\
=& \Delta\tilde{v}'_{ij} + \Delta R'_{ij} \sum_{k=i}^{j-1} \left[\Delta\tilde{R}'_{jk}(\tilde{a}_k - \mathbf{b}_i^a)^\wedge \delta\phi_{ik}\Delta t - \Delta\tilde{R}_{jk}\eta_k^{ad}\Delta t \right] \\
=& \boxed{\Delta\tilde{v_{ij}}' - \Delta R'_{ij}\delta v_{ij}}
\end{aligned} \tag{2}$$

Position $\Delta p'_{ij}$:

$$\begin{aligned}
\Delta p'_{ij} =& \sum_{k=i}^{j-1} \big[\Delta v'_{ik}\Delta t + \frac{1}{2}\Delta R_{ik}(\tilde{a_k} - \mathbf{b}^a_k - \eta^{ad}_k)\Delta t^2\big] \\
\simeq& \sum_{k=i}^{j-1} \big[(\Delta \tilde{v}'_{ik} - \Delta R'_{ik}\delta v_{ik})\Delta t + \\
& \frac{1}{2}\Delta \tilde{R}'_{ij}(I - \delta\phi^{\wedge}_{ik})(\tilde{a}_k - \mathbf{b}^a_i)\Delta t^2 - \frac{1}{2}\Delta \tilde{R}_{ik}\eta^{ad}_k\Delta t^2\big] \\
=& \Delta \tilde{p}'_{ij} + \sum_{k=i}^{j-1} \big[- \Delta R'_{ik}\delta v_{ik}\Delta t + \frac{1}{2}\Delta \tilde{R}'_{ik}(\tilde{a}_k - \mathbf{b}^a_i)^{\wedge}\delta\phi_{ik}\Delta t^2\big] \\
=& \Delta \tilde{p}'_{ij} + (\Delta R'_{ij}\Delta R'^{\top}_{ij}) \\
& \sum_{k=i}^{j-1} \big[- \Delta R'_{ik}\delta v_{ik}\Delta t + \frac{1}{2}\Delta \tilde{R}'_{ik}(\tilde{a}_k - \mathbf{b}^a_i)^{\wedge}\delta\phi_{ik}\Delta t^2\big] \\
=& \Delta \tilde{p}'_{ij} + \Delta R'_{ij} \sum_{k=i}^{j-1} \big[- \Delta \tilde{R}'_{jk}\delta v_{ik}\Delta t + \frac{1}{2}\Delta \tilde{R}'_{jk}(\tilde{a}_k - \mathbf{b}^a_i)^{\wedge}\delta\phi_{ik}\Delta t^2\big] \\
=& \boxed{\Delta \tilde{p}'_{ij} - \Delta R'_{ij}\delta p_{ij}}
\end{aligned} \tag{3}$$

B.2 Derivation of IMU Bias Partial Derivatives

This section details the derivation of partial derivatives for the first-order bias model.
Rotation:

$$\begin{aligned}
\frac{\partial r_{R_iR_j}}{\partial \mathbf{b}^g_i} &= \log([\mathbf{R}_i\Delta\tilde{R}_{ij}\exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}(\delta\mathbf{b}^g_i + \delta\delta\mathbf{b}^g_i))]^{\top}\mathbf{R}_j) \\
&= \log([\mathbf{R}_i\Delta\tilde{R}_{ij}(\exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}^g_i))(\exp(J^k_r(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}^g_i)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}^g_i))]^{\top}\mathbf{R}_j) \\
&= \log(\exp(J^k_r(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}^g_i)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}^g_i)^{\top}(\mathbf{R}_i\Delta\tilde{R}_{ij}\exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}^g_i))^{\top}\mathbf{R}_j) \\
&= \log(\exp(-J^k_r(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}^g_i)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}^g_i)\exp(r_{R_iR_j})) \\
&= \log(\exp(r_{R_iR_j})\exp(-\exp(r_{R_iR_j})^{\top}J^k_r(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}^g_i)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}^g_i)) \\
&= r_{R_iR_j}\boxed{-J^{-1}_r(r_{R_iR_j})\exp(r_{R_iR_j})^{\top}J^k_r(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}^g_i)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}}\delta\delta\mathbf{b}^g_i
\end{aligned} \tag{4}$$

Velocity:

$$\begin{aligned}
\frac{\partial r_{v_iv_j}}{\partial \mathbf{b}_i^g} &= \mathbf{R}_i\Delta\tilde{R}_{ij}\exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}(\delta\mathbf{b}_i^g+\delta\delta\mathbf{b}_i^g))^\top(\mathbf{v}_j-\phi_i(\mathbf{v}_i)-\mathbf{R}_i\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)\\
&= \exp(J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)^\top(\mathbf{R}_i\Delta\tilde{R}_{ij}\exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g))^\top(\mathbf{v}_j-\phi_i(\mathbf{v}_i)-\mathbf{R}_i\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)\\
&= \exp(J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)^\top\phi_i(\mathbf{R}_i)^\top(\mathbf{v}_j-\phi_i(\mathbf{v}_i)-\mathbf{R}_i\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)\\
&= \exp(J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)^\top[r_{v_iv_j}-[\Delta\tilde{R}_{ij}\exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)]^\top\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g]
\end{aligned}$$

Let $A =[\Delta\tilde{R}_{ij}\exp(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)]$.

$$\begin{aligned}
&\approx (I-(J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)^\wedge)[r_{v_iv_j}-A^\top\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g]\\
&= r_{v_iv_j}-A^\top\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g-(J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g)^\wedge r_{v_iv_j}\\
&= r_{v_iv_j}-A^\top\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g+r_{v_iv_j}^\wedge J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\delta\mathbf{b}_i^g\\
&= r_{v_iv_j}+\boxed{\big[r_{v_iv_j}^\wedge J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}-A^\top\frac{\partial\Delta\bar{v}_{ij}}{\partial b^g}\big]}\delta\delta\mathbf{b}_i^g
\end{aligned} \tag{5}$$

$$\begin{aligned}
\frac{\partial r_{v_iv_j}}{\partial \mathbf{b}_i^a} &= \phi_i(\mathbf{R}_i)^\top(\mathbf{v}_j-\phi_i(\mathbf{v}_i)-\mathbf{R}_i\frac{\partial\Delta\bar{v}_{ij}}{\partial b^a}\delta\delta\mathbf{b}_i^a)\\
&= r_{v_iv_j}\boxed{-A^\top\frac{\partial\Delta\bar{v}_{ij}}{\partial b^a}}\delta\delta\mathbf{b}_i^a
\end{aligned} \tag{6}$$

Position:

$$\frac{\partial r_{p_ip_j}}{\partial \mathbf{b}_i^g} = r_{p_ip_j}+\boxed{\big[r_{p_ip_j}^\wedge J_r^k(\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}\delta\mathbf{b}_i^g)\frac{\partial\Delta\bar{R}_{ij}}{\partial b^g}-A^\top\frac{\partial\Delta\bar{p}_{ij}}{\partial b^g}\big]}\delta\delta\mathbf{b}_i^g \tag{7}$$

$$\frac{\partial r_{p_ip_j}}{\partial \mathbf{b}_i^a} = r_{p_ip_j}\boxed{-A^\top\frac{\partial\Delta\bar{p}_{ij}}{\partial b^a}}\delta\delta\mathbf{b}_i^a \tag{8}$$