TOWARDS ROBUST VISUAL-INERTIAL ESTIMATION

by

Kevin Eckenhoff

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Mechanical Engineering

Spring 2020

TOWARDS ROBUST VISUAL-INERTIAL ESTIMATION

by

Kevin Eckenhoff

Approved: _____

Ajay Prasad, Ph.D.
Chair of the Department of Mechanical Engineering

Approved: _____

Levi Thompson, Ph.D.
Dean of the College of Engineering

Approved: _____

Douglas J. Doren, Ph.D.
Interim Vice Provost for Graduate and Professional Education and
Dean of the Graduate College

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Guoquan Huang, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Herbert Tanner, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Ioannis Poulakakis, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

John Leonard, Ph.D.
Member of dissertation committee

ACKNOWLEDGEMENTS

To my wife Molly: For all my work in localization, I'd still be lost without you

None of the work presented in this thesis would be possible without the many people in my life who have supported me throughout the years. I would to extend my gratitude first and foremost to my advisor, Dr. Guoquan (Paul) Huang, whose tireless efforts molded me from a inexperienced mechanical engineer lacking almost any relevant background into a (hopefully) proper SLAM scientist. His ambition and drive have taught me valuable lessons in expecting the absolute best from myself, and I owe all my success to his guidance.

I would like to thank all my teachers at UD, including committee members Dr. Ioannis Poulakakis and Dr. Herbert Tanner, who I was able to learn from in both my undergraduate and graduate studies. I also extend my gratitude to my supervisor during my internship at Facebook Reality Labs, Dr. Anastasios Mourikis, who taught me more than I thought possible during the few months I was with him.

I would like to thank the NSF, DTRA, and Google Daydream for providing funding for my work. In addition I am indebted to David Helwig for his Fellowship which has supported me throughout my graduate career.

I would like to thank all the members of the Robot Perception and Navigation Group (RPNG) here at UD, including Patrick, Yulin, Zheng, Woosik, Jesse, and Nate. We spent many years discussing, arguing, and learning together, and have helped/pushed each other to achieve things we never could have alone. I would like to especially thank my near-constant collaborator, Patrick, and I will miss what a prolific

and successful team we had become. Besides RPNG, I would also like to thank all my friends and colleagues at UD who have made my journey here enjoyable, including fellow graduate students Sushant, Indrajeet, Rohit and Adam. I would like to thank all my friends Adam, Mike, Connor, Danny, Brook, Jack, Taylor, Ian, and Scott for all the fun we had when I wasn't too busy working.

Finally, I would like to thank my family for all the encouragement and support they have offered me over the years. I would especially like to extend my gratitude to my late grandfather, Dr. Ralph Pschunder, who has inspired me ever since I was a child, and whose nameplate I have kept on my desk everyday as a source of motivation. Lastly, and most importantly, I would like to thank my wife, Molly, who has been there for me through all the joy and all the frustration of this journey. Without her, none of this would be possible.

TABLE OF CONTENTS

Appendix

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Visual-inertial navigation systems (VINS), which fuse the information from cameras and inertial measurement units (IMUs) to recover an estimate for the motion of a moving sensing platform, have seen an explosion in popularity in recent years. This is especially apparent in domains where cost, payload, and computational resources are heavily constrained, such as mobile devices and micro aerial vehicles (MAVs). While a standard VINS paradigm assumes a single IMU-camera (monocular or stereo) pair operating in an unknown but *static* environment, in this thesis, we provide three main thrusts which seek to improve overall estimation performance and to generalize these assumptions for building more robust visual-inertial systems.

In the first thrust, we focus on improving the standard VINS paradigm (a single IMU-camera pair operating in a static environment). In particular, we seek to improve the accuracy of incorporating inertial measurements into a graph-based (i.e., batch-optimization) VINS when utilizing IMU preintegration. We provide two models for the evolution of the inertial measurements between sample intervals which yield closed-form but highly accurate solutions to the preintegration equations. We then show that these models offer improved accuracy over the standard, discrete preintegration methods, and incorporate the proposed models into both indirect (feature-based) and direct (intensity-based) visual-inertial systems.

In the second thrust, we incorporate an arbitrary number of "plug-and-play" sensors into VINS. In particular, we first consider utilizing the information from multiple, non-overlapping, *asynchronous* (i.e., each camera is triggered independently) cameras, to allow for robustness to poorly textured regions along certain viewing directions and parallelizability of feature tracking for each camera. The fact that this system does

not require hardware-synchronized sensors allows for easy addition of auxiliary cameras. Naively, due to the measurements being collected at multiple rates, estimation of the pose of the system at *each* measurement time could be performed to fuse the information, yielding an increased computational burden. To combat this, and to allow for real-time estimation, we introduce a linear-interpolation scheme which allows for the estimation of poses at a reduced rate, and utilize this model to perform calibration of both the spatial (relative pose) and temporal (time offset) relationships between each of the involved sensors within the multi-state constraint Kalman Filter (MSKCF) framework. We show in real-world experiments that utilizing additional cameras yields improved VINS performance, and demonstrate that our system is able to accurately calibrate the sensor suite from poor initial guesses.

Considering the fact that even a sensor suite with many cameras may experience measurement depletion due to poor lighting conditions or fully textureless scenarios, we further design a new MSCKF-based estimator to incorporate an arbitrary number of IMUs into its formulation. This allows for an improvement in visual-inertial estimation which does not heavily rely on the environmental conditions, while additionally robustifying the system to failure of a single IMU (due to physical disconnection, high temperatures, or certain vibrational frequencies). In particular, we maintain an estimate for each IMU's state (pose, velocity, and sensor biases). During the prediction phase of the filter, the state of each IMU, as well as the joint covariance of the combined system, is propagated forward. We then derive a novel spatial-temporal constraint on the relative pose between each IMU which acts as an updating measurement for the filter, and allows for the fusion of each IMU's information as well as spatial and temporal calibration between all sensors. Due to the fact that this system can operate as long as one IMU remains active, this makes our method robust to IMU sensor failure. We then show in simulated and real-world experiments that the proposed method offers improved localization performance over a single-IMU system, while additionally being resilient to IMU failures. Finally, based on these two systems, a multi-IMU multi-camera (MiMc)-VINS is developed to fuse all available information from any number

of IMUs and cameras, and is validated in simulations and experiments.

In the third thrust, we focus on the second assumption, namely that the sensor platform operates in a static environment. As the world is often truly dynamic, improper modeling of the surroundings may lead to catastrophic errors as the estimator misinterprets motion of the environment as its own ego-motion. To relax this assumption, we allow for some parts of the environment (i.e., 3D feature points detected by the camera) to remain static, while others act as moving rigid-body objects which follow a motion model described by some estimated motion parameters. Estimating the motion of such objects not only provides more information to the filter, but may even be the *objective* of the sensing platform (e.g., a MAV performing target tracking of another vehicle).

To perform this estimation, we include the pose, motion parameters and local point cloud of the moving object as state variables to be estimated by the filter, along with the standard VINS navigation states. The treatment of the object as a moving point cloud provides our system robustness to viewing the target from multiple directions, while the incorporation of a motion model allows for prediction of the future target state after loss of sight, which is a required component in active tracking scenarios. Several possible motion models are offered which capture many real-world target scenarios seen in practice, and an extensive observability analysis for each model is performed. We then show in both simulated and real-world experiments that our system provides accurate localization and object tracking performance, even when losing sight of the target or viewing the object from multiple directions. Lastly but of practical importance, we investigate in-depth the effect of *incorrect* model selection during tightly-coupled estimation, and design a Schmidt Kalman Filter based estimator to prevent inconsistent motion models from corrupting VINS performance while still properly tracking all correlations.

Chapter 1

INTRODUCTION

1.1 Motivation and Challenges

Accurate localization for autonomous systems is a prerequisite in many robotic applications such as planetary exploration [92], search and rescue [37], and autonomous driving [46]. In many of these scenarios, access to global information such as from a Global Positioning System (GPS), motion capture system, or a prior map of the environment is unavailable. Instead, one typically estimates the robot state and its surroundings based on noisy, local measurements from onboard sensors, by performing simultaneous localization and mapping (SLAM), which has witnessed significant research efforts in the past three decades [15].

Of many possible sensors used in SLAM, micro-electro-mechanical-system (MEMS) inertial measurement units (IMUs) have become ubiquitous. These low-cost and light-weight sensors typically provide local linear acceleration and angular velocity readings, and are well suited for many applications such as micro aerial vehicles (MAVs) [85] and mobile devices [125]. IMUs provide information only about the derivatives of the kinematic states, so estimation must be performed by integrating over these noisy measurements. This may lead to large drifts over long periods of time, making the use of a low-cost IMU alone an unreliable solution. However, IMU readings are highly-informative about short-term motion which is ideal for fusion with measurements from exteroceptive aiding sensors, such as LiDAR and cameras. These sensors compensate for the drift inherent in inertial navigation, while high-rate inertial measurements allow for the tracking of aggressive motion which may be difficult for exteroceptive low-rate sensors alone. For these reasons, fusing the information provided by these sensors to

formulate VINS has become very popular [49, 75, 94, 103]. Despite the vast amount of effort in this direction, developing computationally efficient and robust VINS remains challenging.

In particular, improving the accuracy of VINS (and SLAM in general) remains crucial to wide-deployment of these systems in real world applications. This accuracy is particularly critical, for instance, in obstacle avoidance scenarios, where errors on the order of even a few centimeters may lead to collision/failure. However, most naive avenues for accuracy improvement, such as performing full bundle adjustment [74] across a large set of measurements, lead to an increased computational burden due the large problem size, thereby limiting application for scenarios which require low-latency estimation on resource-constrained devices [117]. As such, providing more accurate solutions to VINS without increasing complexity is a sought-after goal of many researchers.

Many VINS focus on the minimal sensing capabilities of a single IMU-camera pair in order to achieve the lowest computational cost [103]. However, such single-IMU, monocular systems are known to have less stability in the presence of certain maneuvers such as hovering [72] as well as lower accuracy than their multi-sensor counterparts [102]. In addition, lack of redundancy makes the systems sensitive to issues in the data from single sensors. For instance, many vision-based estimators which utilize only a single viewing direction (even when using stereo cameras) experience large drifts in scenarios that lack texture in that direction [38] due to the inability to reliably extract/track features on the image plane. Lastly, *failure* of a given sensor (due to events such as physical disconnection) may be catastrophic for the full system unless redundant back-ups are utilized. As such, we are motivated to build robust and highly accurate systems capable of using the information from an arbitrary number of IMUs and cameras.

Most SLAM systems, including VINS, make a strong assumption that the environment remains *static*. However in most practical scenarios the world is highly

dynamic, and thus improper modeling of the environment may lead to large estimation errors. In addition, perception of moving objects is a required component for applications such as autonomous driving [122], or may be the overall goal of the sensor deployment such as in active tracking [19]. In either case, the ability for VINS to track the motion of external bodies remains highly desirable in practice.

1.2 Contributions

Based on the precise formulations and rigorous solutions of visual-inertial estimation, in this thesis, we address the aforementioned key challenges arising in visual-inertial navigation. In particular, we present three main contributions, each seeking to improve a certain aspect of current VINS algorithms.

1.2.1 Closed-Form Preintegration Methods

In the first chapter, we seek to improve the accuracy of incorporating IMU measurements into graph-based SLAM, which is typically achieved through preintegration [88]. In particular, instead of relying on typical discrete approximation schemes used to perform standard preintegration, we offer two models for the evolution of the measurements between IMU sampling intervals. These models yield closed-form but highly-accurate solutions to the preintegration equations, which we show offers improved accuracy over the state-of-the-art discrete method [43]. Leveraging these highly-accurate preintegration schemes, we design both a sliding-window indirect (feature based) and a full-batch direct (pixel-intensity based) visual-inertial system. We then show in extensive real-world validation that this achieves competitive accuracy to a state-of-the-art VINS [75].

1.2.2 Multi-IMU Multi-Camera VINS

In order to build more robust and efficient VINS, in the second chapter we modify a state-of-the-art filtering approach [94] to allow for an arbitrary number of IMUs and cameras for simple "plug-and-play" operation. In particular, we present an efficient pose-interpolation scheme which allows for the fusion of information from

an arbitrary number of cameras while limiting the increased computational burden on the estimator. In addition, we allow for an arbitrary number of IMU by stacking the state of each IMU in a joint system. We enforce a relative pose-constraint between the rigidly connected IMUs which acts as an update to the estimator, thereby fusing all information together in a tightly-coupled fashion.

As our overall goal is to enable the use of modular, low-cost devices, we perform online spatial (relative-pose) and temporal (time offset) calibration between *all* sensors, while additionally refining the intrinsics of each camera. This prevents the need for tedious offline calibration, facilitating quick deployment of VINS systems. We validate both the consistency and calibration performance of the proposed system in both real and simulated scenarios, and show that the inclusion of additional sensors greatly improves the performance of VINS, while additionally creating resilience to *sensor failures*.

1.2.3 Visual-Inertial Localization and Target Tracking

In the third and final chapter, we further modify the state-of-the-art filtering approach [94] to additionally estimate the state of an external, moving target seen in its cameras. We treat the target as a local point cloud rigidly attached to a moving pose, whose evolution is governed by some underlying, estimated motion parameters. By jointly estimating both the navigation and target states, we can improve the accuracy of both the VINS and tracked target estimates. To capture common targets seen in practice, we offer three possible motion models which govern the evolution of the target, and the proposed system is validated in both real-world and simulated experiments.

We additionally investigate the effect of inconsistent target motion models, and show that while tightly-coupled estimation leads to improved performance in the case that a "proper" model is used, inconsistent modeling of the target's motion degrades estimation performance. To account for this scenario, we propose a target tracking system based on the Schmidt-Kalman Filter (SKF) [112], such that we do not use

target measurements to update the estimates of the navigation states, while guaranteeing that all correlations are still properly (conservatively) tracked. We show in both real and simulated experiments that this, along with a robot-centric representation of the target's pose, leads to accurate and robust estimation even in the presence of inconsistent target models.

1.3 Literature Survey

In this section we review the overall literature related to visual-inertial navigation. We note that for each chapter we present a more specific survey, in order to properly capture the works relevant to each thrust.

1.3.1 Visual-Inertial Navigation

Mourikis and Roumeliotis [94] proposed one of the earliest successful VINS algorithms, known as the multi-state constraint Kalman filter (MSCKF). This Extended Kalman Filtering (EKF) approach used quaternion-based inertial dynamics [118] for state propagation coupled with a novel EKF update step. Rather than adding features seen in the camera images to the state vector, their visual measurements were projected onto the nullspace of the feature Jacobian matrix (akin to feature marginalization [129]), thereby retaining motion constraints that only related to the stochastically cloned camera poses in the state vector [110]. This nullspace projection reduced the computational cost by removing the need to co-estimate features, but prevented the relinearization of the processed features' nonlinear measurements at later time steps.

The standard MSCKF recently has been extended in various directions. For example, both Hesch et al. [58] and Huang et al. [63] improved the filter consistency by enforcing the correct observability properties of the linearized EKF VINS. Guo et al. [51] showed that the inclusion of plane features increases the estimation accuracy. Guo et al. [53] extended to the case of rolling-shutter cameras with inaccurate time synchronization. Recently, Wu et al. [125] further reformulated the VINS problem within a square-root inverse filtering framework for improved computational efficiency

and numerical stability without sacrificing estimation accuracy. Overall, the low computational cost of these filtering-based methods makes them an ideal choice for devices with extreme resource-constraints and low-latency requirements [78, 117]. While these MSCKF-based methods have been shown to exhibit extremely fast and fairly accurate state estimation, they theoretically suffer from a limitation – that is, nonlinear measurements must have a *one-time* linearization before processing, possibly introducing large linearization errors into the estimator. To handle these nonlinearities, Brossard et al. [13] utilized invariant Unscented Kalman Filtering within VINS, at an increased computational cost. Bloesch et al. [9] introduced an indirect VINS that improves the robustness to nonlinearity through an *iterative* EKF. Kottas et al. [73] similarly investigated an iterative Kalman Smoother for indirect VINS.

Batch optimization methods, on the other hand, solve a nonlinear least-squares or bundle adjustment (BA) problem over a set of measurements at each timestep, allowing for the reduction of error through relinearization [74]. The incorporation of *tightly-coupled* VINS in batch optimization methods requires overcoming the high frequency nature and computational complexity of inertial measurements. Leutenegger et al. [75] introduced one of the first keyframe BA VINS approaches (i.e., OKVIS), whereby a set of non-sequential past camera poses and a series of recent inertial states, connected with inertial measurements, was used in nonlinear optimization for accurate trajectory estimation. These inertial factors took the form of a state prediction: every time that the linearization point for the starting inertial state changed, reintegration of the IMU dynamics is required. This presents inefficiencies in the inertial processing, while the authors demonstrated the feasibility of such a scheme for a small number of inertial factors in a sliding window estimator. The work of Guo et al. [52] performed full visual-inertial batch optimization using the data of *multiple* users, and improve efficiency by only reintegrating the propagation means, while keeping the corresponding covariance fixed. First introduced by Lupton et al [88], inertial preintegration has facilitated numerous batch-based methods [43, 86, 103, 120]. In this method IMU measurements are integrated in a local frame *independent* of the current state estimates,

such that reintegration does not need to be performed if the initial state's linearization point changes. Forster et al. [43] presented the state-of-the-art preintegration which performs integration of the orientation on the SO(3) manifold for improved stability.

1.3.2 Visual Processing

A key component to any VINS algorithm is the visual processing pipeline, responsible for transforming dense imagery data to motion constraints that can be incorporated into the estimation problem. Seen as the classical technique, indirect methods of visual SLAM extract and track sparse features in the environment, while using geometric reprojection constraints during estimation. An example of state-of-the-art indirect visual-SLAM methods is ORB-SLAM2 [96], which performs graph-based optimization of camera poses using information from 3D feature point correspondences.

In contrast, direct methods utilize pixel intensities in their formulation and allow for inclusion of a larger percentage of the available image information. LSD-SLAM [39] is a state-of-the-art direct visual-SLAM method which optimizes the transformation between pairs of camera keyframes based on minimizing their intensity error. Note that this approach also optimizes a separate, secondary graph containing keyframe constraints to allow for the incorporation of highly informative loop-closures to correct drift over long trajectories. This work was later extended from a monocular sensor to stereo and omnidirectional cameras for improved accuracy [16, 40]. Other popular direct methods include the work by Engel et al. [38] and Wang et al. [123] which estimate a sparse set of pixel depths associated with keyframes along with camera poses in a tightly-coupled manner, offering low-drift performance. This was later extended to include inertial measurements in the batch optimization [121].

Application of direct methods to the visual-inertial problem has seen recent attention due to their ability to robustly track dynamic motion even in low-texture environments. For example, Bloesch et al. [9, 10] used a patch-based direct method to provide updates with an iterated EKF; Usenko et al. [119] introduced a sliding-window VINS based on the discrete preintegration and direct image alignment; Ling

et al. [85] employed loosely-coupled direct alignment with preintegration factors for tracking aggressive quadrotor motions.

1.4 Notation

Here we review some of the notations utilized in this work. Lowercase, bold lowercase, and bold uppercase (e.g., $t, \mathbf{t}$, and $\mathbf{T}$) represent scalars, vectors, and matrices respectively. $\bar{q}$ are used for quaternions, $\lfloor \mathbf{v} \times \rfloor$ is the skew-symmetric matrix built from the 3×1 vector $\mathbf{v}$, with inverse operation $\lfloor \mathbf{v} \times \rfloor^{\vee} = \mathbf{v}$. We use $\mathbf{n} \sim \mathcal{N}(\mathbf{a}, \mathbf{P})$ to denote that $\mathbf{n}$ is distributed as a Gaussian random variable with mean $\mathbf{a}$ and covariance $\mathbf{P}$. We use $\hat{\mathbf{a}}$ to represent the estimate for variable $\mathbf{a}$ which has error $\tilde{\mathbf{a}}$, with the expectation operator $\mathbb{E}$ such that $\mathbb{E}[\mathbf{a}] = \hat{\mathbf{a}}$. We use ${}^{A}_{B}\mathbf{R}$ to represent the rotation matrix which rotates vectors from frame B into frame A, while ${}^{A}\mathbf{p}_{B}$ is the position of frame B's origin in frame A.

Chapter 2

ESTIMATION BACKGROUND

In the section we review some of the fundamental estimation theory utilized in this thesis. Consider an unknown state (variable), $\mathbf{x}$, which evolves over timesteps $[t_k, t_{k+1}]$ according to the following propagation function:

$$\mathbf{x}_{k+1} = \mathbf{g}\left(\mathbf{x}_k, \mathcal{I}, \mathbf{n}_k\right) \tag{2.1}$$

where $\mathcal{I}$ is the set of proprioceptive measurements (such as wheel odometry, IMU measurements, etc.), while $\mathbf{n}_k \sim \mathcal{N}\left(\mathbf{0}, \mathbf{Q}_k\right)$ is the zero-mean, discrete-time Gaussian noise corrupting the measurements. As an example, in the visual-inertial domain, this propagation function takes the form of integrating the IMU readings to predict the state evolution.

At timestep $k+1$, we also collect several measurements, $\mathbf{z}_{k+1}$ which involve the value of the state at the corresponding step, $\mathbf{x}_{k+1}$:

$$\mathbf{z}_{k+1} = \mathbf{h}_{k+1}\left(\mathbf{x}_{k+1}\right) + \mathbf{n}_{k+1} \tag{2.2}$$

where $\mathbf{h}_{k+1}\left(\cdot\right)$ is the generating measurement function, while $\mathbf{n}_{k+1}$ is the measurement noise with covariance $\mathbf{R}_{k+1}$. In visual-inertial navigation, this measurement typically takes the form of pixel projections collected by the the camera corresponding to 3D features in the environment. The goal is to probabilistically fuse the propagation and measurement information in order to recover an estimate for the state, $\hat{\mathbf{x}}$, which serves as our best guess for the true value of the variable given the noisy measurements.

Note that in many applications, the state lies on a manifold rather than a vector space. In order to represent the estimation problem on a manifold, we employ

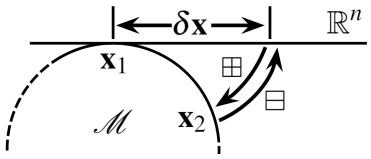


Figure 2.1: Illustration of the state update operations on a manifold. The $\boxplus$ operation maps $\mathbf{x}_1 \in \mathcal{M}$ and a vector $\tilde{\mathbf{x}} \in \mathbb{R}^n$ to a new element $\mathbf{x}_2 \in \mathcal{M}$, while the $\boxminus$ operation maps $\mathbf{x}_1$ and $\mathbf{x}_2$ to the vector $\tilde{\mathbf{x}}$.

the "boxplus" update operation, $\boxplus$, which maps an element from a manifold, $\mathbf{x} \in \mathcal{M}$, and an error vector $\tilde{\mathbf{x}}$ into a new element on $\mathcal{M}$ [56]. As illustrated in Figure 2.1, for a manifold of dimension n, we can define the following operation:

$$\boxplus : \ \mathcal{M} \times \mathbb{R}^n \to \mathcal{M} \tag{2.3}$$

$$\mathbf{x}_1 \boxplus \tilde{\mathbf{x}} = \mathbf{x}_2 \tag{2.4}$$

Similarly, the inverse "boxminus" operation $\boxminus$ is given by:

$$\boxminus : \ \mathcal{M} \times \mathcal{M} \to \mathbb{R}^n \tag{2.5}$$

$$\mathbf{x}_2 \boxminus \mathbf{x}_1 = \tilde{\mathbf{x}} \tag{2.6}$$

In the case of a state in a vector space, $\mathbf{v} \in \mathbb{R}^n$, these operations are the standard addition and subtraction:

$$\mathbf{v}_1 \boxplus \tilde{\mathbf{v}} \triangleq \mathbf{v}_1 + \tilde{\mathbf{v}} \ = \mathbf{v}_2 \tag{2.7}$$

$$\mathbf{v}_2 \boxminus \mathbf{v}_1 \triangleq \mathbf{v}_2 - \mathbf{v}_1 = \tilde{\mathbf{v}} \tag{2.8}$$

In the case of a unit quaternion expressed using the JPL convention, $\bar{q}$, we have [118]:

$$\bar{q}_1 \boxplus \tilde{\boldsymbol{\theta}} \triangleq \begin{bmatrix} \frac{\tilde{\boldsymbol{\theta}}}{2} \\ 1 \end{bmatrix} \otimes \bar{q}_1 \simeq \bar{q}_2 \tag{2.9}$$

$$\bar{q}_2 \boxminus \bar{q}_1 \triangleq 2\mathbf{vec}\left(\bar{q}_2 \otimes \bar{q}_1^{-1}\right) = \tilde{\boldsymbol{\theta}} \tag{2.10}$$

where $\mathbf{vec}\left(\bar{q}\right)$ refers to the vector portion of the quaternion argument (i.e., $\mathbf{vec}([\mathbf{q}^\top q_4]^\top) = \mathbf{q}$). The quaternion multiplication, $\otimes$, is given by:

$$\bar{p} \otimes \bar{q} \triangleq \mathcal{R}\left(\bar{q}\right)\bar{p} = \mathcal{L}\left(\bar{p}\right)\bar{q} \tag{2.11}$$

$$\mathcal{R}\left(\bar{q}\right) = \begin{bmatrix} q_4\mathbf{I} + \lfloor \mathbf{q} \times \rfloor & \mathbf{q} \\ -\mathbf{q}^\top & q_4 \end{bmatrix} \tag{2.12}$$

$$\mathcal{L}\left(\bar{p}\right) = \begin{bmatrix} p_4\mathbf{I} - \lfloor \mathbf{p} \times \rfloor & \mathbf{p} \\ -\mathbf{p}^\top & p_4 \end{bmatrix} \tag{2.13}$$

where for $\mathbf{q} = [q_x \; q_y \; q_z]^\top$, the skew-symmetric matrix is given by:

$$\lfloor \mathbf{q} \times \rfloor = \begin{bmatrix} 0 & -q_z & q_y \\ q_z & 0 & -q_x \\ -q_y & q_x & 0 \end{bmatrix} \tag{2.14}$$

In state estimation, these operations allow us to model the state as being on a manifold while its corresponding *error state* is modeled as a Gaussian distribution in a vector space. In particular, the random variable $\mathbf{x}$ with mean value $\hat{\mathbf{x}}$ takes the form:

$$\mathbf{x} = \hat{\mathbf{x}} \boxplus \tilde{\mathbf{x}} \tag{2.15}$$

$$\tilde{\mathbf{x}} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{\Sigma}\right) \tag{2.16}$$

where $\mathbf{\Sigma}$ is the covariance of the zero-mean error state. As both the propagation and measurement functions are nonlinear, we will need to define linearizations through Taylor series expansions about the current estimates, which will require computing the Jacobians of the propagation function. For our propagation functions (2.1) we have:

$$\tilde{\mathbf{x}}_{k+1} \approx \mathbf{\Phi}\tilde{\mathbf{x}}_k + \mathbf{\Gamma}\mathbf{n}_k \tag{2.17}$$

$$\mathbf{\Phi} = \left.\frac{\partial\left(\mathbf{g}\left(\hat{\mathbf{x}}_k \boxplus \tilde{\mathbf{x}}_k, \mathcal{I}, \mathbf{0}\right) \boxminus \hat{\mathbf{x}}_{k+1}\right)}{\partial \tilde{\mathbf{x}}_k}\right|_{\tilde{\mathbf{x}}_k = \mathbf{0}} \tag{2.18}$$

$$\mathbf{\Gamma} = \left.\frac{\partial\left(\mathbf{g}\left(\hat{\mathbf{x}}_k, \mathcal{I}, \mathbf{n}_k\right) \boxminus \hat{\mathbf{x}}_{k+1}\right)}{\partial \mathbf{n}_k}\right|_{\mathbf{n}_k = \mathbf{0}} \tag{2.19}$$

Note that in many situations we only have measurements which relate to the continuous dynamics of the state, i.e., how the derivative evolves, rather than a direct propagation over a discrete timestep:

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}, \mathcal{I}, \mathbf{n}_c\right) \tag{2.20}$$

$$\mathbb{E}\left[\mathbf{n}_c(t_1)\mathbf{n}_c(t_2)^\top\right] = \mathbf{Q}_c\delta_{t_1t_2} \tag{2.21}$$

where δ_{ij} is the Kronecker delta function given by:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{2.22}$$

From the definition of the dynamics and the $\boxplus$ operation, we find the corresponding linearized error state dynamics [118]:

$$\dot{\tilde{\mathbf{x}}} \approx \mathbf{F}\tilde{\mathbf{x}} + \mathbf{G}\mathbf{n}_c \tag{2.23}$$

Consider the propagation of the system over a single step from times $[t_k, t_{k+1}]$. We define the state-transition matrix as the solution to the following system:

$$\dot{\mathbf{\Phi}}\left(t, t_k\right) = \mathbf{F}\left(t\right)\mathbf{\Phi}\left(t, t_k\right) \tag{2.24}$$

$$\mathbf{\Phi}\left(t_k, t_k\right) = \mathbf{I} \tag{2.25}$$

where $\mathbf{I}$ is the identity matrix. The solution to the error state propagation is given by [67]:

$$\tilde{\mathbf{x}}\left(t_{k+1}\right) = \mathbf{\Phi}\left(t_{k+1}, t_k\right)\tilde{\mathbf{x}}\left(t_k\right) + \int_{t_k}^{t_{k+1}} \mathbf{\Phi}\left(t_{k+1}, t\right)\mathbf{G}\left(t\right)\mathbf{n}_c(t)dt \tag{2.26}$$

$$= \mathbf{\Phi}\left(t_{k+1}, t_k\right)\tilde{\mathbf{x}}\left(t_k\right) + \mathbf{n}_d, \quad \mathbf{n}_d \sim \mathcal{N}\left(\mathbf{0}, \mathbf{Q}_d\right) \tag{2.27}$$

$$\mathbf{Q}_d = \int_{t_k}^{t_{k+1}} \mathbf{\Phi}\left(t_{k+1}, t\right)\mathbf{G}\left(t\right)\mathbf{Q}_c\mathbf{G}\left(t\right)^\top\mathbf{\Phi}\left(t_{k+1}, t\right)^\top dt \tag{2.28}$$

which can be easily identified with the standard form (2.17). In addition to linearizing the propagation function, we will also need to linearize the measurement functions:

$$\mathbf{z}_{k+1} - \mathbf{h}\left(\hat{\mathbf{x}}_{k+1}\right) \approx \mathbf{H}_{k+1}\tilde{\mathbf{x}}_{k+1} + \mathbf{n}_{k+1} \tag{2.29}$$

$$\mathbf{H}_{k+1} = \left.\frac{\partial \mathbf{h}_{k+1}\left(\hat{\mathbf{x}}_{k+1} \boxplus \tilde{\mathbf{x}}_{k+1}\right)}{\partial \tilde{\mathbf{x}}_{k+1}}\right|_{\tilde{\mathbf{x}}_{k+1}=\mathbf{0}} \tag{2.30}$$

2.1 Visual-Inertial Navigation

2.1.1 State Vector

In this section we briefly review the state definitions used in the VINS literature. As standard, the inertial navigation state is given by [94]:

$$\mathbf{x}_I = \begin{bmatrix} {}^I_G\bar{q}^\top & {}^G\mathbf{p}_I^\top & {}^G\mathbf{v}_I^\top & \mathbf{b}_\omega^\top & \mathbf{b}_a^\top \end{bmatrix}^\top \tag{2.31}$$

where ${}^I_G\bar{q}$ is the JPL unit quaternion [118] associated with the rotation matrix, ${}^I_G\mathbf{R}$, which rotates vectors from the global frame of reference $\{G\}$ into the local frame $\{I\}$ of the IMU, $\mathbf{b}_\omega$ and $\mathbf{b}_a$ are the gyroscope and accelerometer biases which corrupt the IMU measurements, ${}^G\mathbf{p}_I$ is the position of the IMU expressed in the global frame, and ${}^G\mathbf{v}_I$ is the corresponding velocity. In addition we define the corresponding error state:

$$\tilde{\mathbf{x}}_I = \begin{bmatrix} {}^I_G\tilde{\boldsymbol{\theta}} & {}^G\tilde{\mathbf{p}}_I^\top & {}^G\tilde{\mathbf{v}}_I^\top & \tilde{\mathbf{b}}_\omega^\top & \tilde{\mathbf{b}}_a^\top \end{bmatrix}^\top \tag{2.32}$$

where ${}^I_G\tilde{\boldsymbol{\theta}}$ is the quaternion error, while the other quantities have standard vector errors.

2.1.2 Inertial Measurement Unit

As the sensor platform navigates in space, the IMU measures both the angular velocity, $\boldsymbol{\omega}_m$, and local linear acceleration, $\mathbf{a}_m$, which are utilized for propagation [118]:

$$\boldsymbol{\omega}_m = {}^I\boldsymbol{\omega}_I + \mathbf{b}_\omega + \mathbf{n}_\omega \tag{2.33}$$

$$\mathbf{a}_m = {}^I\mathbf{a}_I + {}^I_G\mathbf{R}\,{}^G\mathbf{g} + \mathbf{b}_a + \mathbf{n}_a \tag{2.34}$$

where ${}^I\boldsymbol{\omega}_I$ and ${}^I\mathbf{a}_I$ represent the true local angular velocity and local linear acceleration of the IMU, while $\mathbf{n}_\omega$ and $\mathbf{n}_a$ are continuous-time Gaussian white noises, and ${}^G\mathbf{g} \approx$

$[0\ 0\ 9.81]^\top$ is the known global gravity. These measurements relate to the actual motion by the following continuous dynamics:

$$\begin{aligned}
{}^I_G\dot{\mathbf{R}} &= -\lfloor {}^I\boldsymbol{\omega}_I \times \rfloor {}^I_G\mathbf{R} \\
{}^G\dot{\mathbf{p}}_I &= {}^G\mathbf{v}_I \\
{}^G\dot{\mathbf{v}}_I &= {}^G_I\mathbf{R}\,{}^I\mathbf{a}_I \\
\dot{\mathbf{b}}_\omega &= \mathbf{n}_{\omega b} \\
\dot{\mathbf{b}}_a &= \mathbf{n}_{ab}
\end{aligned} \tag{2.35}$$

In particular, we note that both biases are modeled as random walks, and thus their derivatives are Gaussian noises. We can compute the error state dynamics using:

$$\begin{aligned}
{}^I_G\dot{\mathbf{R}} &= \frac{d}{dt}\left(\left(\mathbf{I} - \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor\right) {}^I_G\hat{\mathbf{R}}\right) \\
&\approx -\lfloor {}^I_G\dot{\tilde{\boldsymbol{\theta}}} \times \rfloor {}^I_G\hat{\mathbf{R}} + \left(\mathbf{I} - \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor\right) {}^I_G\dot{\hat{\mathbf{R}}} \\
-\lfloor \boldsymbol{\omega}_m - \mathbf{b}_\omega - \mathbf{n}_\omega \times \rfloor {}^I_G\mathbf{R} &= -\lfloor {}^I_G\dot{\tilde{\boldsymbol{\theta}}} \rfloor {}^I_G\hat{\mathbf{R}} - \left(\mathbf{I} - \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor\right) \lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \times \rfloor {}^I_G\hat{\mathbf{R}}
\end{aligned}$$

Using $\mathbf{b}_\omega = \hat{\mathbf{b}}_\omega + \tilde{\mathbf{b}}_\omega$, we have:

$$\begin{aligned}
-\lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega - \tilde{\mathbf{b}}_\omega - \mathbf{n}_\omega \times \rfloor \left(\mathbf{I} - \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor\right) {}^I_G\hat{\mathbf{R}} &= -\lfloor {}^I_G\dot{\tilde{\boldsymbol{\theta}}} \times \rfloor {}^I_G\hat{\mathbf{R}} \\
- \left(\mathbf{I} - \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor\right) \lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \times \rfloor {}^I_G\hat{\mathbf{R}} & \\
\lfloor \boldsymbol{\omega}_m - \mathbf{b}_\omega \times \rfloor \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor + \lfloor \tilde{\mathbf{b}}_\omega + \mathbf{n}_\omega \times \rfloor - \lfloor \tilde{\mathbf{b}}_\omega + \mathbf{n}_\omega \times \rfloor \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor &= \\
-\lfloor {}^I_G\dot{\tilde{\boldsymbol{\theta}}} \times \rfloor + \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor \lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \rfloor &
\end{aligned}$$

Finally we solve for the derivative of the orientation error:

$$\begin{aligned}
\lfloor {}^I_G\dot{\tilde{\boldsymbol{\theta}}} \times \rfloor &\approx \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor \lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \times \rfloor \\
&\quad - \lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \times \rfloor \lfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor - \lfloor \tilde{\mathbf{b}}_\omega + \mathbf{n}_\omega \times \rfloor \\
&= -\lfloor \lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \times \rfloor {}^I_G\tilde{\boldsymbol{\theta}} \times \rfloor - \lfloor \tilde{\mathbf{b}}_\omega + \mathbf{n}_\omega \times \rfloor \\
\Rightarrow {}^I_G\dot{\tilde{\boldsymbol{\theta}}} &= -\lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \times \rfloor {}^I_G\tilde{\boldsymbol{\theta}} - \tilde{\mathbf{b}}_\omega - \mathbf{n}_\omega
\end{aligned} \tag{2.36}$$

As for the other state elements, we have:

$$
\begin{aligned}
{}^{G}\dot{\mathbf{v}}_I = {}^{G}\hat{\dot{\mathbf{v}}}_I + {}^{G}\tilde{\dot{\mathbf{v}}}_I &= {}^{G}_{I}\mathbf{R}\left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a\right) \\
{}^{G}_{I}\hat{\mathbf{R}}\left(\mathbf{a}_m - \hat{\mathbf{b}}_a\right) + {}^{G}\tilde{\dot{\mathbf{v}}}_I &= {}^{G}_{I}\hat{\mathbf{R}}\left(\mathbf{I} + \lfloor {}^{I}_{G}\tilde{\boldsymbol{\theta}}\times \rfloor\right)\left(\mathbf{a}_m - \hat{\mathbf{b}}_a - \tilde{\mathbf{b}}_a - \mathbf{n}_a\right) \\
{}^{G}\tilde{\dot{\mathbf{v}}}_I &\approx -{}^{G}_{I}\hat{\mathbf{R}}\tilde{\mathbf{b}}_a - {}^{G}_{I}\hat{\mathbf{R}}\mathbf{n}_a - {}^{G}_{I}\hat{\mathbf{R}}\lfloor \mathbf{a}_m - \hat{\mathbf{b}}_a\times \rfloor {}^{I}_{G}\tilde{\boldsymbol{\theta}} \qquad (2.37) \\
{}^{G}\dot{\tilde{\mathbf{p}}}_I &= {}^{G}\tilde{\mathbf{v}}_I \qquad (2.38) \\
\dot{\tilde{\mathbf{b}}}_\omega &= \mathbf{n}_{\omega b} \qquad (2.39) \\
\dot{\tilde{\mathbf{b}}}_a &= \mathbf{n}_{ab} \qquad (2.40)
\end{aligned}
$$

With these we can construct the required error and noise propagation Jacobians.

$$
\begin{aligned}
\begin{bmatrix} {}^{I}_{G}\dot{\tilde{\boldsymbol{\theta}}} \\ {}^{G}\dot{\tilde{\mathbf{p}}}_I \\ {}^{G}\dot{\tilde{\mathbf{v}}}_I \\ \dot{\tilde{\mathbf{b}}}_\omega \\ \dot{\tilde{\mathbf{b}}}_a \end{bmatrix} \approx &\begin{bmatrix} -\lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega\times \rfloor & \mathbf{0}_3 & \mathbf{0}_3 & -\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -{}^{G}_{I}\hat{\mathbf{R}}\lfloor \mathbf{a}_m - \hat{\mathbf{b}}_a\times \rfloor & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & -{}^{G}_{I}\hat{\mathbf{R}} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \begin{bmatrix} {}^{I}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{G}\tilde{\mathbf{p}}_I \\ {}^{G}\tilde{\mathbf{v}}_I \\ \tilde{\mathbf{b}}_\omega \\ \tilde{\mathbf{b}}_a \end{bmatrix} \\
&+ \begin{bmatrix} -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & -{}^{G}_{I}\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \mathbf{n}_\omega \\ \mathbf{n}_a \\ \mathbf{n}_{\omega b} \\ \mathbf{n}_{ab} \end{bmatrix}
\end{aligned} \qquad (2.41)
$$

Now that we have defined the linearized system associated with (2.23), we can use either numerical [94] or analytical (see Chapter 3) integration to integrate both the state and error state dynamics.

2.1.3 Indirect Camera Measurements

A vast majority of this work utilizes indirect visual frontends, and so we here review the required theory. As the sensing platform moves through its environment, its onboard cameras captures images of its surroundings. Consider a 3D point (feature)

expressed as a position in the global frame, ${}^{G}\mathbf{p}_f$. The measurement captured by the camera is given by [94]:

$$\mathbf{z} = \mathbf{\Pi}\left({}^{C}\mathbf{p}_f\right) + \mathbf{n} \tag{2.42}$$

$${}^{C}\mathbf{p}_f = {}^{C}_{I}\mathbf{R}\,{}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_I\right) + {}^{C}\mathbf{p}_I \tag{2.43}$$

$$\Pi\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}\right) = \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \end{bmatrix} \tag{2.44}$$

where $\{{}^{C}_{I}\mathbf{R}, {}^{C}\mathbf{p}_I\}$ are the extrinsics between the IMU and camera. We can then compute the Jacobians with respect to the state, evaluated at the state estimate, $\hat{\mathbf{x}}$:

$$\frac{\partial \mathbf{z}}{\partial \tilde{\mathbf{x}}} = \frac{\partial \mathbf{z}}{\partial {}^{C}\tilde{\mathbf{p}}_f}\frac{{}^{C}\tilde{\mathbf{p}}_f}{\tilde{\mathbf{x}}} \tag{2.45}$$

$$\frac{\partial \mathbf{z}}{\partial {}^{C}\tilde{\mathbf{p}}_f} = \begin{bmatrix} \frac{1}{\hat{z}} & 0 & -\frac{\hat{x}}{\hat{z}^2} \\ 0 & \frac{1}{\hat{z}} & -\frac{\hat{y}}{\hat{z}^2} \end{bmatrix} \tag{2.46}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_f}{\partial {}^{G}\tilde{\mathbf{p}}_f} = {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} \tag{2.47}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_f}{\partial {}^{G}\tilde{\mathbf{p}}_I} = -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} \tag{2.48}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_f}{\partial {}^{I}_{G}\tilde{\boldsymbol{\theta}}} = {}^{C}_{I}\hat{\mathbf{R}}\lfloor {}^{I}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_f - {}^{G}\hat{\mathbf{p}}_I\right)\times\rfloor \tag{2.49}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_f}{\partial {}^{C}\tilde{\mathbf{p}}_I} = \mathbf{I}_3 \tag{2.50}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_f}{\partial {}^{C}_{I}\tilde{\boldsymbol{\theta}}} = \lfloor {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_f - {}^{G}\hat{\mathbf{p}}_I\right)\times\rfloor \tag{2.51}$$

Note that we report the Jacobians with respect to the calibration parameters as we can perform online estimation of these values by treating these as random variables and adding them to our state vector to be estimated.

2.2 Batch Optimization

Batch optimization, or bundle adjustment (BA), uses a graph representation of the estimation problem where nodes are quantities to be solved for and edges are

measurements that relate nodes within the graph to each other. In the case of graph SLAM [50], the graph nodes can correspond to historical robot states and features in the environment, while the edges represent collected measurements from sensors which relate the incident nodes. As an example within VINS, measuring a 3D feature from an image would add an edge between the feature position and the sensor pose at the time the image was captured. The state evolution of the robot from one time to the next based on the collected IMU readings would create an edge between the corresponding start and end historical states.

Using this graph formulation and under the assumption of independent zero-mean Gaussian noise for all measurements, we can find a maximum a posteriori (MAP) estimate of all states by solving the following nonlinear least-squares problem [74]:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_i \frac{1}{2} \left|\left|\mathbf{r}_i\left(\mathbf{x}\right)\right|\right|^2_{\mathbf{\Lambda}_i} := \underset{\mathbf{x}}{\operatorname{argmin}} \sum_i c_i\left(\mathbf{x}\right) \tag{2.52}$$

where $\mathbf{r}_i$ is the error/residual of the i-th measurement (typically defined as $\mathbf{r}_i = \mathbf{h}_i(\mathbf{x}) - \mathbf{z}_i$ for vector measurements), $\mathbf{\Lambda}_i$ is the associated information matrix (inverse covariance), and $||\mathbf{v}||^2_{\mathbf{\Lambda}} = \mathbf{v}^\top \mathbf{\Lambda} \mathbf{v}$ represents the squared energy norm. For simplicity we let $c_i\left(\mathbf{x}\right)$ denote the cost associated with measurement i. Note that as a common practice, a (Huber or Cauchy) robust cost function of Equation (2.52) is often used to compensate for outliers, in particular when fusing visual measurements [54].

In the standard batch VINS formulation [75], we have the following cost associated with propagating over an interval:

$$c_{IMU}(\mathbf{x}) = \frac{1}{2} \left|\left|\mathbf{x}_{k+1} \boxminus \mathbf{g}\left(\mathbf{x}_k, \mathcal{I}, \mathbf{0}\right)\right|\right|^2_{\mathbf{Q}_k^{-1}} \tag{2.53}$$

while for vision measurements, we have:

$$c_f(\mathbf{x}) = \frac{1}{2} \left|\left|\mathbf{h}\left(\mathbf{x}\right) - \mathbf{z}\right|\right|^2_{\mathbf{R}_k^{-1}} \tag{2.54}$$

Optimization is typically performed iteratively, e.g., through a Gauss-Newton or Levenberg–Marquard method, by linearizing the nonlinear measurements about the

current estimate, $\hat{\mathbf{x}}$, and defining a new weighted linear least squares problem in terms of the error state $\tilde{\mathbf{x}}$:

$$\hat{\tilde{\mathbf{x}}} = \underset{\tilde{\mathbf{x}}}{\operatorname{argmin}} \sum_i \frac{1}{2} ||\mathbf{r}_i(\hat{\mathbf{x}}) + \mathbf{H}_i \tilde{\mathbf{x}}||^2_{\mathbf{\Lambda}_i} \tag{2.55}$$

$$\mathbf{H}_i = \frac{\partial \mathbf{h}_i(\hat{\mathbf{x}} \boxplus \tilde{\mathbf{x}})}{\partial \tilde{\mathbf{x}}}\Big|_{\tilde{\mathbf{x}}=\mathbf{0}} \tag{2.56}$$

We can see that the original optimization problem has been converted into finding the optimal *correction* vector, $\tilde{\mathbf{x}}$, to the current state estimate. The optimal solution can be found by solving the following normal equation:

$$\left(\sum_i \mathbf{H}_i^\top \mathbf{\Lambda}_i \mathbf{H}_i\right) \hat{\tilde{\mathbf{x}}} = -\sum_i \mathbf{H}_i^\top \mathbf{\Lambda}_i \mathbf{r}_i(\hat{\mathbf{x}}) \tag{2.57}$$

$$\iff \mathbf{\Lambda} \hat{\tilde{\mathbf{x}}} = -\mathbf{g} \tag{2.58}$$

After obtaining the an estimate for the optimal correction, $\hat{\tilde{\mathbf{x}}}$, we update our current estimate at the m-th iteration as: $\hat{\mathbf{x}}^{(m+1)} = \hat{\mathbf{x}}^{(m)} \boxplus \hat{\tilde{\mathbf{x}}}$, and repeat the optimization process. After convergence, we will be left with the following distribution:

$$\mathbf{x} = \hat{\mathbf{x}} \boxplus \tilde{\mathbf{x}} \tag{2.59}$$

$$\tilde{\mathbf{x}} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}) \tag{2.60}$$

$$\mathbf{\Sigma} = \left(\sum_i \mathbf{H}_i^\top \mathbf{\Lambda}_i \mathbf{H}_i\right)^{-1} \tag{2.61}$$

where the measurement Jacobians, $\mathbf{H}_i$, are evaluated at the final state estimate. We note that while we have presented the full Gauss-Newton optimization, approximate solvers such as iSAM2 [69] are often used to speed-up performance at the cost of some accuracy.

2.3 Extended Kalman Filtering

As compared to batch methods which solve an optimization problem over the entire history of measurements at every timestep, Extended Kalman Filtering (EKF) processes measurements sequentially to track both the state estimate, $\hat{\mathbf{x}}$, and error

covariance, $\mathbf{P}$, over time. While this does not allow for relinearization of old measurements, its low computational cost makes it an optimal choice for many resource-constrained applications [125]. In the EKF formulation, at every timestep k we track the estimate and covariance as:

$$\mathbf{x}_k = \hat{\mathbf{x}}_k \boxplus \tilde{\mathbf{x}}_k \tag{2.62}$$

$$\tilde{\mathbf{x}}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{P}_k) \tag{2.63}$$

As new proprioceptive information arrives, the EKF works by linearizing the nonlinear prediction function, and performing a propagation of both the state estimate and covariance. In particular, based on the standard propagation function (2.1) we have:

$$\hat{\mathbf{x}}_{k+1} = \mathbf{g}(\hat{\mathbf{x}}_k, \mathcal{I}, \mathbf{0}) \tag{2.64}$$

$$\mathbf{P}_{k+1} = \mathbf{\Phi}\mathbf{P}_k\mathbf{\Phi}^\top + \mathbf{\Gamma}\mathbf{Q}_k\mathbf{\Gamma}^\top \tag{2.65}$$

After propagation, the EKF then seeks to update the state using newly collected measurements:

$$\mathbf{z}_{k+1} = \mathbf{h}_{k+1}(\mathbf{x}_{k+1}) + \mathbf{n}_{k+1} \tag{2.66}$$

$$\mathbf{n}_{k+1} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{k+1}) \tag{2.67}$$

After performing linearization, the EKF update is achieved:

$$\mathbf{H} = \left.\frac{\partial \mathbf{h}\left(\hat{\mathbf{x}}^{\ominus}_{k+1} \boxplus \tilde{\mathbf{x}}\right)}{\partial \tilde{\mathbf{x}}}\right|_{\tilde{\mathbf{x}}=\mathbf{0}} \tag{2.68}$$

$$\mathbf{S} = \mathbf{H}\mathbf{P}^{\ominus}_{k+1}\mathbf{H}^\top + \mathbf{R}_{k+1} \tag{2.69}$$

$$\mathbf{K} = \mathbf{P}_{k+1}\mathbf{H}^\top\mathbf{S}^{-1} \tag{2.70}$$

$$\hat{\mathbf{x}}^{\oplus}_{k+1} = \hat{\mathbf{x}}^{\ominus}_{k+1} \boxplus \mathbf{K}\left(\mathbf{z} - \mathbf{h}\left(\hat{\mathbf{x}}^{\ominus}_{k+1}\right)\right) \tag{2.71}$$

$$\mathbf{P}^{\oplus}_{k+1} = \mathbf{P}^{\ominus}_{k+1} - \mathbf{K}\mathbf{S}\mathbf{K}^\top \tag{2.72}$$

where the $\ominus$ and $\oplus$ operators denote the state estimate/covariance before and after update respectively.

2.4 Multi-State Constraint Kalman Filter

As a large portion of this thesis (thrusts two and three) is based on the MSCKF [94], we here present its formulation. MSCKF is an EKF-VINS which extracts all the information contained in visual feature measurements about the motion of the platform without storing the involved landmarks into the state vector, leading to large computational savings. In particular, we note that because the visual measurements are a function of the unknown feature position, ${}^{G}\mathbf{p}_f$, we need to naively add these to our state vector in order to utilize the EKF update, which may become prohibitively expensive for large number of visual measurements. To remedy this, the MSCKF collects the entire history of visual tracks for a given feature, and extracts all the information contained in the measurements about the ego-motion of the sensing platform across a short interval.

To achieve this, in addition to the standard evolving state, the MSCKF also maintains the pose of the IMU corresponding to a sliding window of image times. In particular, at timestep k and for a window size of N, our IMU "clones" are given by:

$$\mathbf{x}_{cl} = \begin{bmatrix} {}^{I_{k-1}}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_{k-1}}^\top & \cdots & {}^{I_{k-N+1}}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_{k-N+1}}^\top \end{bmatrix}^\top \tag{2.73}$$

To add these clones to our state, at each imaging time we utilize stochastic cloning to augment our state vector [110]. In particular, let $\mathbf{x}$ denote the current state (with covariance $\mathbf{P}$) and $\mathbf{v}$ be a new variable that we wish to add to our state, such that $\mathbf{v} = \mathbf{c}(\mathbf{x})$. In the case that we wish to directly clone a state element without modifying it, this function simply becomes an identity transformation. Based on the function we define, we augment our estimate and covariance as:

$$\mathbf{x}^{\oplus} = \begin{bmatrix} \mathbf{x}^\top & \mathbf{v}^\top \end{bmatrix}^\top \tag{2.74}$$

$$\hat{\mathbf{x}}^{\oplus} = \begin{bmatrix} \hat{\mathbf{x}}^\top & \mathbf{c}(\hat{\mathbf{x}})^\top \end{bmatrix}^\top \tag{2.75}$$

$$\mathbf{P}^{\oplus} = \begin{bmatrix} \mathbf{P} & \mathbf{P}\mathbf{C}^\top \\ \mathbf{C}\mathbf{P} & \mathbf{C}\mathbf{P}\mathbf{C}^\top \end{bmatrix} \tag{2.76}$$

where $\mathbf{C}$ is the Jacobian of the cloning function. As this function typically only involves a small portion of our state, its resulting sparsity can be exploited to greatly speed up the computation of the augmented covariance.

Given our state of stochastic clones, Let $\mathbf{r}$ denote the $m \times 1$ stacked vector of residuals associated with this feature, $\mathbf{r}_k = \mathbf{z}_k - \mathbf{\Pi}\left({}^{C_k}\hat{\mathbf{p}}_f\right)$. These measurements are a function of both the feature and the clones we have created. We use the state and feature estimates to evaluate ${}^{C_k}\hat{\mathbf{p}}_f$ and to compute the measurement Jacobians:

$$\mathbf{r} = \mathbf{H}_x\tilde{\mathbf{x}} + \mathbf{H}_f{}^{G}\tilde{\mathbf{p}}_f + \mathbf{n} \tag{2.77}$$

where $\mathbf{H}_x$ and $\mathbf{H}_f$ refer to the stacked Jacobians of the residuals with respect to the state and feature variables respectively, while $\mathbf{n}$ is the $m \times 1$ stacked noise vector with covariance $\sigma^2\mathbf{I}_m$. The key idea of the MSCKF is to remove the dependency of the feature measurements on the unknown landmark by projecting this system onto the nullspace of the feature Jacobian, which is spanned by the columns of matrix $\mathbf{N}$.

$$\mathbf{N}^\top\mathbf{H}_f = \mathbf{0} \tag{2.78}$$

$$\mathbf{N}^\top\mathbf{r} = \mathbf{N}^\top\mathbf{H}_x\tilde{\mathbf{x}} + \mathbf{N}^\top\mathbf{H}_f{}^{G}\tilde{\mathbf{p}}_f + \mathbf{N}^\top\mathbf{n} \tag{2.79}$$

$$\Rightarrow \mathbf{r}' = \mathbf{H}'_x\tilde{\mathbf{x}} + \mathbf{n}' \tag{2.80}$$

where the transformed noise, $\mathbf{n}'$, has covariance $\mathbf{N}^\top\left(\sigma^2\mathbf{I}_m\right)\mathbf{N} = \sigma^2\mathbf{I}_{m-3}$. As this transformed measurement is only a function of the stochastic clones in our state, it can be directly utilized in the EKF update.

Chapter 3

CLOSED-FORM IMU PREINTEGRATION METHODS

In this chapter we focus on improving the "standard" VINS-paradigm of a single IMU-camera pair. In particular, we look to improve the accuracy of how IMU readings are processed. The recent development of preintegration has allowed for the efficient inclusion of high-rate IMU measurements in graph-based SLAM [42, 43, 88]. In this chapter, building upon our prior work [31], we investigate in-depth the optimal use of preintegration by providing models and their closed-form solutions for the preintegrated measurement dynamics, allowing for more accurate computation of the inertial factors for use in graph optimization of VINS. We show that these models are especially effective in highly dynamic scenarios involving low frequency IMU. In particular, the main contributions of this work include [29]:

- We advocate for two new preintegration models (i.e., piecewise constant measurements and piecewise constant local true acceleration, instead of piecewise constant global acceleration as assumed in existing methods) to better capture the underlying motion dynamics and offer the analytical solutions to the preintegration equations. We have open sourced the proposed preintegration to better contribute to our research community.[1]
- Using the proposed closed-form preintegration, we develop an indirect, tightly-coupled, sliding-window optimization based visual-inertial odometry (VIO), which marginalizes out features from the state vector when moving to the next time window to enable real-time performance of bounded computational cost.
- With the proposed closed-form IMU preintegration, we further develop a loosely-coupled, direct VINS, which fuses preintegrated inertial measurements with direct image alignment results.

[1] The open source of the proposed closed-form preintegration is available at: https://github.com/rpng/cpi

- We conduct thorough Monte-Carlo simulation analysis of different preintegration models by varying motion dynamics and IMU sampling rates. We also perform extensive real-world experiments to validate the proposed VINS using our preintegration by comparing with a state-of-the-art method.

3.1 Related Work

First introduced by Lupton et al. [88], inertial preintegration is a computationally efficient alternative to the standard inertial measurement integration, e.g., as performed in EKF propagation. The authors employed the discrete integration of the inertial measurement dynamics in a *local* frame of reference, preventing the need to reintegrate the state dynamics at each optimization step. While this addresses the computational complexity issue, this method suffers from singularities due to the use of Euler angles in the orientation representation. To improve the stability of this preintegration, an on-manifold representation was introduced by Forster et al. [42, 43] which presents a singularity-free orientation representation on the $SO(3)$ manifold, incorporating IMU preintegration into an efficient graph-based VINS algorithm.

While Shen et al. [114] introduced preintegration in the continuous form, they still discretely sampled the measurement dynamics without offering closed-form solutions. This left a significant gap in the theoretical completeness of preintegration theory from a continuous-time perspective. Albeit, Qin et al. [103] later extended to a robust tightly-coupled monocular visual-inertial localization system. As compared to the discrete approximation of the preintegrated measurement and covariance calculations used in previous methods, in our prior work [31], we have derived the closed-form solutions to both the measurement and covariance preintegration equations and showed that these solutions offer improved accuracy over the discrete methods, especially in the case of highly dynamic motion.

In this chapter, based on our preliminary results [30, 31], we provide a solid theoretical foundation for closed-form preintegration and show that it can be easily incorporated into different graph-based sensor fusion methods. We investigate the improved accuracy afforded by two different models of closed-form preintegration and

scenarios in which they exhibit superior performance. We further develop both indirect and direct graph-based VINS and demonstrate their competitive performance to state-of-the-art methods.

3.2 Closed-Form Preintegration

In this section, we present in detail the proposed closed-form IMU preintegration based on two different realistic inertial models.

3.2.1 Standard IMU Processing

The incorporation of *tightly-coupled* VINS in batch optimization methods requires overcoming the high frequency nature and computational complexity of the inertial measurements. Given a series of IMU measurements, $\mathcal{I}$, collected over a time interval $[t_k, t_{k+1}]$, the standard (graph-based) IMU processing considers the following propagation function (2.1):

$$\mathbf{x}_{k+1} = \mathbf{g}\left(\mathbf{x}_k, \mathcal{I}, \mathbf{n}_k\right) \tag{3.1}$$

That is, the future state at time step $k+1$ is a function of the current state at step k, the IMU measurements $\mathcal{I}$, and the corresponding measurement noise $\mathbf{n}_k$. Conditioning on the current state, the expected value of the next state is found by evaluating the propagation function with zero noise:

$$\breve{\mathbf{x}}_{k+1} = \mathbf{g}\left(\mathbf{x}_k, \mathcal{I}, \mathbf{0}\right) \tag{3.2}$$

which implies that we perform integration of the state dynamics in the absence of noise.

The residual for use in batch optimization of this propagation now constrains the start and end states of the interval and is given by (here we rewrite Equation (2.52)):

$$c_{IMU}(\mathbf{x}) = \frac{1}{2}\left|\left|\mathbf{x}_{k+1} \boxminus \breve{\mathbf{x}}_{k+1}\right|\right|^2_{\mathbf{Q}_k^{-1}} \tag{3.3}$$

$$= \frac{1}{2}\left|\left|\mathbf{x}_{k+1} \boxminus \mathbf{g}\left(\mathbf{x}_k, \mathcal{I}, \mathbf{0}\right)\right|\right|^2_{\mathbf{Q}_k^{-1}} \tag{3.4}$$

where $\mathbf{Q}_k$ is the linearized, discrete-time noise covariance computed from the IMU noise characterization and is a *function of the state*. This noise covariance matrix

and the propagation function can be found by the integration of Equations (2.35) and their associated error state dynamics, to which we refer the reader to [94, 118]. It is clear from (3.2) that ideally we need to constantly re-evaluate the propagation function $\mathbf{g}(\cdot)$ and the residual covariance $\mathbf{Q}_k$ whenever the linearization point (state estimate) changes. However, the high frequency nature of the IMU sensors and the complexity of the propagation function and the noise covariance motivates the development of inertial preintegration.

3.2.2 Model 1: Piecewise Constant Measurements

IMU preintegration seeks to directly reduce the computational complexity of incorporating inertial measurements by removing the need to re-integrate the propagation function and noise covariance. This is achieved by processing IMU measurements in a *local* frame of reference, yielding measurements that are, in contrast to Equation (3.2), independent of the state [88].

Specifically, by denoting $\Delta T = t_{k+1} - t_k$, we have the following relationship between a series of IMU measurements, the start state, and the resulting end state [31]:

$$\begin{aligned}
{}^{G}\mathbf{p}_{k+1} &= {}^{G}\mathbf{p}_k + {}^{G}\mathbf{v}_k \Delta T - \frac{1}{2}{}^{G}\mathbf{g}\Delta T^2 \\
&\quad + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a\right) du ds \qquad (3.5) \\
{}^{G}\mathbf{v}_{k+1} &= {}^{G}\mathbf{v}_k - {}^{G}\mathbf{g}\Delta T \\
&\quad + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a\right) du \qquad (3.6) \\
{}^{k+1}_{G}\mathbf{R} &= {}^{k+1}_{k}\mathbf{R}\ {}^{k}_{G}\mathbf{R} \qquad (3.7) \\
\mathbf{b}_{\omega_{k+1}} &= \mathbf{b}_{\omega_k} + \int_{t_k}^{t_{k+1}} \mathbf{n}_{\omega b}\ du \qquad (3.8) \\
\mathbf{b}_{a_{k+1}} &= \mathbf{b}_{a_k} + \int_{t_k}^{t_{k+1}} \mathbf{n}_{ab}\ du \qquad (3.9)
\end{aligned}$$

where u and s are dummy variables in the integration. From the above, we define the following preintegrated IMU measurements:

$$ {}^{k}\boldsymbol{\alpha}_{k+1} = \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a \right) duds \tag{3.10} $$

$$ {}^{k}\boldsymbol{\beta}_{k+1} = \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a \right) du \tag{3.11} $$

Note that along with the preintegrated inertial measurements in Equations (3.10) and (3.11), the preintegrated relative-orientation measurement ${}^{k+1}_{k}\bar{q}$ (or ${}^{k+1}_{k}\mathbf{R}$) can be obtained from the integration of the gyro measurements. To remove the dependencies of the above preintegrated measurements on the true biases, we linearize about the current bias estimates at time step t_k, $\mathbf{b}^{\star}_{a_k}$ and $\mathbf{b}^{\star}_{\omega_k}$. Defining $\Delta\mathbf{b} = \mathbf{b} - \mathbf{b}^{\star}$, we have (noting that time indices are occasionally omitted to keep expressions concise, which however can be easily inferred from the context):

$$ {}^{k}_{G}\mathbf{R} \left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_k - {}^{G}\mathbf{v}_k \Delta T + \frac{1}{2} {}^{G}\mathbf{g} \Delta T^2 \right) \simeq \tag{3.12} $$
$$ {}^{k}\boldsymbol{\alpha}_{k+1} \left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k} \right) + \frac{\partial \boldsymbol{\alpha}}{\partial \mathbf{b}_{\omega}} \Big|_{\mathbf{b}^{\star}_{\omega_k}} \Delta\mathbf{b}_{\omega} + \frac{\partial \boldsymbol{\alpha}}{\partial \mathbf{b}_{a}} \Big|_{\mathbf{b}^{\star}_{a_k}} \Delta\mathbf{b}_{a} $$

$$ {}^{k}_{G}\mathbf{R} \left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_k + {}^{G}\mathbf{g} \Delta T \right) \simeq \tag{3.13} $$
$$ {}^{k}\boldsymbol{\beta}_{k+1} \left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k} \right) + \frac{\partial \boldsymbol{\beta}}{\partial \mathbf{b}_{\omega}} \Big|_{\mathbf{b}^{\star}_{\omega_k}} \Delta\mathbf{b}_{\omega} + \frac{\partial \boldsymbol{\beta}}{\partial \mathbf{b}_{a}} \Big|_{\mathbf{b}^{\star}_{a_k}} \Delta\mathbf{b}_{a} $$

$$ {}^{k+1}_{G}\mathbf{R} \, {}^{k}_{G}\mathbf{R}^{\top} \simeq \mathbf{R} \left(\frac{\partial \mathbf{R}}{\partial \mathbf{b}_{\omega}} \Big|_{\mathbf{b}^{\star}_{\omega_k}} \Delta\mathbf{b}_{\omega} \right) {}^{k+1}_{k}\mathbf{R} \left(\mathbf{b}^{\star}_{\omega_k} \right) \tag{3.14} $$

Note that Equations (3.12) and (3.13) are simple Taylor series expansions for our ${}^{k}\boldsymbol{\alpha}_{k+1}$ and ${}^{k}\boldsymbol{\beta}_{k+1}$ measurements, while Equation (3.14) models an additional rotation induced due to a change of the linearization point (estimate) of the gyro bias [42, 31].

The preintegrated measurement's mean values, ${}^{k}\breve{\boldsymbol{\alpha}}_{k+1}$, ${}^{k}\breve{\boldsymbol{\beta}}_{k+1}$, and ${}^{k+1}_{k}\breve{\bar{q}}$, must be computed for use in graph optimization. It is important to note that current preintegration methods [43, 85, 88] are all based on discrete integration of the measurement dynamics through Euler or midpoint integration. In particular, the discrete approximation used by Forster et al. [43] in fact corresponds to a piecewise constant *global acceleration* model (expressed in the fixed global frame of reference), which may be

easily violated in realistic navigation. By contrast, we here offer *closed-form* solutions for the measurement means under the assumptions of piecewise constant (local) *measurements* and piecewise constant *local acceleration* (expressed in local coordinates) which will be presented later in Section 3.2.3. For brevity we omit some of the details in the derivations, however these can be found in Appendix A.

3.2.2.1 Computing Preintegration Mean:

Between two image times, t_k and t_{k+1}, the IMU receives a series of inertial measurements. We denote τ as the step at which an IMU measurement is received, and $\tau + 1$ as the step of the *next* IMU reading. The time associated with each of these steps is given by t_τ and $t_{\tau+1}$, respectively. The relative orientation between the interval, ${}^{k+1}_{k}\breve{q}$, can be found using successive applications of the zeroth order quaternion integrator [118]. Based on the definitions of ${}^{k}\boldsymbol{\alpha}_{k+1}$ and ${}^{k}\boldsymbol{\beta}_{k+1}$ (see Equations (3.10) and (3.11)), we have the following continuous-time dynamics at every step u with $t_u \in [t_\tau, t_{\tau+1}]$:

$$ {}^{k}\dot{\boldsymbol{\alpha}}_u = {}^{k}\boldsymbol{\beta}_u \tag{3.15} $$

$$ {}^{k}\dot{\boldsymbol{\beta}}_u = {}^{k}_{u}\mathbf{R}\,(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a) \tag{3.16} $$

From these governing differential equations, we formulate the following *linear* system that describes the evolution of the measurements by taking the expectation operation:

$$ \begin{bmatrix} {}^{k}\dot{\breve{\boldsymbol{\alpha}}}_u \\ {}^{k}\dot{\breve{\boldsymbol{\beta}}}_u \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_u \\ {}^{k}\breve{\boldsymbol{\beta}}_u \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ {}^{k}_{u}\breve{\mathbf{R}} \end{bmatrix} (\mathbf{a}_m - \mathbf{b}^{\star}_{a_k}) \tag{3.17} $$

Given $\boldsymbol{a}_m$ and $\boldsymbol{\omega}_m$ sampled at time t_τ and assuming that these local IMU measurements are *piecewise constant* during $[t_\tau, t_{\tau+1}]$, we analytically solve the above linear time-varying (LTV) system to obtain the updated preintegration mean values, which

are computed as follows (see Appendix A):

$$\begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau+1} \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau+1} \end{bmatrix} = \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau} + {}^{k}\breve{\boldsymbol{\beta}}_{\tau}\Delta t + \mathbf{A}_{\tau}\hat{\mathbf{a}} \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau} + \mathbf{B}_{\tau}\hat{\mathbf{a}} \end{bmatrix} \tag{3.18}$$

$$\mathbf{A}_{\tau} = {}^{k}_{\tau+1}\breve{\mathbf{R}}\Big(\frac{\Delta t^2}{2}\mathbf{I}_{3\times 3} + \frac{|\hat{\boldsymbol{\omega}}|\Delta t\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|\hat{\boldsymbol{\omega}}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{(|\hat{\boldsymbol{\omega}}|\Delta t)^2 - 2\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - 2(|\hat{\boldsymbol{\omega}}|\Delta t)\sin(|\hat{\boldsymbol{\omega}}|\Delta t) + 2}{2|\hat{\boldsymbol{\omega}}|^4}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2\Big) \tag{3.19}$$

$$\mathbf{B}_{\tau} = {}^{k}_{\tau+1}\breve{\mathbf{R}}\Big(\Delta t\mathbf{I}_{3\times 3} - \frac{1-\cos(|\hat{\boldsymbol{\omega}}|(\Delta t))}{|\hat{\boldsymbol{\omega}}|^2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|\hat{\boldsymbol{\omega}}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2\Big) \tag{3.20}$$

where we have employed the definitions: $\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}_m - \mathbf{b}^{\star}_{\omega_k}$, $\hat{\mathbf{a}} = \mathbf{a}_m - \mathbf{b}^{\star}_{a_k}$, and $\Delta t = t_{\tau+1} - t_{\tau}$. Clearly, these *closed-form* expressions reveal the higher order affect of the angular velocity on the preintegrated measurements due to the evolution of the orientation over the IMU samping interval.

3.2.2.2 Computing Preintegration Covariance:

In order to derive the preintegrated measurement covariance, we first write the linearized *measurement* error system as follows [28]:

$$\begin{bmatrix} {}^{u}_{k}\dot{\tilde{\boldsymbol{\theta}}} \\ {}^{k}\dot{\tilde{\boldsymbol{\alpha}}}_{u} \\ {}^{k}\dot{\tilde{\boldsymbol{\beta}}}_{u} \\ \dot{\widetilde{\mathbf{b}}}_{\omega} \\ \dot{\widetilde{\mathbf{b}}}_{a} \end{bmatrix} = \begin{bmatrix} -\lfloor\hat{\boldsymbol{\omega}}\times\rfloor & \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ -{}^{k}_{u}\breve{\mathbf{R}}\lfloor\hat{\mathbf{a}}\times\rfloor & \mathbf{0} & \mathbf{0} & \mathbf{0} & -{}^{k}_{u}\breve{\mathbf{R}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} {}^{u}_{k}\tilde{\boldsymbol{\theta}} \\ {}^{k}\tilde{\boldsymbol{\alpha}}_{u} \\ {}^{k}\tilde{\boldsymbol{\beta}}_{u} \\ \widetilde{\mathbf{b}}_{\omega} \\ \widetilde{\mathbf{b}}_{a} \end{bmatrix} + \begin{bmatrix} -\mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -{}^{k}_{u}\breve{\mathbf{R}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{n}_{\omega} \\ \mathbf{n}_{a} \\ \mathbf{n}_{\omega b} \\ \mathbf{n}_{ab} \end{bmatrix} \tag{3.21}$$

$$\iff \dot{\mathbf{r}} = \mathbf{F}\mathbf{r} + \mathbf{G}\mathbf{n} \tag{3.22}$$

which is akin to the standard VINS error state propagation equations in a local frame of reference (2.41).

It is important to note that in contrast to our previous work [30, 31], we here couple the preintegration bias and measurement evolution for improved accuracy. Note also that the bias error terms in Equation (3.21), $\widetilde{\mathbf{b}}_{\omega}$ and $\widetilde{\mathbf{b}}_{a}$, describe the deviation of

the bias over the interval due to the random-walk drift, rather than the error of the current bias estimate. The discrete state transition matrix $\mathbf{\Phi}(t_{\tau+1}, t_\tau)$ can be computed either analytically in closed-form or numerically using Runge-Kutta methods based on the following continuous-time differential equation (see [58, 118]):

$$\dot{\mathbf{\Phi}}(t_u, t_\tau) = \mathbf{F}(u)\ \mathbf{\Phi}(t_u, t_\tau) \tag{3.23}$$

$$\mathbf{\Phi}(t_\tau, t_\tau) = \mathbf{I} \tag{3.24}$$

The propagation of the measurement covariance, $\mathbf{P}$, over the time interval $t_\tau \in [t_k, t_{k+1}]$, takes the following form:

$$\mathbf{P}_k = \mathbf{0} \tag{3.25}$$

$$\mathbf{P}_{\tau+1} = \mathbf{\Phi}(t_{\tau+1}, t_\tau)\ \mathbf{P}_\tau\ \mathbf{\Phi}(t_{\tau+1}, t_\tau)^\top + \mathbf{Q}_\tau \tag{3.26}$$

$$\mathbf{Q}_\tau = \int_{t_\tau}^{t_{\tau+1}} \mathbf{\Phi}(t_{\tau+1}, u)\mathbf{G}(u)\mathbf{Q}_c\mathbf{G}(u)^\top\mathbf{\Phi}(t_{\tau+1}, u)^\top du \tag{3.27}$$

where $\mathbf{Q}_c$ is the continuous-time IMU noise covariance. To keep presentation concise, the discrete-time noise covariance $\mathbf{Q}_\tau$, can be computed similarly as in [118].

3.2.2.3 Preintegration Measurement Residuals and Jacobians:

For use in optimization, we form the associated preintegration measurement cost and residual as follows:

$$c_{IMU}(\mathbf{x}) = \frac{1}{2}\left|\left|\mathbf{r}_{IMU}(\mathbf{x})\right|\right|^2_{\mathbf{P}^{-1}_{k+1}} \tag{3.28}$$

$$\mathbf{r}_{IMU}(\mathbf{x}) = \left[\begin{array}{c} 2\mathbf{vec}\left({}^{k+1}_{G}\bar{q} \otimes {}^{k}_{G}\bar{q}^{-1} \otimes {}^{k+1}_{k}\breve{\bar{q}}^{-1} \otimes \bar{q}_b\right) \\ \hline \left(\begin{array}{c} {}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_k - {}^{G}\mathbf{v}_k\Delta T + \frac{1}{2}{}^{G}\mathbf{g}\Delta T^2\right) \\ -\mathbf{J}_\alpha\left(\mathbf{b}_{\omega_k} - \mathbf{b}^\star_{\omega_k}\right) - \mathbf{H}_\alpha\left(\mathbf{b}_{a_k} - \mathbf{b}^\star_{a_k}\right) - {}^{k}\breve{\boldsymbol{\alpha}}_{k+1} \end{array}\right) \\ \hline \left(\begin{array}{c} {}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_k + {}^{G}\mathbf{g}\Delta T\right) \\ -\mathbf{J}_\beta\left(\mathbf{b}_{\omega_k} - \mathbf{b}^\star_{\omega_k}\right) - \mathbf{H}_\beta\left(\mathbf{b}_{a_k} - \mathbf{b}^\star_{a_k}\right) - {}^{k}\breve{\boldsymbol{\beta}}_{k+1} \end{array}\right) \\ \hline \mathbf{b}_{\omega_{k+1}} - \mathbf{b}_{\omega_k} \\ \hline \mathbf{b}_{a_{k+1}} - \mathbf{b}_{a_k} \end{array}\right] \tag{3.29}$$

where we have employed $\bar{q}_b = \begin{bmatrix} \frac{\boldsymbol{\theta}}{||\boldsymbol{\theta}||}\ \sin\left(\frac{||\boldsymbol{\theta}||}{2}\right) \\ \cos\left(\frac{||\boldsymbol{\theta}||}{2}\right) \end{bmatrix}$ and $\boldsymbol{\theta} = \mathbf{J}_q\left(\mathbf{b}_{\omega_k} - \mathbf{b}^\star_{\omega_k}\right)$. In the above expressions, $\mathbf{J}_q$, $\mathbf{J}_\alpha$, $\mathbf{J}_\beta$, $\mathbf{H}_\alpha$, and $\mathbf{H}_\beta$, are the Jacobian matrices of the pertinent residuals with respect to the biases, which are used to correct the measurements due to a change in the initial bias estimate $\mathbf{b}^\star$, thus compensating for the fact that preintegrated measurements have been linearized about $\mathbf{b}^\star_{\omega_k}$ and $\mathbf{b}^\star_{a_k}$ without having to recompute the required integrals whenever the bias estimates change (see Equations (3.12) and (3.13)). In particular, using the fact that our preintegrated measurement means are linear in the acceleration bias $\mathbf{b}_a$ (see Equation (3.18)), we have the following dynamics of its Jacobians (see Equations (3.19) and (3.20)):

$$\begin{aligned} \begin{bmatrix} \frac{\partial \boldsymbol{\alpha}}{\partial \mathbf{b}_a} \\ \frac{\partial \boldsymbol{\beta}}{\partial \mathbf{b}_a} \end{bmatrix} &=: \begin{bmatrix} \mathbf{H}_\alpha(\tau+1) \\ \mathbf{H}_\beta(\tau+1) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{H}_\alpha(\tau) + \mathbf{H}_\beta(\tau)\Delta t - \mathbf{A}_\tau \\ \mathbf{H}_\beta(\tau) - \mathbf{B}_\tau \end{bmatrix} \end{aligned} \tag{3.30}$$

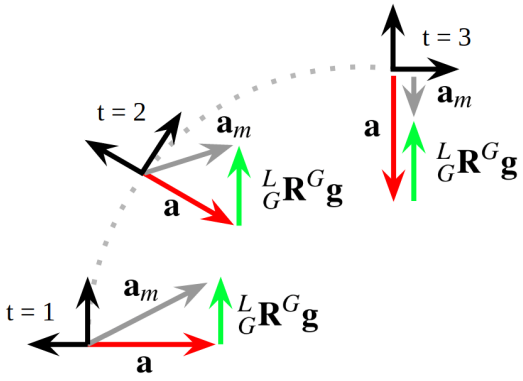


Figure 3.1: An example of an IMU rotating against gravity. It can be seen that the true local acceleration $\mathbf{a}$ (red) remains constant, while its local measurement $\mathbf{a}_m$ (grey) changes continuously due to the effect of gravity (green).

Similarly, for the gyroscope bias Jacobians, we have:

$$\begin{aligned}\begin{bmatrix}\frac{\partial \boldsymbol{\alpha}}{\partial \mathbf{b}_\omega}\\ \frac{\partial \boldsymbol{\beta}}{\partial \mathbf{b}_\omega}\end{bmatrix} &=: \begin{bmatrix}\mathbf{J}_\alpha(\tau+1)\\ \mathbf{J}_\beta(\tau+1)\end{bmatrix}\\ &= \begin{bmatrix}\mathbf{J}_\alpha(\tau)+\mathbf{J}_\beta(\tau)\Delta t+\frac{\partial \mathbf{A}_\tau \hat{\mathbf{a}}}{\partial \mathbf{b}_\omega}\\ \mathbf{J}_\beta(\tau)+\frac{\partial \mathbf{B}_\tau \hat{\mathbf{a}}}{\partial \mathbf{b}_\omega}\end{bmatrix}\end{aligned} \tag{3.31}$$

Finally, the orientation Jacobian with respect to gyroscope bias can be found incrementally as:

$$\mathbf{J}_q(\tau+1) = {}^{\tau+1}_{\tau}\breve{\mathbf{R}}\mathbf{J}_q(\tau) + \mathbf{J}_r(\hat{\boldsymbol{\omega}}\Delta t)\Delta t \tag{3.32}$$

where $\mathbf{J}_r(\cdot)$ is the right Jacobian of $SO(3)$ and is defined as [20]:

$$\mathbf{J}_r(\boldsymbol{\phi}) = \mathbf{I}_{3\times 3} - \frac{1-\cos(\parallel \boldsymbol{\phi} \parallel)}{\parallel \boldsymbol{\phi} \parallel^2}\lfloor \boldsymbol{\phi}\times \rfloor + \frac{\parallel \boldsymbol{\phi} \parallel - \sin(\parallel \boldsymbol{\phi} \parallel)}{\parallel \boldsymbol{\phi} \parallel^3}\lfloor \boldsymbol{\phi}\times \rfloor^2 \tag{3.33}$$

Moreover, the measurement Jacobians of these preintegrated measurements can also be computed analytically. For the detailed derivations and closed-form expressions of the preintegrated measurements and Jacobians, the reader is referred to Appendix A.

3.2.3 Model 2: Piecewise Constant Local Acceleration

The previous preintegration (Model 1) assumes that noiseless IMU measurements can be approximated as remaining constant over a sampling interval, which,

however, might not always be a good approximation (see Figure 3.1). In this section, we propose a new preintegration model that instead assumes piecewise constant *true* local acceleration during the sampling time interval, which may better approximate motion dynamics in practice. To this end, we first rewrite Equations (3.5) and (3.6) as:

$$ {}^{G}\mathbf{p}_{k+1} = {}^{G}\mathbf{p}_k + {}^{G}\mathbf{v}_k \Delta T + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R}\mathbf{a}\, duds \tag{3.34} $$

$$ {}^{G}\mathbf{v}_{k+1} = {}^{G}\mathbf{v}_k + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R}\mathbf{a}\, du \tag{3.35} $$

Note that we have moved the effect of gravity back inside the integrals. We then define the following vectors:

$$ \Delta\mathbf{p} = \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R}\mathbf{a}\, duds \tag{3.36} $$

$$ \Delta\mathbf{v} = \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R}\mathbf{a}\, du \tag{3.37} $$

which essentially are the true local position displacement and velocity change during $[t_k, t_{k+1}]$, and yields:

$$ \Delta\dot{\mathbf{p}} = \Delta\mathbf{v} \tag{3.38} $$

$$ \Delta\dot{\mathbf{v}} = {}^{k}_{u}\mathbf{R}\mathbf{a} \tag{3.39} $$

In particular, between two IMU measurement times inside the preintegration interval, $[t_\tau, t_{\tau+1}] \subset [t_k, t_{k+1}]$, we assume that the *local* acceleration will be constant:

$$ \forall t_u \in [t_\tau, t_{\tau+1}], \quad \mathbf{a}(t_u) = \mathbf{a}(t_\tau) \tag{3.40} $$

Using this sampling model we can rewrite (3.39) as:

$$ \Delta\dot{\mathbf{v}} = {}^{k}_{u}\mathbf{R}\left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a - {}^{\tau}_{k}\mathbf{R}\,{}^{k}_{G}\mathbf{R}\,{}^{G}\mathbf{g}\right) \tag{3.41} $$

We now write the relationship of the states at the beginning and end of the interval as (see (3.34) and (3.35)):

$$ {}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_k - {}^{G}\mathbf{v}_k \Delta T\right) = \Delta\mathbf{p} \tag{3.42} $$

$$ {}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_{k}\right) = \Delta\mathbf{v} \tag{3.43} $$

It is important to note that, since $\Delta\mathbf{p}$ and $\Delta\mathbf{v}$ are functions of both the biases *and* the initial orientation, we perform the following linearization with respect to these states:

$$ \begin{aligned} {}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_{k} - {}^{G}\mathbf{v}_{k}\Delta T\right) &\simeq \Delta\mathbf{p}\left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k}, {}^{k}_{G}\bar{q}^{\star}\right) \\ &+ \frac{\partial\Delta\mathbf{p}}{\partial\mathbf{b}_{\omega}}\Big|_{\mathbf{b}^{\star}_{\omega_k}}\Delta\mathbf{b}_{\omega} + \frac{\partial\Delta\mathbf{p}}{\partial\mathbf{b}_{a}}\Big|_{\mathbf{b}^{\star}_{a_k}}\Delta\mathbf{b}_{a} + \frac{\partial\Delta\mathbf{p}}{\partial\tilde{\boldsymbol{\theta}}_{k}}\Big|_{{}^{k}_{G}\bar{q}^{\star}}\Delta\boldsymbol{\theta}_{k} \end{aligned} $$

$$ {}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_{k}\right) \simeq \Delta\mathbf{v}\left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k}, {}^{k}_{G}\bar{q}^{\star}\right) \tag{3.44} $$

$$ + \frac{\partial\Delta\mathbf{v}}{\partial\mathbf{b}_{\omega}}\Big|_{\mathbf{b}^{\star}_{\omega_k}}\Delta\mathbf{b}_{\omega} + \frac{\partial\Delta\mathbf{v}}{\partial\mathbf{b}_{a}}\Big|_{\mathbf{b}^{\star}_{a_k}}\Delta\mathbf{b}_{a} + \frac{\partial\Delta\mathbf{v}}{\partial\Delta\boldsymbol{\theta}_{k}}\Big|_{{}^{k}_{G}\bar{q}^{\star}}\Delta\boldsymbol{\theta}_{k} \tag{3.45} $$

where $\Delta\boldsymbol{\theta}_{k} = 2\mathbf{vec}\left({}^{k}_{G}\bar{q} \otimes {}^{k}_{G}\bar{q}^{\star-1}\right)$ is the rotation angle change associated with the change of the linearization point of quaternion ${}^{k}_{G}\bar{q}$.

3.2.3.1 Computing Preintegration Mean:

To compute the new preintegrated measurement mean values, we first determine the continuous-time dynamics of the expected preintegration vectors by taking expectations of Equations (3.38) and (3.41), given by:

$$ \Delta\breve{\dot{\mathbf{p}}} = \Delta\breve{\mathbf{v}} \tag{3.46} $$

$$ \Delta\breve{\dot{\mathbf{v}}} = {}^{k}_{u}\breve{\mathbf{R}}\left(\mathbf{a}_{m} - \mathbf{b}^{\star}_{a_k} - {}^{\tau}_{k}\breve{\mathbf{R}}\,{}^{k}_{G}\mathbf{R}^{\star G}\mathbf{g}\right) \tag{3.47} $$

As in the case of Model 1 (see Section 3.2.2.1), we can formulate a linear system of the new preintegration measurement vectors and find the closed-from solutions. Specifically, we can integrate these differential equations and obtain the solution similar to Equation (3.18), while using the new definition: $\hat{\mathbf{a}} = \mathbf{a}_{m} - \mathbf{b}^{\star}_{a_k} - {}^{\tau}_{k}\breve{\mathbf{R}}\,{}^{k}_{G}\mathbf{R}^{\star G}\mathbf{g}$, which serves as the estimate for the piecewise constant local acceleration over the sampling interval.

3.2.3.2 Computing Preintegration Covariance:

To compute the new preintegration measurement covariance, we first determine the differential equations for the corresponding preintegration measurement errors (see Equations (3.38), (3.41), (3.46) and (3.47)):

$$\begin{aligned}
\Delta\dot{\tilde{\mathbf{p}}} &= \Delta\mathbf{v} - \Delta\breve{\mathbf{v}} = \Delta\tilde{\mathbf{v}} \\
\Delta\dot{\tilde{\mathbf{v}}} &= {}^{k}_{u}\breve{\mathbf{R}}\left(\mathbf{I} + \lfloor {}^{u}_{k}\tilde{\boldsymbol{\theta}}\times\rfloor\right)\left(\mathbf{a}_m - \mathbf{b}^{\star}_{a_k} - \tilde{\mathbf{b}}_a - \left(\mathbf{I} - \lfloor {}^{\tau}\tilde{\boldsymbol{\theta}}_k\times\rfloor\right){}^{\tau}_{k}\breve{\mathbf{R}}^{k}_{G}\mathbf{R}^{\star G}\mathbf{g} - \mathbf{n}_a\right) \\
&\quad - {}^{k}_{u}\breve{\mathbf{R}}\left(\mathbf{a}_m - \mathbf{b}^{\star}_{a_k} - {}^{\tau}_{k}\breve{\mathbf{R}}^{k}_{G}\mathbf{R}^{\star G}\mathbf{g}\right) \\
&= -{}^{k}_{u}\breve{\mathbf{R}}\lfloor\hat{\mathbf{a}}\times\rfloor{}^{u}_{k}\tilde{\boldsymbol{\theta}} - {}^{k}_{u}\breve{\mathbf{R}}\tilde{\mathbf{b}}_a - {}^{k}_{u}\breve{\mathbf{R}}\lfloor{}^{\tau}\breve{\mathbf{g}}\times\rfloor{}^{\tau}_{k}\tilde{\boldsymbol{\theta}} - {}^{k}_{u}\breve{\mathbf{R}}\mathbf{n}_a
\end{aligned} \tag{3.48}$$

where ${}^{\tau}\breve{\mathbf{g}}$ represents the estimate for gravity in the sampled τ frame. It is important to notice that, in the above expressions, we have used two angle errors: (i) ${}^{u}\tilde{\boldsymbol{\theta}}_k$ corresponds to the active local IMU orientation error, and (ii) ${}^{\tau}_{k}\tilde{\boldsymbol{\theta}}$ corresponds to the cloned orientation error at the sampling time t_τ. In addition, the bias errors $\tilde{\mathbf{b}}$ describe the deviation of the bias from the starting value over the interval due to bias drift. With this, we have the following time evolution of the full preintegrated measurement errors:

$$\begin{aligned}
&\begin{bmatrix} {}^{u}_{k}\dot{\tilde{\boldsymbol{\theta}}} \\ \Delta\dot{\tilde{\mathbf{p}}} \\ \Delta\dot{\tilde{\mathbf{v}}} \\ \dot{\tilde{\mathbf{b}}}_\omega \\ \dot{\tilde{\mathbf{b}}}_a \\ {}^{\tau}_{k}\dot{\tilde{\boldsymbol{\theta}}} \end{bmatrix} = \mathbf{F}\begin{bmatrix} {}^{u}_{k}\tilde{\boldsymbol{\theta}} \\ \Delta\tilde{\mathbf{p}} \\ \Delta\tilde{\mathbf{v}} \\ \tilde{\mathbf{b}}_\omega \\ \tilde{\mathbf{b}}_a \\ {}^{\tau}_{k}\tilde{\boldsymbol{\theta}} \end{bmatrix} + \begin{bmatrix} -\mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -{}^{k}_{u}\breve{\mathbf{R}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}\begin{bmatrix} \mathbf{n}_\omega \\ \mathbf{n}_a \\ \mathbf{n}_{\omega b} \\ \mathbf{n}_{ab} \end{bmatrix} \\
&\iff \quad \dot{\mathbf{r}} = \mathbf{F}\mathbf{r} + \mathbf{G}\mathbf{n}
\end{aligned} \tag{3.49}$$

where

$$\mathbf{F} = \begin{bmatrix} -\lfloor \hat{\boldsymbol{\omega}} \times \rfloor & \mathbf{0} & \mathbf{0} & -\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -{}^{k}_{u}\breve{\mathbf{R}} \lfloor \hat{\mathbf{a}} \times \rfloor & \mathbf{0} & \mathbf{0} & \mathbf{0} & -{}^{k}_{u}\breve{\mathbf{R}} & -{}^{k}_{u}\breve{\mathbf{R}} \lfloor {}^{\tau}\breve{\mathbf{g}} \times \rfloor \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3.50}$$

In analogy to Equations (3.23), (3.24), and (3.27), we can determine the new state-transition matrix $\boldsymbol{\Phi}(t_{\tau+1}, t_\tau)$ and the new discrete noise covariance $\mathbf{Q}_\tau$. With that, we now propagate the measurement covariance over the time interval $t_\tau \in [t_k, t_{k+1}]$ as follows:

$$\mathbf{P}_k = \mathbf{0} \tag{3.51}$$

$$\mathbf{P}_{\tau+1} = \boldsymbol{\Phi}(t_{\tau+1}, t_\tau)\mathbf{P}_\tau \boldsymbol{\Phi}(t_{\tau+1}, t_\tau)^\top + \mathbf{Q}_\tau \tag{3.52}$$

$$\mathbf{P}_{\tau+1} = \boldsymbol{\Gamma}\mathbf{P}_{\tau+1}\boldsymbol{\Gamma}^\top \tag{3.53}$$

where $\boldsymbol{\Gamma}$ is the permutation matrix that allows us to replace the previous static orientation error ${}^{\tau}_{k}\tilde{\boldsymbol{\theta}}$ by the new one ${}^{\tau+1}_{k}\tilde{\boldsymbol{\theta}}$ simply by cloning the current local orientation error ${}^{u}_{k}\tilde{\boldsymbol{\theta}}$ at the end of current sampling interval $t_u = t_{\tau+1}$ when moving from the current measurement time interval $[t_\tau, t_{\tau+1}]$ to the next one, and is given by:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \tag{3.54}$$

The resulting preintegrated measurement covariance is then extracted from the top left 15×15 block of $\mathbf{P}_{k+1}$ after the propagation with Equations (3.51)-(3.53) over the entire preintegration interval $[t_k, t_{k+1}]$.

3.2.3.3 Preintegration Measurement Residuals and Jacobians:

As we linearize this preintegration with respect to the IMU biases and the initial orientation, it is important to compute the Jacobians with respect to these quantities. In particular, we note that the solution to the preintegration equation for Model 2 can be expressed as:

$$\begin{aligned}\Delta\breve{\mathbf{p}}_{\tau+1} &= \Delta\breve{\mathbf{p}}_{\tau} + \Delta\breve{\mathbf{v}}_{\tau}\Delta t + \mathbf{A}_{\tau}\left(\mathbf{a}_m - \mathbf{b}_a^{\star} - {}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\right)\\ \Delta\breve{\mathbf{v}}_{\tau+1} &= \Delta\breve{\mathbf{v}}_{\tau} + \mathbf{B}_{\tau}\left(\mathbf{a}_m - \mathbf{b}_a^{\star} - {}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\right)\end{aligned} \tag{3.55}$$

where $\mathbf{A}_\tau$ and $\mathbf{B}_\tau$ are defined the same as in Equations (3.19) and (3.20). Letting $\mathbf{O}_\alpha$ and $\mathbf{O}_\beta$ denote the Jacobians of the position and velocity preintegrated measurements with respect to the initial orientation, we have:

$$\begin{bmatrix}\mathbf{J}_\alpha(\tau+1)\\ \mathbf{J}_\beta(\tau+1)\end{bmatrix} = \begin{bmatrix}\mathbf{J}_\alpha(\tau) + \mathbf{J}_\beta(\tau)\Delta t\\ \mathbf{J}_\beta(\tau)\end{bmatrix} + \begin{bmatrix}\frac{\partial\mathbf{A}_\tau\hat{\mathbf{a}}}{\partial\mathbf{b}_\omega} + \mathbf{A}_\tau\lfloor{}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\times\rfloor\mathbf{J}_q(\tau)\\ \frac{\partial\mathbf{B}_\tau\hat{\mathbf{a}}}{\partial\mathbf{b}_\omega} + \mathbf{B}_\tau\lfloor{}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\times\rfloor\mathbf{J}_q(\tau)\end{bmatrix}$$

$$\begin{bmatrix}\mathbf{H}_\alpha(\tau+1)\\ \mathbf{H}_\beta(\tau+1)\end{bmatrix} = \begin{bmatrix}\mathbf{H}_\alpha(\tau) + \mathbf{H}_\beta(\tau)\Delta t - \mathbf{A}_\tau\\ \mathbf{H}_\beta(\tau) - \mathbf{B}_\tau\end{bmatrix} \tag{3.56}$$

$$\begin{bmatrix}\mathbf{O}_\alpha(\tau+1)\\ \mathbf{O}_\beta(\tau+1)\end{bmatrix} = \begin{bmatrix}\mathbf{O}_\alpha(\tau) + \mathbf{O}_\beta(\tau)\Delta t\\ \mathbf{O}_\beta(\tau)\end{bmatrix} - \begin{bmatrix}\mathbf{A}_\tau{}_k^{\tau}\breve{\mathbf{R}}\lfloor{}_G^k\mathbf{R}^{\star G}\mathbf{g}\times\rfloor\\ \mathbf{B}_\tau{}_k^{\tau}\breve{\mathbf{R}}\lfloor{}_G^k\mathbf{R}^{\star G}\mathbf{g}\times\rfloor\end{bmatrix} \tag{3.57}$$

We note that Equation (3.57) reveals that only changes in the initial orientation *perpendicular* to local gravity (${}^k\mathbf{g}$) will cause a change in the preintegrated measurement. As these directions of orientation are observable and thus are expected to have small errors, this highlights the fact that our linearization scheme about the initial orientation is appropriate.

At this point, using these Jacobians, we can write the residual associated with

the new preintegrated IMU measurement as follows:

$$\mathbf{r}_{IMU}(\mathbf{x}) = \begin{bmatrix} 2\mathbf{vec}\left({}^{k+1}_{G}\bar{q} \otimes {}^{k}_{G}\bar{q}^{-1} \otimes {}^{k+1}_{k}\breve{\bar{q}}^{-1} \otimes \bar{q}_b\right) \\ \hline \left({}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_k - {}^{G}\mathbf{v}_k\Delta T\right) - \mathbf{J}_\alpha\left(\mathbf{b}_{\omega_k} - \mathbf{b}^{\star}_{\omega_k}\right) - \right. \\ \left. \mathbf{H}_\alpha\left(\mathbf{b}_{a_k} - \mathbf{b}^{\star}_{a_k}\right) - \mathbf{O}_\alpha\ 2\mathbf{vec}\left({}^{k}_{G}\bar{q} \otimes {}^{k}_{G}\bar{q}^{\star-1}\right) - \Delta\breve{\mathbf{p}}\right) \\ \hline \left({}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_k\right) - \mathbf{J}_\beta\left(\mathbf{b}_{\omega_k} - \mathbf{b}^{\star}_{\omega_k}\right) - \right. \\ \left. \mathbf{H}_\beta\left(\mathbf{b}_{a_k} - \mathbf{b}^{\star}_{a_k}\right) - \mathbf{O}_\beta\ 2\mathbf{vec}\left({}^{k}_{G}\bar{q} \otimes {}^{k}_{G}\bar{q}^{\star-1}\right) - \Delta\breve{\mathbf{v}}\right) \\ \hline \mathbf{b}_{\omega_{k+1}} - \mathbf{b}_{\omega_k} \\ \hline \mathbf{b}_{a_{k+1}} - \mathbf{b}_{a_k} \end{bmatrix} \tag{3.58}$$

The resulting measurement Jacobians are necessary for a batch optimization, which we analytically computed as shown in Appendix A.

3.3 Visual-Inertial Navigation

To demonstrate the applicability of the proposed closed-form preintegration (CPI) theory presented in the preceding section, in this section, we develop two sliding-window optimization-based sensor fusion schemes for visual-inertial navigation systems (VINS) that utilize our inertial preintegration.

3.3.1 Tightly-Coupled Indirect VIO

As an IMU-camera sensor suite moves through an unknown environment, visual feature keypoints can be extracted and tracked from the images to provide motion information about the platform. We refer the reader to Section 2.1.3 for details on the measurements involved with our indirect system.

In order to maintain only a sliding window of states, rather than estimating the entire trajectory, we utilize marginalization, as explained in Appendix B. Using all the

visual measurements available in a sliding window along with the preintegrated IMU measurements and marginalization prior, we solve the following optimization problem that tightly couples all available measurement residuals:

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \Big(||\mathbf{r}_{marg}(\mathbf{x})||_2^2 + \sum_{p \in \mathcal{P}} ||\mathbf{r}_{IMU}(\mathbf{x})||^2_{\mathbf{P}_p^{-1}} \tag{3.59}$$

$$+ \sum_{(f,j,k) \in \mathcal{C}} ||\mathbf{r}_f(\mathbf{x})||^2_{\mathbf{R}_f^{-1}} \Big) \tag{3.60}$$

where $\mathbf{r}_f$ is the measurement residual of the corresponding indirect feature measurement and $\mathbf{R}_k$ is the measurement noise covariance. In addition, $\mathcal{C}$ and $\mathcal{P}$ are the set of feature and preintegrated measurements, respectively, while $\mathbf{r}_{marg}(\mathbf{x})$ is the residual of the marginal prior (see Equation (B.8)). We want to point out again that in practice we instead employ a (Huber or Cauchy) robust cost function on the last visual error term in Equation (3.59), while we here omit the detailed derivations of this standard treatment to keep presentation concise, we do have a similar treatment in our ensuing loosely-coupled direct VINS (see Equation (3.65)).

3.3.1.1 Inverse-depth Representation:

A well-known disadvantage of the above representation for features is that points at infinity are difficult to utilize. To mitigate this issue, we instead employ an *inverse-depth* representation [22]. In particular, we represent a feature using the inverse coordinates in the camera frame where it was first observed. Denoting $\{C_{a,i}\}$ the frame of reference of the "anchoring" camera, which is associated with the i-th camera frame and the anchoring time a, we have the following inverse-depth representation (see [94]):

$${}^{C_{a,i}}\mathbf{m}_f = \begin{bmatrix} \alpha \\ \beta \\ \rho \end{bmatrix} \quad \Rightarrow \quad {}^{C_{a,i}}\mathbf{p}_f = \frac{1}{\rho} \begin{bmatrix} \alpha \\ \beta \\ 1 \end{bmatrix} \tag{3.61}$$

where we also show the relationship between the inverse-depth representation of the feature ${}^{C_{a,i}}\mathbf{m}_f$ and the corresponding 3D position in the anchor frame ${}^{C_{a,i}}\mathbf{p}_f$. The feature position in the j-th camera frame at time step k can be computed as follows:

$$
\begin{aligned}
{}^{C_{k,j}}\mathbf{p}_f &= {}^{C_{k,j}}_{C_{a,i}}\mathbf{R}\frac{1}{\rho}\begin{bmatrix}\alpha\\ \beta\\ 1\end{bmatrix} + {}^{C_{k,j}}\mathbf{p}_{C_{a,i}} \qquad (3.62)\\
{}^{C_{k,j}}_{C_{a,i}}\mathbf{R} &= {}^{C_j}_{I}\mathbf{R}\,{}^{k}_{G}\mathbf{R}\,{}^{a}_{G}\mathbf{R}^{\top}{}^{C_i}_{I}\mathbf{R}^{\top}\\
{}^{C_{k,j}}\mathbf{p}_{C_{a,i}} &= {}^{C_j}_{I}\mathbf{R}\,{}^{k}_{G}\mathbf{R}\big({}^{G}\mathbf{p}_a + {}^{a}_{G}\mathbf{R}^{\top I}\mathbf{p}_{C_i} - {}^{G}\mathbf{p}_k\big) + {}^{C_j}\mathbf{p}_I
\end{aligned}
$$

Note that due to the projective geometry of the perspective projection (2.44), $\mathbf{\Pi}(\mathbf{x}) = \mathbf{\Pi}(\rho\mathbf{x})$, we can multiply both sides of Equation (3.62) by ρ and have the equivalent measurement model:

$$\mathbf{z}_{fjk} = \mathbf{\Pi}\left(\mathbf{h}\right) + \mathbf{n}_f \qquad (3.63)$$

$$\mathbf{h} = \begin{bmatrix}h_1\\ h_2\\ h_3\end{bmatrix} := \rho^{C_{k,j}}\mathbf{p}_f = {}^{C_{k,j}}_{C_{a,i}}\mathbf{R}\begin{bmatrix}\alpha\\ \beta\\ 1\end{bmatrix} + \rho^{C_{k,j}}\mathbf{p}_{C_{a,i}} \qquad (3.64)$$

The measurement Jacobians of this inverse-depth model can be found in Appendix C. Note that this measurement model is numerically stable and can handle points at infinity, thus allowing for the gain of feature direction information from these far-off feature points.

3.3.2 Loosely-Coupled Direct VINS

To further validate the proposed closed-form preintegration theory, in the following, by leveraging our prior work [30], we develop a *loosely-coupled* VINS algorithm based on direct image alignment and IMU preintegration. In particular, we estimate the relative frame-to-frame motion through direct alignment of image pixels. These relative-motion constraints then allow us to efficiently perform loop closure without explicitly detecting/tracking (or matching) features.

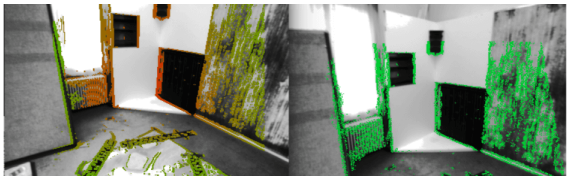


Figure 3.2: Visualization of selected depth map pixels with a large intensity gradient (left). Keyframe pixels are projected onto the query frame as a result of the optimized direct alignment of the frame-to-frame relative transformation (right).

Consider the case where we wish to directly align a current frame C_2 against a keyframe C_1 (see Figure 3.2). Finding the optimal transformation can be formulated as an optimization problem over the total (warped) pixel intensity difference (i.e., photometric error):

$$\underset{C_1}{\overset{C_2}{\breve{\mathbf{T}}}} = \underset{\underset{C_1}{\overset{C_2}{\mathbf{T}}}}{\operatorname{argmin}} \sum_f \gamma \left(\underbrace{\frac{1}{\sigma_r^2} \left(\underbrace{I_{C_2}\left(\underset{C_1}{\overset{C_2}{\mathbf{T}}}\, {}^{C_1}\mathbf{p}_f\right) - I_{C_1}\left({}^{C_1}\mathbf{p}_f\right)}_{e_f} \right)^2}_{v_f} \right) \tag{3.65}$$

where $\underset{C_1}{\overset{C_2}{\mathbf{T}}}$ is the transformation between the two camera frames parameterized by the relative quaternion $\underset{C_1}{\overset{C_2}{\bar{q}}}$ and relative position ${}^{C_1}\mathbf{p}_{C_2}$, while $I_{C_i}(\cdot)$ returns the intensity of a given point projected into the image frame, and $\gamma(\cdot)$ is the Huber cost. The pixel's position in the keyframe, ${}^{C_1}\mathbf{p}_f$ can be found via an online or stereo pair depth map computation. This position is treated as a noisy parameter in the residual allowing for computation of the residual sigma, σ_r, with the summation being over all pixels f with valid depth estimates and high gradients along the epipolar line. The Huber cost function $\gamma(\cdot)$ with parameter k is defined as [27]:

$$\gamma(r) = \begin{cases} r, & \text{if } r < k^2 \\ 2k\sqrt{r} - k^2, & \text{otherwise} \end{cases} \tag{3.66}$$

The purpose of the Huber cost is to down-weight large residuals which occur naturally in image alignment due to occlusions, and has been used extensively in the literature (e.g., [39]).

Note that the covariance of each residual σ_r^2 encodes the uncertainty due to errors in the intensity measurements as well as the disparity map:

$$\sigma_r^2 = 2\sigma_{int}^2 + \left(\frac{\partial e_f}{\partial d}\right)^2 \sigma_d^2 \tag{3.67}$$

where σ_{int}^2 denotes the covariance of the intensity reading, $\frac{\partial e_f}{\partial d}$ is the Jacobian of the residual e_f (3.65) with respect to the measured disparity d, and σ_d^2 is the covariance of the disparity measurement. In the case of a depth map computed from a stereo pair as considered in this work, we define $\mathbf{t}$ as the pixel coordinates, z as the pixel depth, and b as the baseline between the stereo pair. The Jacobian $\frac{\partial e_f}{\partial d}$ can be calculated using the chain rule of differentiation as follows (see Equation (3.65)):

$$\begin{aligned}\frac{\partial e_f}{\partial d} =& \frac{\partial I_{C_2}}{\partial \mathbf{t}} \frac{\partial \mathbf{t}}{\partial {}^{C_2}\mathbf{p}_f} \frac{\partial {}^{C_2}\mathbf{p}_f}{\partial {}^{C_1}\mathbf{p}_f} \frac{\partial {}^{C_1}\mathbf{p}_f}{\partial z} \frac{\partial z}{\partial d} \\ =& \begin{bmatrix} I_{C_{2_x}} & I_{C_{2_y}} \end{bmatrix} \begin{bmatrix} \frac{f_x}{{}^{C_2}\mathbf{p}_{f_j}(3)} & 0 & -\frac{f_x\, {}^{C_2}\mathbf{p}_{f_j}(1)}{{}^{C_2}\mathbf{p}_{f_j}(3)^2} \\ 0 & \frac{f_y}{{}^{C_2}\mathbf{p}_{f_j}(3)} & -\frac{f_y\, {}^{C_2}\mathbf{p}_{f_j}(2)}{{}^{C_2}\mathbf{p}_{f_j}(3)^2} \end{bmatrix} {}^{C_2}_{C_1}\mathbf{R} \frac{{}^{C_1}\mathbf{p}_{f_j}}{z} \frac{-f_x b}{d^2}\end{aligned} \tag{3.68}$$

where $I_{C_{2_x}}$ and $I_{C_{2_y}}$ are the image gradients in the x and y directions respectively, while f_x and f_y are the focal lengths of the camera.

The covariance of the pixel disparity, σ_d^2, is obtained based on the observation that this disparity is the maximum likelihood estimate for a single measurement graph, with the residual being the difference in intensity between the pixel in the left, I_{C_1L}, and right, I_{C_1R}, images in the keyframe stereo pair, which can be formulated as follows:

$$\breve{d} = \underset{d}{\operatorname{argmin}} \frac{1}{\sigma_{rd}^2} \Big(\underbrace{I_{C_1L}(v,u) - I_{C_1R}(v,u-d)}_{e_d} \Big)^2 \tag{3.69}$$

where the covariance associated with this residual can be found as $\sigma_{rd}^2 = 2\sigma_{int}^2$, and comes from uncertainty in the intensity readings. The covariance on our disparity estimate can then be approximated as:

$$\sigma_d^2 = \left(\frac{\partial e_d}{\partial d}^2 \frac{1}{\sigma_{rd}^2} \right)^{-1} = \sigma_{rd}^2 \left(\frac{1}{I_{C_1R_x}} \right)^2 \tag{3.70}$$

where $I_{C_1R_x}$ is the x-gradient of the pixel in the right image which is selected as the match.

Once we have determined the photometric error covariance σ_r^2, we now solve the direct alignment problem (3.65) using the Levenberg-Marquadt method. In particular, at each iteration we solve the following normal equation:

$$\left(\left(\sum_f w_f \mathbf{J}_f^\top \mathbf{J}_f\right) + \lambda \text{diag}\left(\sum_f w_f \mathbf{J}_f^\top \mathbf{J}_f\right)\right) {}^{C_2}_{C_1}\tilde{\mathbf{T}} = -\sum_f w_f \mathbf{J}_f^\top e_f\left({}^{C_2}_{C_1}\breve{\mathbf{T}}\right) \tag{3.71}$$

where λ is the damping parameter, and $e_f({}^{C_2}_{C_1}\breve{\mathbf{T}})$ is the residual due to the f-th pixel in the alignment, evaluated at the current estimate (linearization point) for the relative transformation, ${}^{C_2}_{C_1}\breve{\mathbf{T}}$. The weight w_f is computed at each iteration as follows:

$$w_f = \frac{\partial \gamma(v_f)}{\partial v_f}\frac{1}{\sigma_f^2} \tag{3.72}$$

$$\frac{\partial \gamma(v_f)}{\partial v_f} = \begin{cases} 1, & \text{if } v_f < k^2 \\ \frac{k}{\sqrt{v_f}}, & \text{otherwise} \end{cases} \tag{3.73}$$

where v_f is the raw cost fed into the Huber norm (see Equation (3.65)), and k is a design parameter. The Jacobian matrix $\mathbf{J}_f$ is of the direct alignment measurement residual with respect to the state, computed as:

$$\mathbf{J}_f = \begin{bmatrix} I_{C_{2_x}} & I_{C_{2_y}} \end{bmatrix} \begin{bmatrix} \frac{f_x}{{}^{C_2}\mathbf{p}_f(3)} & 0 & -\frac{f_x\,{}^{C_2}\mathbf{p}_f(1)}{({}^{C_2}\mathbf{p}_f(3))^2} \\ 0 & \frac{f_y}{{}^{C_2}\mathbf{p}_f(3)} & -\frac{f_y\,{}^{C_2}\mathbf{p}_f(2)}{({}^{C_2}\mathbf{p}_f(3))^2} \end{bmatrix} \begin{bmatrix} \lfloor {}^{C_2}\mathbf{p}_f \times \rfloor & -{}^{C_2}_{C_1}\mathbf{R} \end{bmatrix} \tag{3.74}$$

After optimization, we will be left with a Gaussian distribution on our estimated relative *camera* pose. We can then transform this into a distribution on the relative *IMU* pose (denoted k and j for the keyframe and query frame IMU states respectively) using covariance propagation:

$${}^{C_2}_{C_1}\mathbf{T} = {}^{C_2}_{C_1}\breve{\mathbf{T}} \boxplus {}^{C_2}_{C_1}\tilde{\mathbf{T}}, \quad \text{where } {}^{C_2}_{C_1}\tilde{\mathbf{T}} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_c) \tag{3.75}$$

$${}^{j}_{k}\mathbf{T} = {}^{j}_{k}\breve{\mathbf{T}} \boxplus {}^{j}_{k}\tilde{\mathbf{T}}, \quad \text{where } {}^{j}_{k}\tilde{\mathbf{T}} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_i) \tag{3.76}$$

$$\mathbf{\Sigma}_i = \frac{\partial {}^{j}_{k}\tilde{\mathbf{T}}}{\partial {}^{C_2}_{C_1}\tilde{\mathbf{T}}} \mathbf{\Sigma}_c \frac{\partial {}^{j}_{k}\tilde{\mathbf{T}}}{\partial {}^{C_2}_{C_1}\tilde{\mathbf{T}}}^\top \tag{3.77}$$

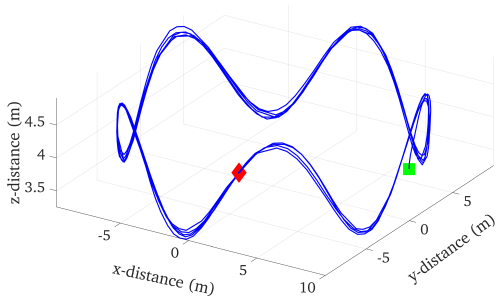


Figure 3.3: The ground truth trajectory of a MAV flying in a circle sinusoidal path generated in the Gazebo simulator. The total trajectory length is 307 meters with an average velocity of 6.13 m/s. Start and end positions are denoted with a green square and red diamond, respectively.

where $\mathbf{\Sigma}_c = (\sum_f w_f \mathbf{J}_f^\top \mathbf{J}_f)^{-1}$ is the covariance of the zero-mean alignment error. From this, the relative pose measurement that connects the IMU keyframe and query frame has the following residual:

$$\mathbf{r}_d(\mathbf{x}) = \begin{bmatrix} 2\mathbf{vec}\left({}^j_G\bar{q} \otimes {}^k_G\bar{q}^{-1} \otimes {}^j_k\breve{\bar{q}}^{-1}\right) \\ \\ {}^k_G\mathbf{R}\left({}^G\mathbf{p}_j - {}^G\mathbf{p}_k\right) - {}^k\breve{\mathbf{p}}_j \end{bmatrix} \tag{3.78}$$

whose Jacobians with respect to the state are provided in Appendix D, which will be used during graph optimization.

Using this visual measurement residual, along with the preintegrated IMU measurements, we have the following optimization problem for the loosely-coupled direct VINS, which can be solved analogously as in Equation (3.59):

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{d \in \mathcal{D}} ||\mathbf{r}_d(\mathbf{x})||^2_{\mathbf{\Sigma}_i^{-1}} + \sum_{p \in \mathcal{P}} ||\mathbf{r}_{IMU}(\mathbf{x})||^2_{\mathbf{P}_p^{-1}} \tag{3.79}$$

where $\mathcal{D}$ and $\mathcal{P}$ are the set of direct alignment relative pose and preintegrated measurements, respectively. Note that this direct image alignment allows for computationally efficient incorporation of large-scale loop closures due to the direct compression of intensity residuals into a single informative relative motion measurement.

Table 3.1: Analysis of the effect of different IMU frequencies on estimation accuracy. Note that each frequency run has a slightly different trajectory, and thus only the relative spread within a given frequency should be considered.

	MODEL-1		**MODEL-2**		**DISCRETE**	
Units	m	deg	m	deg	m	deg
100 Hz	0.096	0.327	**0.093**	**0.300**	0.107	0.328
200 Hz	0.051	0.204	**0.049**	**0.179**	0.058	0.207
400 Hz	0.033	0.107	**0.033**	**0.101**	0.035	0.109
800 Hz	**0.030**	0.085	0.030	**0.085**	0.031	0.086

3.4 Monte-Carlo Simulation Analysis

To validate the proposed closed-form preintegration theory, we first perform extensive Monte-Carlo simulations in various conditions in terms of sampling rates and motion dynamics. In particular, to better model the motion dynamics of a physical system, we leverage the open-source Gazebo simulator of a micro air vehicle (MAV) [71] which allows for direct realistic simulation and collection of true inertial and pose data (constrained by the physical MAV motion). The simulated datasets were generated as follows: (i) the MAV was commanded to follow a series of waypoints after takeoff, (ii) the ground truth of 100 Hz inertial and pose information was recorded, (iii) 80 synthetic stereo visual feature measurements (uv-coordinates) were created for each camera frame using the true pose information at a static 10Hz frequency. Following the commonly-used IMU model [118], the true inertial measurements were corrupted with an additive discrete bias and white noise using the noise parameters from the VI-Sensor [98], while the features' uv-coordinates were corrupted with an additive white noise to each axis with one pixel standard deviation.

In our tests, we used the popular GTSAM [24] framework to construct, optimize, and marginalize our graph using the included fixed-lag smoother. To ensure a fair comparison, we evaluate our preintegration methods against the state-of-art *discrete* preintegration [42], by using the on-manifold preintegrator class within the GTSAM implementation to compute the required measurement means, bias Jacobians, and

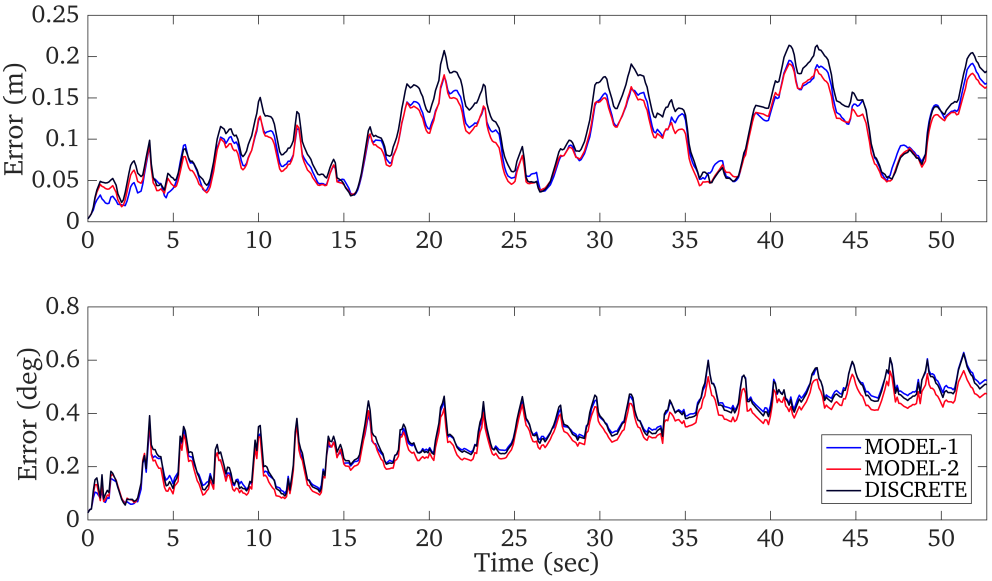


Figure 3.4: Monte-Carlo simulation results averaged over 50 runs: (top) position RMSE, and (bottom) orientation RMSE. In this test, physically-realistic synthetic data was generated using a Gazebo MAV simulator. It is clear that the proposed closed-form preintegration outperforms the state-of-the-art discrete approach [42].

covariances. We constructed all graphs side by side, ensuring that the measurements inserted are exactly the same, and thus fair to all methods. For simplicity we used the tightly-coupled indirect features in the graph, in which features are automatically marginalized out after three seconds (that is, no map was created for loop closures, and thus, the system is a visual-inertial odometry (VIO) system). Note also that we initialized all systems to the ground-truth pose, and with zero bias. Figure 3.3 shows the true simulated trajectory generated using Gazebo in our simulations.

The Monte-Carlo simulation comparison results of root mean squared error (RMSE) averaged over 50 runs are shown in Figure 3.4. Evidently, the proposed preintegration using piecewise constant local acceleration model (i.e., **Model 2**) is slightly better with an RMSE (averaged over all time steps and all runs) of 0.093 meters and 0.300 degrees than that using the piecewise constant measurement model (i.e., **Model 1**) with an RMSE of 0.096 meters and 0.327 degrees. More importantly, both methods are shown to outperform the discrete state-of-the-art method [42], which has an RMSE

of 0.107 meters and 0.328 degrees. It is important to point out that the superior performance (though by a small margin in this MAV test, with larger improvement margins expected for higher dynamics not constrained by MAV motion) endowed by the proposed closed-form preintegration using the new inertial models over the discrete one does not incur extra computational overhead during graph-based VINS optimization.

Furthermore, we investigate the effect that the IMU frequency has on the relative performance of the preintegration methods under consideration. Using the same simulation setup, the MAV was commanded to follow the trajectory with different Gazebo simulation frequencies. It is important to note that since we are using a physical simulation, the true trajectory will vary from frequency to frequency since the controller will perform differently, however, this is acceptable since we are looking at the relative performance within a given frequency. Table 3.1 shows the averaged RMSE results of different IMU frequencies. It can be seen that the proposed closed-form preintegration methods have greater impact when the frequency of the IMU is lower (i.e., significantly better performance); while at higher frequencies, the above methods become less distinguishable. This implies that the proposed closed-form preintegration methods are better suited for applications with limited IMU frequency, which is often the case for low-cost MEMS sensors on resource-constrained devices.

3.5 Real-World Experimental Validations

3.5.1 Tightly-Coupled Indirect VIO

In our tightly-coupled VIO system, we use stereo vision due to its superior estimation performance as compared to monocular systems [102]. When a pair of stereo images arrive, we perform KLT tracking [6] of FAST [109] features that have been extracted in an uniform grid over the image. Stereo correspondence information is known by initializing new features in the left image and KLT tracking them into the right. The set of stereo tracks from the current image is then tracked temporally forward at each future time step, while also ensuring to initialize new feature tracks if the number of active tracks falls under our desired active feature threshold. To reject

outliers we perform 8-point RANSAC between both the temporal and stereo left-to-right matches. We have found that this frontend provides a good balance between track longevity, computational speed, and accuracy. If an active feature is successfully tracked, the normalized image coordinates are added as measurements associated with that feature. To robustify our system to outliers, we utilized the Cauchy loss function for all image measurements.

Inspired by Leutenegger et al., [75], we maintain a sliding window of IMU states in the estimator that consists of two sub-windows. The first, denoted as the inertial window, contains the full 15 DOF IMU state and refers to the most recent imaging times. The second window, called the pose window, contains a set of pose-only clones (that is, only the orientation and position are maintained). At every imaging time we create a new corresponding IMU node. The IMU readings collected over the interval are preintegrated to both predict the new state and to form a preintegrated IMU measurement between the previous and new state.

After tracking, we formulate the sliding-window batch optimization (i.e., BA) problem using all features with a sufficient number of tracks as well as all nodes in the inertial and pose windows. The measurements contained in this graph are: (i) the prior, (ii) the visual measurements for the active features, and (iii) the preintegration factors between the inertial window states (see Equation (3.59)). We use the Ceres Solver with an elimination ordering that takes advantage of the sparsity of the problem through the Schur Complement [1, 74].

If the inertial window has reached its maximum length, we flag the oldest state's velocity and biases for marginalization. If the pose window also reaches its maximum length, we add both the oldest pose and all features it has seen into the marginal state list. Performing marginalization yields a new prior factor that has absorbed the old prior, the marginalized feature measurements, and the oldest preintegration factor. The oldest IMU state whose velocity and biases have been marginalized is then moved into the pose window.

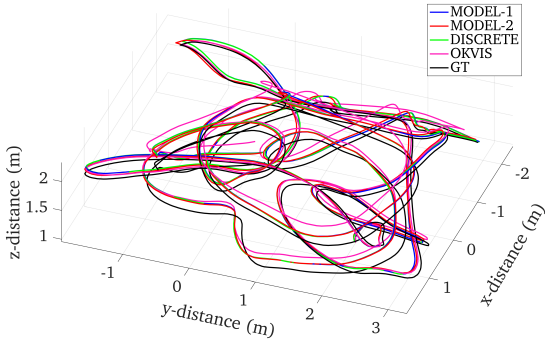
Table 3.2: Average absolute RMSE results of the tightly-coupled indirect VINS for the EuRoC MAV sequences averaged over 10 runs. All systems were initialized with the ground truth state. The smallest position and orientation errors have been highlighted.

	MODEL-1		**MODEL-2**		**DISCRETE**		**OKVIS**	
Units	m	deg	m	deg	m	deg	m	deg
V1_01_easy	0.2522	2.749	**0.2160**	2.503	0.2547	2.781	0.2356	**2.458**
V1_02_med	0.1342	**0.942**	**0.1214**	1.215	0.1344	1.001	0.1996	2.321
V1_03_diff	0.1101	0.880	**0.0953**	**0.809**	0.1012	0.830	0.1830	3.498
V2_01_easy	0.1429	1.069	**0.1426**	1.148	**0.1426**	1.118	0.1806	**0.973**
V2_02_med	0.1297	1.390	**0.1223**	**1.135**	0.1375	1.450	0.1695	2.334
V2_03_diff	0.2982	2.159	**0.2800**	**1.769**	0.3055	2.052	0.3483	8.327
MH_01_easy	0.1817	1.398	**0.1653**	1.761	0.2050	1.321	0.2523	**0.728**
MH_02_easy	0.1533	0.691	**0.1498**	**0.525**	0.1564	0.599	0.2523	0.728
MH_03_med	0.2993	1.024	**0.2627**	0.968	0.2800	**0.840**	0.3193	1.903
MH_04_diff	0.3312	**0.849**	0.3515	0.974	0.3488	0.852	**0.2145**	1.022
MH_05_diff	0.3939	**0.692**	0.3971	0.715	**0.3835**	0.809	0.5432	0.738

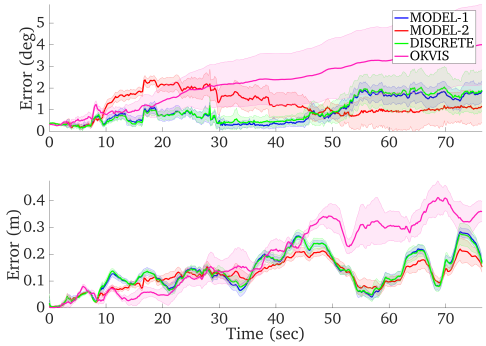
3.5.1.1 EuRoC MAV Dataset:

We compared our tightly-coupled indirect VIO system with a state-of-the-art open-source VINS: the Open Keyframe-based Visual-Inertial SLAM (OKVIS) [75], although several different VINS methods were recently introduced (e.g., [9]). It should be noted that the well-known open-source implementation of OKVIS[2] in fact employs the method of inertial preintegration, while only triggering full reintegration if the linearization point changes sufficiently and thus improving the efficiency. We performed this comparison on the EuRoC MAV dataset [14], which has become the standard method for evaluating VINS algorithms and provides 20hz stereo pairs with a 200hz MEMS ADIS16448 IMU. Our tightly-coupled preintegration-based system was run with inertial and pose sliding windows of 6 and 8 with a maximum of 300 extracted features. Stereo-OKVIS was run with 4 and 6 inertial and keyframes with 300 features. These parameters were selected to ensure real-time performance with both systems having minimal dropped frames. It should be noted that depending on the tuning

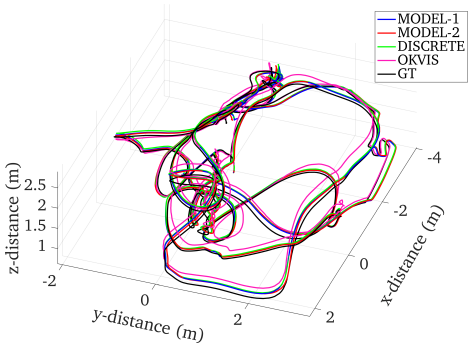
[2] https://github.com/ethz-asl/okvis



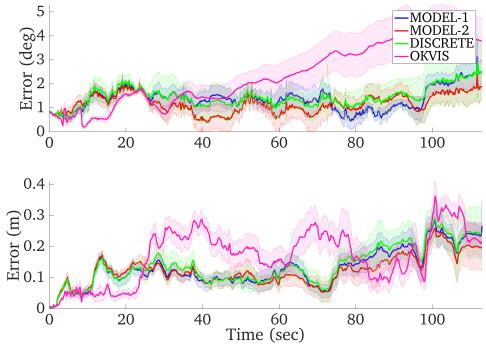
(a) Average trajectory estimates for "V1_02_med".



(b) Average position and orientation RMSE for "V1_02_med".



(c) Average trajectory estimates for "V2_02_med".



(d) Average position and orientation RMSE for "V2_02_med".

Figure 3.5: Average trajectory and RMSE error over ten runs for the "V1_02_med" (top) and "V2_02_med" (bottom) sequences of the proposed tightly-coupled indirect VIO system. The one-sigma bound on the mean error is also shown and can be interpreted as the repeatability of the system (due to some randomness occurred in visual tracking). Note that this is not the same as estimator uncertainty and instead shows the variance of the VIO systems. The total trajectory lengths are 80 and 88 meters, respectively.

parameters used in the VINS algorithms, their performance may vary (e.g., see [25]). Note also that our VIO system uses a sliding window of poses as well as the inertial window connected with preintegrated measurements but does *not* keep any kind of map (to allow intra-window loop closures), while OKVIS employs a set of keyframes where mapped points are maintained. Nevertheless, to provide a direct comparison, we initialize both systems with the true orientation, biases, velocity, and position such that no post-processing yaw alignment is needed. Note also that due to some randomness that may occur during the visual tracking frontend (e.g., RANSAC-based outlier rejection), variations in the VINS results can be observed even if running the same algorithm on the same sequences. To limit this variability of the algorithm, we perform 10 runs on the real-world sequences and average the results.

The "V1_02_med" and "V2_02_med" average trajectories can be seen in Figures 3.5a and 3.5c where we plot the estimated trajectories of our VIO and OKVIS along with the ground truth. Figures 3.5b and 3.5d show the averaged RMSE results of our VIO algorithm based on the proposed closed-form preintegration with the two models as compared to OKVIS, which were computed at every time step and then averaged over all the runs; while the averaged RMSE results are shown in Table 3.2. To show the repeatability/variability between runs, we also plot the standard deviation of the runs, noting that this should not be confused with the estimator uncertainty bounds commonly found in the literature.

We additionally evaluated the trajectories of the proposed models using the odometry error metric [132]. As compared to the absolute RMSE value, this metric splits the trajectory into small segments of predetermined lengths, aligns the start of each segment to the ground truth, and then computes the error of the ending pose of the segment in respect to the ground truth. This allows for insights of how drift is a function of distance. Following the method proposed by [132], each of the ten runs performed by each model on the EuRoC MAV sequences were evaluated and the total odometric error over all sequences was computed. Figure 3.6 and Table 3.3, show the resulting odometric error for trajectory segments of {7,14,21,28,35} meters.

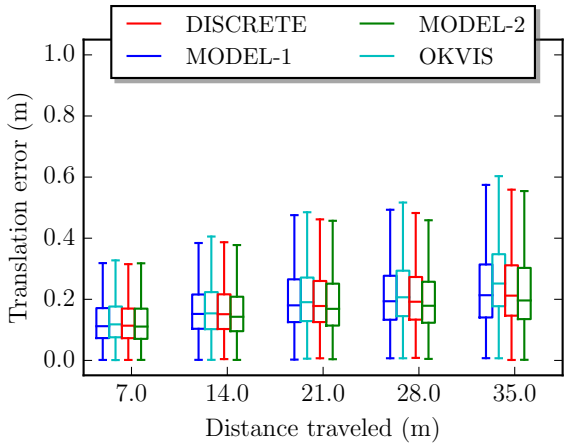


Figure 3.6: Boxplot of the odometric translation error statistics for the tightly-coupled indirect system evaluated over all of the EuRoC MAV sequences. Errors were computed using the odometry metric over trajectory segments of {7,14,21,28,35} meters in length. The middle box spans the first and third quartiles, while the whiskers are the upper and lower limits.

Table 3.3: Mean odometric translation errors for the tightly-coupled indirect system evaluated over all of the EuRoC MAV sequences. Errors were evaluated over trajectory segments of {7,14,21,28,35} meters in length. All errors are in meters.

	MODEL-1	**MODEL-2**	**DISCRETE**	**OKVIS**
7 m	0.137	**0.135**	0.136	0.142
14 m	0.177	**0.169**	0.177	0.185
21 m	0.220	**0.209**	0.217	0.226
28 m	0.229	**0.216**	0.226	0.245
35 m	0.247	**0.238**	0.246	0.287

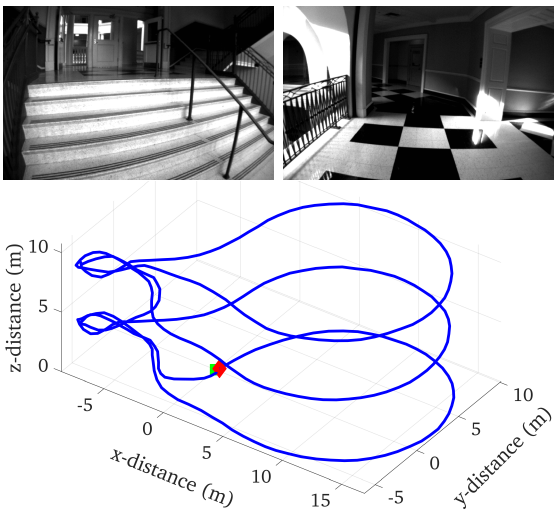
These results clearly demonstrate that our VIO system can offer competitive performance to OKVIS; that is, we see instances where our method outperforms OKVIS, while in others OKVIS is superior. Between the two proposed preintegration models, for these experiments, Model 2 offers the best performance. These results clearly validate the accuracy gains from our advanced preintegration modeling.

3.5.1.2 UD Indoor Datasets:

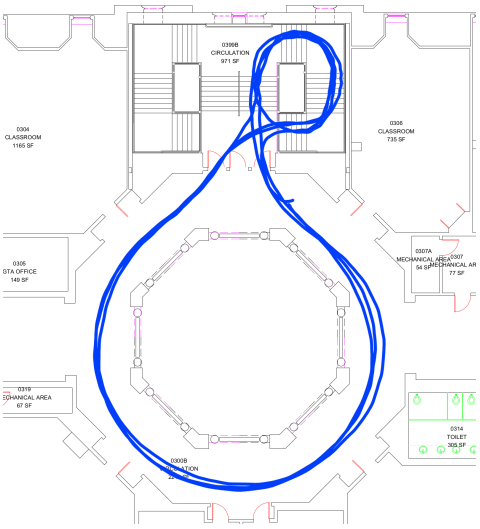
We further performed relatively large-scale (as compared to the EuRoC MAV dataset) indoor experiments in two buildings at the University of Delaware (UD) using

our hand-held VI-Sensor with an IMU frequency of 400 Hz. In these experiments, because no ground truth was available, we initialized the system by keeping the device stationary for a short period of time (e.g., 2 seconds) so that the initial orientation and biases could be found, while the position and velocities were initialized as zero. To account for poor calibration of the sensor suite, both the IMU-to-camera spatial calibration parameters as well as the camera *intrinsics* were estimated online. This was done by adding these quantities into the state and using the *raw* image coordinates as measurements, while expressing these as a function of the normalized pixel coordinates as well as the camera intrinsics [80]. See Section 4.2 for a further discussion. The first indoor experiment was performed in the UD Gore Hall, in which the trajectory starts on the first floor, traverses the staircase to the third floor, and returns to the bottom floor, making a loop on each level. To evaluate the estimation performance the trajectory returns to the original starting location. The 3D trajectory estimate is shown in Figure 3.7a while its projection onto the building floor plan is shown in Figure 3.7b. We ran each preintegration model ten times across the 228 meter long dataset and averaged the results. Model 1 had an ending error of 0.763 m (0.33% of the path), Model 2 had an ending error 0.747 m (0.33%), discrete preintegration achieved 0.765 (0.34%), and OKVIS achieved an ending error of 0.762 m (0.33%) showing the improvement (although slight) due to closed-form preintegration.

The second indoor experiment was conducted in the UD Smith Hall. Starting on the second floor, we traversed along a rectangular wall before descending the stairs, looping around the first floor, then returning up the stairs, looping one and a half times around the upper level before returning to the starting position. Model 1 had an ending error of 0.632 m (0.28% of the path), Model 2 had an ending error 0.788 m (0.35%), discrete preintegration achieved 0.768 m (0.34%), and OKVIS achieved 1.699 m (0.75%) over the 230 meter trajectory. These results indicate that the motion assumptions of Model 1 best fit the true trajectory in this scenario. Note that this scenario was more challenging than the first experiment, primarily due to the fact that during this test, there were people walking around, lighting conditions were varying, and some parts of



(a) Example images (top) and 3D trajectory (bottom).



(b) Projection of the estimated trajectory onto the floor plan.

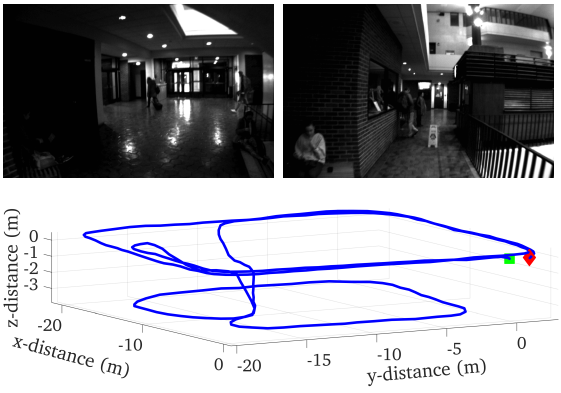
Figure 3.7: The trajectory estimates of the indoor experiment performed in the UD Gore Hall. Two example images from the dataset can be seen in (a), while the starting and ending locations are shown by a green square and red diamond in the plot, respectively. Note that the three floors have similar layouts, and thus only one floor plan is shown in plot (b).

the environment lacked good features to detect and track (see Figure 3.8a).

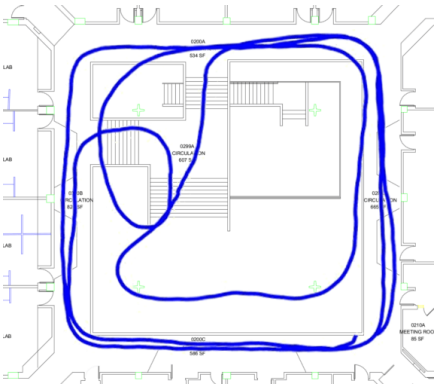
The 3D trajectory estimate and its projection onto the floor plan are shown in Figures 3.8a and 3.8b, respectively. These results clearly demonstrate that our VIO systems using the proposed closed-form preintegration are able to perform accurate 3D motion tracking in relatively large-scale complex environments.

3.5.2 Loosely-Coupled Direct VINS

When a stereo pair arrives, as in the preceding indirect VIO, we perform the proposed closed-form preintegration from the previous IMU state to the current state. We then check the list of stored keyframes for a suitable candidate for direct image alignment, based on a field-of-view constraint between the candidate and the current image. If no such acceptable candidate is found, a new keyframe is created from the previous image pair and its depth map is computed using the OpenCV function `StereoSGBM`. In particular, in order to perform course-to-fine alignment, the depth



(a) Example images (top) and 3D trajectory (bottom).



(b) Projection of the estimated trajectory onto the floor plan.

Figure 3.8: The results of the indoor experiment performed in the UD Smith Hall. Two example left camera images are shown in (a). In plot (b), the starting and ending locations of the trajectory estimates are shown as a green square and red diamond, respectively. Note that the trajectory shown in (c) occurs on both the second and first floors.

map is computed for multiple image pyramid levels. Starting at the coarsest image level, we perform iterative image alignment, using the larger levels to further refine the course image alignment transform. This image alignment optimization was implemented in a CUDA kernel for GPU acceleration, thus allowing for the system to achieve real-time performance. After convergence, we recover the relative-pose constraint and add it as a factor to our direct-VINS graph. Note that as compared to our indirect VIO method, we do not perform marginalization and thus allow later incorporation of loop closures. To handle this increase of computational complexity and allow for real-time performance, we leverage the iSAM2 incremental smoothing implementation within the GTSAM framework [24, 69]. However, the proposed framework can be further refined, for example, by more intelligently selecting keyframes.

3.5.2.1 EurocMav dataset:

To validate our direct VINS approach, we perform tests on the same EurocMav sequences as before, which allow for direct comparison to a ground-truth trajectory [14]. The results of the proposed direct VINS using two different preintegration models are shown in Table 3.4. Clearly, in scenarios in which a large amount of loop closures

Table 3.4: Average absolute RMSE results for the EurocMav sequences over ten runs using the proposed direct VINS algorithm. All systems were initialized with the ground truth state. The smallest position and orientation errors have been highlighted.

	MODEL-1		**MODEL-2**		**DISCRETE**	
Units	m	deg	m	deg	m	deg
V1_01_easy	**0.2445**	**2.218**	0.2482	2.246	0.2530	2.223
V1_02_med	0.1598	1.767	**0.1309**	**1.483**	0.1763	1.899
V1_03_diff	**0.0990**	**1.180**	0.1030	1.279	0.1030	1.234
V2_01_easy	**0.1627**	2.089	0.1940	1.956	0.1664	**1.533**
V2_02_med	0.1809	2.530	**0.1665**	2.527	0.1688	**2.309**
V2_03_diff	0.9337	6.187	**0.8927**	5.425	1.0137	**4.998**
MH_01_easy	**0.2947**	2.270	0.3277	**2.148**	0.3217	2.226
MH_02_easy	**0.1882**	1.650	0.2136	1.582	0.2008	**1.483**
MH_03_med	0.2330	2.096	0.2295	2.121	**0.2288**	**2.092**
MH_04_diff	0.4792	**2.513**	0.4867	2.627	**0.4724**	2.562
MH_05_diff	**0.2884**	**1.664**	0.3014	1.722	0.2946	1.700

are present (e.g., "V1_03_diff"), this system can outperform the tightly-coupled VIO system (see Section 3.3.1 and Table 3.2). However, when such loop closures are not available, the loosely-coupled systems suffer from larger drifts, as can be seen from the result of "V2_01_easy".

In these experiments, Model 1 tended to offer improved performance as compared to discrete preintegration (although not in all cases), while both proposed methods provided similar levels of performance to each other, with each having trajectories where they outperform the other. In addition, the proposed direct VINS is sensitive to the tuning parameters, which in this experiment were chosen as *identical* across all sequences, rather than finding an optimal set per scenario. This led to situations such as "V2_03_diff", in which some of the runs yield incorrect loop closures despite our system attempting to reject these, greatly corrupting the resulting trajectory estimates. However, as the purpose of this work is to show the accuracy of the proposed preintegration, instead of the robustness of the utilized front-ends, these results along with the previous simulation results strongly suggest that our preintegration models

can be, and should be, used when designing graph-based VINS.

3.6 Conclusion

In this chapter, we have analytically derived closed-form inertial preintegration and successfully applied it to graph-based visual-inertial navigation systems (VINS). In particular, we have advocated for two new preintegration models for the evolution of IMU measurements across sampling intervals. In the first, we assume that the inertial measurements remain piecewise constant; while in the second, we incorporate a piecewise constant *local* acceleration model into the preintegration framework. We have validated through extensive Monte-Carlo simulations that both models outperform the state-of-the-art *discrete* preintegration. Furthermore, we have utilized this closed-form preintegration theory and developed two different VINS algorithms primarily to show the advantages of the proposed preintegration. In the first, we formulated an indirect (feature-based), tightly-coupled, sliding-window optimization based VIO system that offers competitive (if not better) performance to a state-of-the-art graph-based VINS algorithm. The second VINS method was developed instead based on loosely-coupled direct image alignment with the proposed preintegrations, allowing for efficient incorporation of informative loop closures.

Chapter 4

MULTI-IMU MULTI-CAMERA VINS

Within the VINS literature, most efforts have focused on the minimal sensing case of a *single* camera and IMU [58, 57, 77, 94, 103]. While 3D motion tracking with minimal sensing capability is of interest, in practice, it is highly desirable to optimally and efficiently fuse *all* information from *multiple* sensors to improve estimation robustness and accuracy [102]. In particular, we note that most single-view systems are greatly susceptible to loss of texture in a given viewing direction [38], and thus single-camera methods may suffer greatly from measurement depletion. However, even with multiple cameras, conditions such as poor lighting may cause the estimator to rely solely on its IMU due to the lack of visual information. As such, building VINS which leverage multiple sensors (both cameras and IMUs) remains of great interest. In addition, systems which do not utilize redundant sensors may become useless once required components fail.

In practice, fusing information from multiple sensors that each collect local information requires knowing the spatial relationship (relative pose) between each sensor. In addition, if the sensors are not hardware synchronized (i.e., they are not electronically triggered to collect readings at the same time) then representing the state at each sensing time may become computationally infeasible, and, due to latency issues, the clocks may differ between different sensors, leading to nontrivial time offsets between measurement timestamps [81]. These can be combated through manufacturing all components as a single tightly-coupled sensing unit, but this may become prohibitively expensive for widespread applications. By contrast, in this chapter, we seek to enable *plug-and-play* functionality for VINS, wherein different sensors can be freely

added/removed without requiring hardware synchronization or exhaustive offline calibration.

We perform this task in two main sections, by first deriving a multi-camera system which can efficiently fuse information from multiple asynchronous cameras through efficient pose interpolation while calibrating all sensors [34]. We then introduce the multi-IMU VINS system which estimates the navigation states of *multiple* IMUs and performs simultaneous information fusion and calibration [32]. This is achieved through a novel spatial-temporal constraint on the relative pose between IMU that acts as an updating measurement to the combined filter. We finally present a combined estimator that can utilize an arbitrary number of sensors, while performing all calibrations *online*, and thoroughly evaluate its performance in terms of estimation accuracy and consistency. While in the previous chapter we focused on graph-based VINS, in many applications we cannot afford the increased computation of these systems, and must instead rely on less expensive filtering approaches [125]. As such, for these results, we focus on robustifying the lightweight MSCKF through the incorporation of an arbitary number of sensors.

4.1 Related Works

4.1.1 Multi-Camera Systems

While monocular-VINS has been widely studied (e.g., see [58, 57, 60, 63, 77, 93, 103] and references therein), one straightforward extended configuration over the monocular setting is to use a *stereo* camera, wherein the *two* cameras are mounted such that they observe the same spatial volume from offset camera centers at the same image time. Stereo vision enables 3D triangulation of features seen in the overlapping view without requiring motion of the sensor platform, thus allowing for the direct recovery of scale if the spatial transforms between cameras are known. Motivated by this increase in robustness, Sun et al. [117] developed the MSCKF-based stereo-VINS with the particular application to high-speed aerial vehicles. Paul et al. [102] extended the inverse square-root version of the MSCKF (namely SR-ISWF) [125] to provide

real-time VINS on mobile devices while allowing for a configuration of both stereo and binocular (non-overlapping) cameras, and showed that the inclusion of more visual information improves the estimation accuracy.

While stereo cameras provide robustness due to their ability to perform feature triangulation and scale recovery even without the IMU, they remain vulnerable to dynamic environmental motion and textureless regions in its given viewing direction. More importantly, the requirement of an overlapping field of view and synchronous camera triggering may not easily extend to an *arbitrary* number of *plug-and-play* cameras – which is a highly-desirable characteristic and could greatly promote the widespread deployment of VINS in practice. Additionally, due to the enforcement of cross-image matching (for example matching features from the left to right stereo image) the process of visual tracking is coupled and cannot be directly parallelized. For these reasons, we propose a general multi-camera VINS in this work that can tightly fuse the visual information from an *arbitrary* number of *non-overlapping*, *asynchronous* and heterogeneous cameras and the IMU measurements, so that the proposed approach is robust to environmental conditions and single-camera failure. We stress that in the proposed system, we do *not* perform any cross-image matching, since we have non-overlapping images and instead allow each camera feed to be processed independently and in parallel.

Houben et al. [59] extended the ORB-SLAM [95] to a system of multiple cameras with varying viewing directions and an IMU for UAVs within a graph-SLAM framework but assumed known sensor calibration and simultaneous triggering of all involved cameras. Recently, Paul et al. [101] addressed the problem of increased computational burden in stereo-VINS and proposed an alternating stereo-VINS algorithm. In their system, the *two* cameras in a stereo pair were triggered in an alternating manner, preventing the need to process both images at the same time while still taking advantage of the offset camera centers provided by a stereo configuration. In addition, they further reduced computation by explicitly estimating the historical IMU poses corresponding

to only *one* of the camera's imaging times, while using pose interpolation to represent the state at intermediate times corresponding to the other camera. While in this work we use a similar interpolation scheme to reduce computation, we simultaneously perform time offset and spatial calibration between $n \geq 2$ cameras.

An integral part of any multi-sensor fusion system is the spatial (relative transformation), temporal (time offset), and intrinsic (e.g., focal length, camera center, and distortion parameters) calibration parameters for each sensor, as errors in the values of these parameters can greatly degrade localization performance – if not catastrophically. Calibration can be broadly divided into two main categories. Offline methods perform a computationally expensive solution process in exchange for providing highly accurate calibration estimates. In particular, Furgale et al. [44] developed a multi-sensor calibration system that performed spatial, temporal, and intrinsic calibration of an arbitrary number of cameras along with an IMU. However, performing offline calibration could be a tedious process that limits deployment time and requires the calibration to be repeated if the sensor configuration changes. In addition, treating the calibration parameters provided by these methods as "known" (zero uncertainty) may lead to unmodeled errors, thereby introducing estimation inconsistency [82]. By contrast, online methods treat the calibration parameters as random variables, simultaneously estimating them along with the navigation states. While many VINS algorithms perform online calibration of the spatial extrinsic transform between the camera and IMU, relatively few also estimate the time offset between them [75, 117]. Systems that *do* perform online temporal calibration [78, 80, 103], however, are typically limited to a *single* IMU-camera pair. One of the most notably complete systems in this category is by Li et al. [82], who performed online calibration of both the extrinsic parameters between the IMU and camera as well as the *intrinsics* of both sensors.

4.1.2 Multi-IMU Systems

To date, almost all VINS algorithms utilize a single IMU, and thus these VINS algorithms remain vulnerable to single IMU failure, and are unable to leverage the

additional information these redundant sensors provide. In reality, sensors certainly may experience failures preventing the estimator from acquiring new measurements from the faulty sensor. If this sensor (such as IMUs in VINS) is required to fully constrain the estimation problem, its failure will result in the collapse of the entire system. Such failures can occur in practice due to sensor disconnection (due to impact), high temperatures, or sensitivity to vibrations [4]. To compensate for this issue, redundant sensors (i.e., hardware redundancy) are typically used [68]. Therefore, adding more IMUs into VINS appears to be a straightforward solution for improving the system resilience against sensor failures, in particular, given the low cost of IMUs. However, to the best of our knowledge, *few* VINS utilize *multiple* IMUs while performing real-time estimation. While outside of VINS, fusing multiple IMUs has been widely studied [8, 66, 105], e.g., with the application to human motion tracking [41], these methods neither perform visual-inertial fusion nor online spatial/temporal calibration as in this work. Ma et al. [89] fused a tactical grade IMU, stereo camera, leg odometry, and GPS measurements in an EKF alongside a navigation-grade *gyroscope* for estimating the motion of a quadruped robot, but without calibration or the ability to use acceleration measurements from a second IMU.

For offline calibration, Rehder et al. [107] used a continuous-time basis function representation [44] of the sensor trajectory to calibrate both the extrinsics and intrinsics of a multi-sensor system in a batch fashion. As this B-spline representation allows for the direct computation of expected local angular velocity and local linear acceleration, the difference between the expected and measured inertial readings served as errors in the batch optimization formulation. Kim et al. [70] reformulated IMU preintegration [31, 43, 88] by transforming the inertial readings from a first IMU frame into a second frame. This allowed for spatial calibration with online initialization between an IMU and other sensors (including other IMUs), but did not include temporal calibration while also relying on computing angular accelerations from gyroscope measurements without optimal characterization of their uncertainty.

Recently, Zhang et al. [130] proposed a method for fusing multiple IMU's by setting up a "virtual" IMU and estimating its acceleration and angular velocity through least-squares using all the inertial measurements collected by every IMU. These synthetic readings, which have substantially smaller noises than those of each individual IMU, could then be used in a visual-inertial navigation system directly. While this was shown to offer competitive results along with large computational savings compared to the method proposed in this thesis, it requires *perfectly* calibrated sensors, which may be difficult to achieve in practice. In addition, we note that because our system simply applies relative pose constraints to fuse the information, these updates can be used even if the calibration is *time-varying*.

4.2 Multi-Camera VINS

In this section, we design a general multi-camera VINS (mc-VINS) algorithm that is capable of tightly fusing the visual information from an *arbitrary* number of *non-overlapping*, *asynchronous* cameras and IMU measurements within the MSCKF framework, while limiting the increased computational burden on the estimator. In comparison to the state-of-the-art VINS [77, 57, 103, 75], we not only online calibrate *all* spatial and temporal calibration parameters between the used sensors but also jointly estimate the *intrinsics* for each camera, allowing for online refinement of these parameters. In particular, the main contributions of this section are [34]:

- By leveraging the computationally-efficient MSCKF framework, we develop the tightly-coupled multi-camera VINS (mc-VINS) with online intrinsic and extrinsic calibration. The proposed mc-VINS is able to utilize all information from any number of cameras without constraints on sensor configurations since both spatial/temporal calibration parameters and intrinsics of each camera are simultaneously estimated online.
- Instead of maintaining stochastic clones for each camera, we only perform cloning of the IMU poses at imaging times of a freely chosen "base" camera, and use interpolation on the $SO(3) \times \mathbb{R}^3$ manifold to represent the pose at an arbitrary intermediate time for all other cameras. By representing this interpolation as a function of the unknown time offset between cameras, we can perform both spatial and temporal calibration between all sensors (including the IMU), and

additionally refine the *intrinsic* parameters for each camera to allow for high-fidelity estimation.

- We validate the proposed mc-VINS on real multi-camera visual-inertial sensor platforms using different camera configurations and in various environments.

4.2.1 Multi-Camera Estimation

In this section, within the standard MSCKF framework [94], we present in detail the proposed multi-camera (mc)-VINS with online instrinsic and extrinsic sensor calibration. As before we maintain the inertial state:

$$\mathbf{x}_I = \begin{bmatrix} {}^I_G\bar{q}^\top & {}^G\mathbf{p}_I^\top & {}^G\mathbf{v}_I^\top & \mathbf{b}_\omega^\top & \mathbf{b}_a^\top \end{bmatrix}^\top \tag{4.1}$$

We also estimate the typical MSCKF pose clones corresponding to N old imaging times:

$$\mathbf{x}_{cl} = \begin{bmatrix} {}^{I_{k-1}}_G\bar{q}^\top & {}^G\mathbf{p}_{I_{k-1}}^\top & \dots & {}^{I_{k-N+1}}_G\bar{q}^\top & {}^G\mathbf{p}_{I_{k-N+1}}^\top \end{bmatrix}^\top \tag{4.2}$$

Compared to the standard MSCKF, with the multi-camera setting under consideration, we additionally estimate *extrinsics* including both the spatial (relative pose) and temporal (time offset) calibration parameters, as well as the *intrinsics* of each camera. Specifically, each camera state contains the following parameters:

$$\mathbf{x}_{ci} = \begin{bmatrix} {}^{Ci}_I\bar{q}^\top & {}^{C_i}\mathbf{p}_I^\top & {}^{C_i}t_{C_b} & \boldsymbol{\zeta}_i^\top \end{bmatrix}^\top \tag{4.3}$$

$$\boldsymbol{\zeta}_i = \begin{bmatrix} f_{xi} & f_{yi} & p_{xi} & p_{yi} & \mathbf{d}_i^\top \end{bmatrix}^\top \tag{4.4}$$

where ${}^{C_i}t_{C_b}$ is the time offset between camera i and the base camera, $f_{xi},\ f_{yi}$ represent the focal lengths, $p_{xi},\ p_{yi}$ denote the location of the principal point, and $\mathbf{d}_i$ refers to the vector of distortion parameters whose length/definition depends on the camera model being used (see [54]). For the base camera, however, we store its temporal misalignment with respect to the IMU, ${}^{C_b}t_I$ as the relative time offset to itself is zero.

4.2.2 Multi-Camera Propagation

It is important to note that due to hardware latency of on-board processing, the time reported by the base camera will *differ* from the same time expressed in the

IMU's clock. We consider a time ${}^{C_b}t$ as expressed in the base camera's clock, which is related to the same instant represented in the IMU clock, ${}^{I}t$, by a time offset ${}^{C_b}t_I$, i.e.,

$$ {}^{I}t = {}^{C_b}t + {}^{C_b}t_I \tag{4.5} $$

As this time offset is unknown, we include it as a parameter in our state vector to be estimated. With the estimate of this time offset ${}^{C_b}\hat{t}_I$, whenever we receive the $(k+1)$-th image with reported time ${}^{C_b}t_{k+1}$, we perform propagation of our IMU up to the *estimated* time of the image as expressed in the IMU clock [see (4.5)]: ${}^{I}\hat{t}_{k+1} = {}^{C_b}t_{k+1} + {}^{C_b}\hat{t}_I$. Specifically, we propagate the IMU from its current time ${}^{I}\hat{t}_k$ (actually the estimate of the IMU time for k-th image at the time of the previous propagation, *before* update) up to this new time by processing all IMU measurements $\mathcal{I}$ collected over the time interval $[{}^{I}\hat{t}_k, {}^{I}\hat{t}_{k+1}]$, based on the conventional IMU dynamics (2.35):

$$ \mathbf{x}_{I({}^{I}\hat{t}_{k+1})} = \mathbf{g}\Big(\mathbf{x}_{I({}^{I}\hat{t}_k)}, \mathcal{I}, \mathbf{n}_I\Big) \tag{4.6} $$

4.2.2.1 State Augmentation

After propagating to time step $k+1$, we only have a state estimate of the IMU at the *estimated* time ${}^{I}\hat{t}_{k+1}$, while we actually need to express all camera measurements as a function of the IMU pose at the *true* time ${}^{I}t_{k+1}$. To accomplish this, we utilize stochastic cloning [110] as described previously. Using this methodology, the following linearized stochastic cloning is performed to create an estimate of the IMU at this true time in a manner analogous to [78]:

$$ {}^{G}\mathbf{p}_{I({}^{I}t_{k+1})} \approx {}^{G}\mathbf{p}_{I({}^{I}\hat{t}_{k+1})} + {}^{G}\mathbf{v}_{I({}^{I}\hat{t}_{k+1})}\,{}^{C_b}\tilde{t}_I \tag{4.7} $$

$$ {}^{I({}^{I}t_{k+1})}_{G}\mathbf{R} \approx \mathrm{Exp}\left(-{}^{I}\boldsymbol{\omega}_I\left({}^{I}\hat{t}_{k+1}\right){}^{C_b}\tilde{t}_I\right){}^{I({}^{I}\hat{t}_{k+1})}_{G}\mathbf{R} \tag{4.8} $$

Where $\mathrm{Exp}(\cdot)$ is the $SO(3)$ matrix exponential which maps a vector in $\mathbb{R}^3$ to a rotation matrix [20], and $\boldsymbol{\omega}\left({}^{I}\hat{t}_{k+1}\right)$ is the true angular velocity at time ${}^{I}\hat{t}_{k+1}$. That is, because we can write the pose at the true time as a function of the pose at the estimated time, as well as the time offset error (both of which are contained in our state), we can

augment our state to contain this new pose. The Jacobians for this cloning are given by:

$$\frac{\partial^G \tilde{\mathbf{p}}_{I({}^I t_{k+1})}}{\partial^G \tilde{\mathbf{p}}_{I({}^I \hat{t}_{k+1})}} = \mathbf{I}_3, \ \frac{\partial^G \tilde{\mathbf{p}}_{I({}^I t_{k+1})}}{\partial^{C_b} \tilde{t}_I} = {}^G \hat{\mathbf{v}}_{I({}^I \hat{t}_{k+1})}$$
$$\frac{\partial_G^{I({}^I t_{k+1})} \tilde{\boldsymbol{\theta}}}{\partial_G^{I({}^I \hat{t}_{k+1})} \tilde{\boldsymbol{\theta}}} = \mathbf{I}_3, \ \frac{\partial_G^{I({}^I t_{k+1})} \tilde{\boldsymbol{\theta}}}{\partial^{C_b} \tilde{t}_I} = \boldsymbol{\omega}_m({}^I \hat{t}_{k+1}) - \hat{\mathbf{b}}_w({}^I \hat{t}_{k+1})$$

Note that errors in the linear and angular velocity estimates do not affect the cloning up to first order as these multiply the time offset error. With these Jacobians, we obtain the covariance of the augmented state.

4.2.3 Multi-Camera Measurements Update

Consider a 3D feature, ${}^G \mathbf{p}_f$, which is captured by the i-th camera at imaging time ${}^{C_b} t_k$ with respect to the *base* camera. In this work we track features extracted uniformly using FAST [109] and tracked independently for each camera's image stream using KLT [6], with outliers rejected via 8-point RANSAC. The measurement function for a measurement captured at time ${}^{C_b} t := t$ is given by:

$$\mathbf{z}_k = \mathbf{w}_i \left(\mathbf{\Pi} \left({}^{C_i(t)} \mathbf{p}_f \right), \ \boldsymbol{\zeta}_i \right) + \mathbf{n}_k, \tag{4.9}$$

where $\mathbf{w}_i(\cdot)$ is the function mapping the normalized image coordinates onto the image plane based on the camera intrinsics $\boldsymbol{\zeta}_i$ and the camera model used (e.g., radial-tangential or fisheye [100]), and ${}^{C_i(t)} \mathbf{p}_f = [x \ y \ z]^\top$ is the position of the feature expressed in camera i's frame at true time t:

$${}^{C_i(t)} \mathbf{p}_f = {}^{C_i}_{I} \mathbf{R} \, {}^{I(t)}_{G} \mathbf{R} \left({}^G \mathbf{p}_f - {}^G \mathbf{p}_{I(t)} \right) + {}^{C_i} \mathbf{p}_I \tag{4.10}$$

Letting $\mathbf{a} = \mathbf{\Pi} \left({}^{C_i(t)} \mathbf{p}_f \right)$ denote the normalized image coordinates, we note that when computing Jacobians for this function, dependence on the intrinsics comes from the image plane mapping function $\mathbf{w}_i$:

$$\frac{\partial \mathbf{z}_k}{\partial \tilde{\boldsymbol{\zeta}}_i} = \frac{\partial \mathbf{w}_i \left(\mathbf{a}, \ \boldsymbol{\zeta}_i \right)}{\partial \tilde{\boldsymbol{\zeta}}_i} \tag{4.11}$$

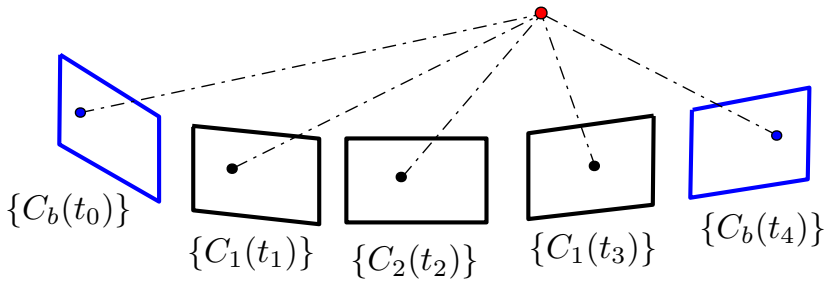


Figure 4.1: Illustration of how asynchronous multi-camera measurements are collected. We have cloned at the base camera imaging times: $\{C_b(t_0)\}$, $\{C_b(t_4)\}$ (blue). A series of measurements between these times from other non-base cameras C_1 and C_2 are received, requiring us to interpolate these poses in terms of the base camera so that they can be utilized in the MSCKF update.

while for other variables $\mathbf{v}$, the dependency is from ${}^{C_i(t)}\mathbf{p}_f$:

$$\frac{\partial \mathbf{z}_k}{\partial \tilde{\mathbf{v}}} = \frac{\partial \mathbf{w}_i\left(\mathbf{a},\ \boldsymbol{\zeta}_i\right)}{\partial \tilde{\mathbf{a}}} \frac{\partial \tilde{\mathbf{a}}}{\partial {}^{C_i(t)}\tilde{\mathbf{p}}_f} \frac{\partial {}^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial \tilde{\mathbf{v}}} \tag{4.12}$$

In summary, rather than using the undistorted, normalized pixel coordinates as measurements (as is typically done), we utilize the *raw* image coordinates, allowing us to calibrate the intrinsics of each camera along with the rest of the state.

4.2.3.1 Asynchronous Multi-Camera Measurements

Note that the cloned state $\mathbf{x}_{cl}$ (4.2) only contains the IMU poses at the true imaging times of the *base* camera, while the camera measurements (4.10) require that for all non-base cameras we can express the pose at *their* image times, which may not align with those of the base camera (see Figure 4.1). Clearly, without solving this issue, the inclusion of all other measurements that are not collected synchronously with the base camera cannot be written as functions of the state and used in the update. Naively, one could simply perform stochastic cloning at *every* imaging time, however this would lead to a greatly increased computational burden. For instance, utilizing m distinct asynchronous cameras each operating at the same frequency would require multiplying the state size by m due to the large number of required clones. Therefore, we instead employ a $SO(3) \times \mathbb{R}^3$ linear interpolation method between these poses to allow for the incorporation of measurements at arbitrary times. We note that while higher-order interpolation schemes may be utilized for improved accuracy,

in what follows we leverage a linear scheme for its computational efficiency, and later investigate the effect of this order choice in Section 4.4.1.

In particular, assuming the measured time reported by the i-th camera as ${}^{C_i}t$, to express this time in the base camera's clock, we must compensate for the time offset, i.e., ${}^{C_b}t = {}^{C_i}t + {}^{C_i}t_{C_b} := t$. Let ${}^{C_b}t_1 := t_1$ and ${}^{C_b}t_2 := t_2$ denote the times for the bounding base camera/IMU clones between which ${}^{C_i}t + {}^{C_i}\hat{t}_{C_b} := \hat{t}$ falls. We linearly interpolate the cloned poses at t_1 and t_2 to find the pose at the measurement time:

$$ {}^{I(t)}_{G}\mathbf{R} = \text{Exp}\left(\lambda \text{Log}\left({}^{I(t_2)}_{G}\mathbf{R}\,{}^{G}_{I(t_1)}\mathbf{R}\right)\right){}^{I(t_1)}_{G}\mathbf{R} \tag{4.13} $$

$$ {}^{G}\mathbf{p}_{I(t)} = (1-\lambda){}^{G}\mathbf{p}_{I(t_1)} + \lambda {}^{G}\mathbf{p}_{I(t_2)} \tag{4.14} $$

$$ \lambda = \frac{t - t_1}{t_2 - t_1} \tag{4.15} $$

where $\text{Log}(\cdot)$ is the inverse operation of $\text{Exp}(\cdot)$ which maps a rotation matrix to a vector in $\mathbb{R}^3$. These equations essentially interpolate both the orientation and position under the approximation of constant linear and angular velocity over the interval. As images of the base camera typically arrive at high rate (around 20 Hz in most applications) as compared to the physical camera motion, we argue that this serves as a good approximation. Due to the fact that marginalization is only ever performed on the oldest clone, any required camera pose will typically be within 25 milliseconds of a clone in the sliding window. Note that a similar interpolation scheme was used by Paul and Roumeliotis [101] to reduce the complexity of processing a synchronized stereo pair, but without either the intrinsic and extrinsic (spatial and temporal) calibration of multiple non-base cameras as focused on in this work.

Let $\mathbf{x}(t_1)$, $\mathbf{x}(t_2)$ and $\mathbf{x}(t)$ denote the IMU clones at the neighboring times, and the interpolated value, respectively (corresponding to times t_1, t_2, and t. By substituting the above expressions into the measurement function, we have the following Jacobians with respect to the bounding IMU clones and the time offset ${}^{C_i}t_{C_b}$:

$$ \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial \tilde{\mathbf{x}}(t_1)} = \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial \tilde{\mathbf{x}}(t)}\frac{\partial \tilde{\mathbf{x}}(t)}{\partial \tilde{\mathbf{x}}(t_1)} \tag{4.16} $$

$$ \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial \tilde{\mathbf{x}}(t_2)} = \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial \tilde{\mathbf{x}}(t)}\frac{\partial \tilde{\mathbf{x}}(t)}{\partial \tilde{\mathbf{x}}(t_2)} \tag{4.17} $$

$$\frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^{C_i}\tilde{t}_{C_b}} = \frac{^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial\tilde{\mathbf{x}}(t)}\frac{\partial\tilde{\mathbf{x}}(t)}{\partial^{C_i}\tilde{t}_{C_b}} \tag{4.18}$$

To compute these Jacobians, we use the following approximations for a small angle $\boldsymbol{\psi}$ [43]:

$$\begin{aligned}\text{Exp}\left(\boldsymbol{\theta}+\boldsymbol{\psi}\right) &\approx \text{Exp}\left(\mathbf{J}_l\left(\boldsymbol{\theta}\right)\boldsymbol{\psi}\right)\text{Exp}\left(\boldsymbol{\theta}\right) &(4.19)\\ &\approx \text{Exp}\left(\boldsymbol{\theta}\right)\text{Exp}\left(\mathbf{J}_r\left(\boldsymbol{\theta}\right)\boldsymbol{\psi}\right) &(4.20)\end{aligned}$$

where $\mathbf{J}_l(\cdot)$ and $\mathbf{J}_r(\cdot)$ are the left and right Jacobians of $SO(3)$ [20], respectively. By defining for convenience ${}^2_1\hat{\boldsymbol{\phi}} = \text{Log}\left({}^{I(t_2)}_{I(t_1)}\hat{\mathbf{R}}\right)$, we have the following Jacobians with detailed derivations found in Appendix F.1:

$$\begin{aligned}
\frac{\partial^{I(t)}_G\tilde{\boldsymbol{\theta}}}{\partial^{I(t_1)}_G\tilde{\boldsymbol{\theta}}} &= -\left(\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right)\mathbf{J}_r^{-1}\left({}^2_1\hat{\boldsymbol{\phi}}\right) - \text{Exp}\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right)\right)\\
\frac{\partial^{I(t)}_G\tilde{\boldsymbol{\theta}}}{\partial^{I(t_1)}_G\tilde{\boldsymbol{\theta}}} &= \hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right)\mathbf{J}_l^{-1}\left({}^2_1\hat{\boldsymbol{\phi}}\right)\\
\frac{\partial^{I(t)}_G\tilde{\boldsymbol{\theta}}}{\partial^{C_i}\tilde{t}_{C_b}} &= -\frac{1}{t_2-t_1}{}^2_1\hat{\boldsymbol{\phi}}\\
\frac{\partial^G\tilde{\mathbf{p}}_{I(t)}}{\partial^G\tilde{\mathbf{p}}_{I(t_1)}} &= (1-\hat{\lambda})\mathbf{I},\quad \frac{\partial^G\tilde{\mathbf{p}}_{I(t)}}{\partial^G\tilde{\mathbf{p}}_{I(t_1)}} = \hat{\lambda}\mathbf{I}\\
\frac{\partial^G\tilde{\mathbf{p}}_{I(t)}}{\partial^{C_i}\tilde{t}_{C_b}} &= \frac{1}{t_2-t_1}\left({}^G\hat{\mathbf{p}}_{I(t_2)} - {}^G\hat{\mathbf{p}}_{I(t_1)}\right)
\end{aligned} \tag{4.21}$$

As a result, by incorporating this representation, we can estimate both the spatial and temporal calibration parameters for all cameras, while *only* storing those poses related to the base camera's imaging times.

4.2.3.2 Note on Complexity

A key advantage of the proposed approach is the ability to parallelize the visual tracking front-end. Since all cameras can be processed independently, we argue that one can perform "camera-edge" visual tracking allowing for the horizontal scaling of the visual front-ends. As seen in Figure 4.2, the visual front-ends can be treated as independent and images from all cameras can be processed in parallel. For example,

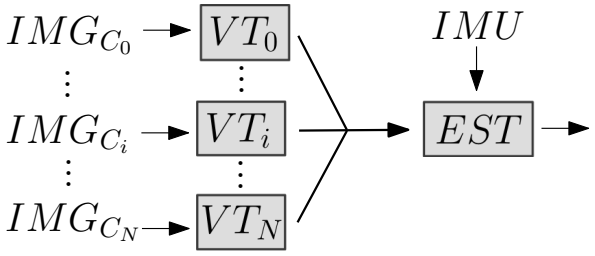


Figure 4.2: Illustration showing how the proposed system horizontally scales as more images are added. Simply scaling of the visual tracker (VT) allows for the parallelization of feature tracking, which feeds these tracks to the estimator (EST) for processing.

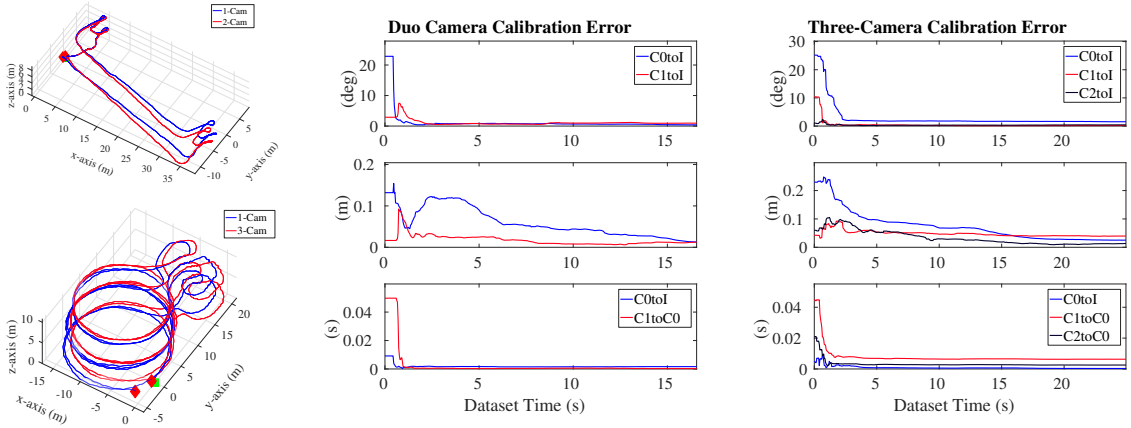


Figure 4.3: The trajectories of the proposed mc-VINS and minimal sensing cases are shown for both the 140 meter long duo-camera (top-left) and 440 meter long three-camera (bottom-left) datasets. The calibration error for the duo-camera configuration (center) and the three-camera configuration (right) show only the first few seconds of the total dataset and also plot the camera to IMU time offset in the bottom most time offset error plots.

in practice we could let each camera have a local micro-computer or hardware embedded processor ("camera-edge" processing) that performs feature tracking that upon completion can be sent to the centralized estimator for asynchronous fusion. The only introduced increase in computation is the possibly larger magnitude of features used during update, which can be managed by careful selection of a subset of features to be processed.

Figure 4.4: Overview picture of the MAV (top). The camera configuration with the frontwards camera (CAM0) and downwards facing camera (CAM1) are shown in the bottom.

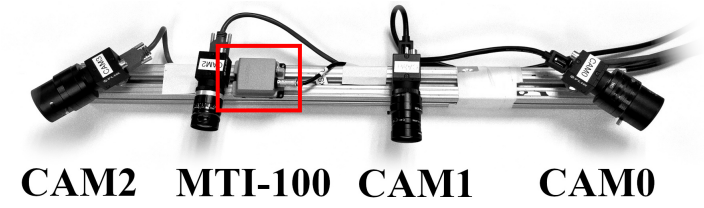


Figure 4.5: Overview picture of the three-camera configuration. Note that in this experiment only one of the forward facing cameras was used (CAM1), along the the two side-facing fisheye cameras (CAM0/2).

4.2.4 Multi-Cam Experiential Results

4.2.4.1 Duo-Camera Configuration

For our first experiment, a custom 3D printed mount was used to attach two Omnivision OV7251 fisheye global shutter cameras to the Snapdragon Flight.[1] The cameras were mounted in a binocular configuration such that one faced forward and the other faced downward as shown in Figure 4.4. The overall size of the quadrotor was 10in $\times$ 11in, with 5in rotors. The Snapdragon Flight was mounted underneath. The cameras were triggered at 30Hz and the onboard InvenSense MPU-9250 IMU ran at 500Hz. The hand-held dataset was recorded at the University of Delaware's Spencer Lab, in which the quadrotor traveled from the first floor, traversed the staircase to the third floor, and returned to the starting location to allow for computation of the start-end error. The approximately 143 meter trajectory traveled can be seen in Figure 4.3 (top-left). The extrinsic calibration values provided by the offline calibration toolbox Kalibr [44], which are expected to be close to the actual values, were taken as a proxy

[1] https://developer.qualcomm.com/hardware/qualcomm-flight

for the ground-truth. To illustrate calibration, we manually perturbed these ground-truth values, which were then used as the initial estimates in the filter. The intrinsics values provided from Kalibr were not perturbed in this experiment, but were refined online. We bounded the maximum number of features extracted from each image at 250, and utilized an MSCKF window size of 15. To account for the randomness in RANSAC-based vision, the averaged results from ten runs are reported.

As shown in Figure 4.3 (center), the calibration parameters quickly converge from poor initial guesses to the results reported by Kalibr. This validates the proposed method of online calibration. As no ground-truth position estimates were available, and as our trajectory returned to the starting location, the difference in the reported start and end locations serves as our error metric. The start-end error for this trajectory was approximately 0.45 meters, amounting to 0.33% of the total path. This shows that even with incorrect initial calibration, our system can recover and, by utilizing the information from multiple image streams, can provide accurate trajectory estimates. By contrast, the same system using only the base camera provided an error of 0.65 m (0.48%), of the path, thereby validating the improvement of the multi-camera system. To highlight the effect of intrinsics refinement, we also ran the duo-camera estimator *without* online intrinsic refinement, and found the error to be 1.37 m (1.00%), a clear decrease in accuracy. Lastly, during processing, the system remained real-time, as the time to perform single image feature tracking (which can be run in parallel for each camera) plus MSCKF update utilizing the information from all cameras took on average 0.023 seconds (44 Hz) compared to the 30 Hz rate of each camera.

4.2.4.2 Three-Camera Configuration

For our second experiment, a custom rig with multiple Pointgrey Blackfly cameras and an MTI-100 IMU was designed and is shown in Figure 4.5. One Blackfly faced forward, while two others which were given fisheye lenses and tilted in opposite directions. Each camera operated asynchronously at 20 Hz. An approximately 440 meter

closed-loop trajectory was performed across multiple floors in Gore Hall at the University of Delaware under poor lighting conditions, with the resulting trajectory shown in Figure 4.3 (bottom-left). As in the previous experiment, we report the averaged results from ten runs. As we have increased the number of cameras as compared to the previous configuration, we utilized a smaller MSCKF window size of 12 to ensure real-time performance (in this experiment, image tracking plus MSCKF update took on average 0.03 seconds). As shown in Figure 4.3 (right), our system was able to perform accurate calibration of the involved systems, despite having extremely poor initial guesses (over 20 degree and 0.2 meter error for the base camera to IMU transformation), our system was able to converge to values close to those reported by Kalibr, while the final position error was approximately 0.61 meters (0.14%). By contrast, the same dataset processed with the base camera alone yielded a final ending error of 3.58 meters (0.80%). This large increase in performance clearly validates our desire to include more cameras into the estimation process. For this experiment, we found that online intrinsic calibration did not improve the estimate (the non-calibrating version had 0.49 m error), indicating that the Kalibr parameters were very accurate in this scenario.

4.3 Multi-IMU VINS

In this section, we develop a multi-IMU VINS (mi-VINS) algorithm that is able to improve estimation by combining the information from multiple IMU, while additionally being resilient to IMU sensor failures. In particular, we propose an EKF-based estimation architecture that generalizes the multi-state constraint Kalman filter (MSCKF) from a single IMU setup [94] to a multi-IMU case, such that the proposed estimator can readily utilize all the information from multiple IMUs and can seamlessly recover from unit failures. Moreover, in order to properly fuse the information of multiple IMUs, the proposed mi-VINS performs online sensor calibration of both the spatial (relative pose) and temporal (time offset) calibration parameters between sensors. Specifically, the main contributions of this section include [32]:

- We propose a tightly-coupled EKF-based estimation architecture that optimally

fuses asynchronous measurements from multiple IMUs and a camera. In particular, we maintain and propagate each IMU's state estimate as well as the joint covariance, and by arbitrarily choosing one IMU as the base whose poses are stochastically cloned as in the standard MSCKF, we can perform an EKF update using both visual measurements and rigid-body relative pose constraints between the IMUs.

- We perform *online* sensor calibration refinement of both the spatial *and* temporal calibration parameters between all sensors, allowing us to consistently fuse their asynchronous measurements without the need to perform a tedious offline calibration process, instead relying on only rough initial guesses.
- The proposed mi-VINS is resilient to IMU sensor failure. If the base IMU fails, an auxiliary IMU is "promoted" as a new base, offering uninterrupted localization solutions via estimate and covariance propagation.
- The proposed mi-VINS is validated in both simulations and real experiments, and is shown to provide high-precision navigation even in cases of sensor failures.

4.3.1 Multi-IMU Estimation

As compared to the standard MSCKF [94] that uses only a single IMU, we generalize it to incorporate an inertial state for *each* IMU on the sensor platform allowing for update and propagation of each. If $(N+1)$ IMUs are rigidly mounted on the sensor platform, their state vector is given by:

$$\mathbf{x}_I = \begin{bmatrix} \mathbf{x}_{I_0}^\top & \cdots & \mathbf{x}_{I_N}^\top \end{bmatrix}^\top \tag{4.22}$$

where $\mathbf{x}_{I_i}$ is the navigation state of the i-th IMU [118]:

$$\mathbf{x}_{I_i} = \begin{bmatrix} {}^{I_i}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_i}^\top & {}^{G}\mathbf{v}_{I_i}^\top & \mathbf{b}_{i\omega}^\top & \mathbf{b}_{ia}^\top \end{bmatrix}^\top \tag{4.23}$$

To allow for utilizing visual feature measurements, we select an *arbitrary* IMU to serve as the "base", denoted by $\{I_b\}$, which can be changed over the trajectory if needed. As in the standard MSCKF, we keep a sliding window of stochastically cloned poses of this base IMU. Specifically, at time step k (which is corresponding to the k-th received image), we also maintain a sliding window of the base IMU clones at M past imaging times t_j $(j = k-M+1, \cdots, k)$, alongside with $\mathbf{x}_I$:

$$\mathbf{x}_{cl} = \begin{bmatrix} {}^{I_b(t_k)}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_b(t_k)}^\top & \cdots & {}^{I_b(t_{k-M+1})}_{G}\bar{q}^\top & {}^{G}\mathbf{p}_{I_b(t_{k-M+1})}^\top \end{bmatrix}^\top \tag{4.24}$$

Lastly, as will be explained in further sections, we also estimate *online* the spatial and temporal calibration parameters, $\mathbf{x}_e$, between each sensor and the base IMU. In particular, we maintain in our calibration state the relative pose between the base IMU and *ith* (out of M) camera, ${}^{C_i}\mathbf{x}_{I_b} = \begin{bmatrix} {}^{C_i}_{I_b}\bar{q}^\top & {}^{C_i}\mathbf{p}^\top_{I_b} \end{bmatrix}^\top$, as well as the pose between each auxiliary IMU and the base, ${}^{I_b}\mathbf{x}_{I_i} = \begin{bmatrix} {}^{I_i}_{I_b}\bar{q}^\top & {}^{I_b}\mathbf{p}^\top_{I_i} \end{bmatrix}^\top$. In addition to these spatial quantities, we also estimate the time offset between the base camera and base IMU, ${}^{C_b}t_{I_b}$, as well as the time offsets between each auxiliary IMU and the base, ${}^{I_i}t_{I_b}$. This results in our total state vector as [see (4.22) and (4.24)]:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_I^\top & \mathbf{x}_{cl}^\top & \mathbf{x}_e^\top \end{bmatrix}^\top = \begin{bmatrix} \mathbf{x}_I^\top & \mathbf{x}_S^\top \end{bmatrix}^\top \tag{4.25}$$

$$\mathbf{x}_e = \begin{bmatrix} {}^{C_0}\mathbf{x}_{I_b}^\top & \dots & {}^{C_M}\mathbf{x}_{I_b}^\top & {}^{I_b}\mathbf{x}_{I_0}^\top & \dots & {}^{I_b}\mathbf{x}_{I_N}^\top & {}^{C_b}t_{I_b} & {}^{I_0}t_{I_b} & \dots & {}^{I_N}t_{I_b} \end{bmatrix}^\top \tag{4.26}$$

where $\mathbf{x}_S$ is the set of *static* quantities whose true values do not evolve over time.

4.3.2 mi-VINS Propagation

IMU (gyroscope and accelerometer) measurements, which are used to propagate state estimates and covariance, are given by:

$$\boldsymbol{\omega}_{im} = {}^{I_i}\boldsymbol{\omega}_{I_i} + \mathbf{b}_{i\omega} + \mathbf{n}_{i\omega} \tag{4.27}$$

$$\mathbf{a}_{im} = {}^{I_i}\mathbf{a}_{I_i} + {}^{I_i}_{G}\mathbf{R}\,{}^{G}\mathbf{g} + \mathbf{b}_{ia} + \mathbf{n}_{ia} \tag{4.28}$$

where ${}^{I_i}\boldsymbol{\omega}_{I_i}$ and ${}^{I_i}\mathbf{a}_{I_i}$ are the true angular velocity and local linear acceleration of the i-th IMU, $\mathbf{n}_{i\omega}$ and $\mathbf{n}_{ia}$ are continuous-time Gaussian white noises corrupting the measurements, and ${}^{G}\mathbf{g} \simeq [0\ 0\ 9.81]^\top$ is the gravity in the global frame.

At the current time step k (corresponding to the current imaging time t_k), we propagate the current state estimate forward to the next time step $k+1$ (corresponding to the next new image time t_{k+1}), by using all the IMU measurements available in the time window $[t_k, t_{k+1}]$, which are denoted by $\mathcal{I}_i$ for the i-th IMU ($i = 0, \cdots, N$), based on the IMU dynamics $\mathbf{g}(\cdot)$ [17]:

$$\mathbf{x}_{I_i}(t_{k+1}) = \mathbf{g}\left(\mathbf{x}_{I_i}(t_k), \mathcal{I}_i, \mathbf{n}_{I_i}\right) \tag{4.29}$$

$$\Rightarrow \hat{\mathbf{x}}_{I_i}(t_{k+1}) = \mathbf{g}\left(\hat{\mathbf{x}}_{I_i}(t_k), \mathcal{I}_i, \mathbf{0}\right) \tag{4.30}$$

where $\mathbf{n}_{I_i}$ is the stacked vector of the IMU noises. The error covariance is propagated as follows (see [118]):

$$\mathbf{P}(t_{k+1}) = \mathbf{\Phi}_k \mathbf{P}(t_k) \mathbf{\Phi}_k^\top + \mathbf{Q}_k \tag{4.31}$$

$$\mathbf{\Phi}_k = \mathbf{Diag}\left(\mathbf{\Phi}_0(t_{k+1}, t_k), \ldots, \mathbf{\Phi}_N(t_{k+1}, t_k), \mathbf{I}\right) \tag{4.32}$$

$$\mathbf{Q}_k = \mathbf{Diag}\left(\mathbf{Q}_0, \ldots, \mathbf{Q}_N, \mathbf{0}\right)$$

where $\mathbf{\Phi}_i(t_{k+1}, t_k)$ is the linearized state-transition matrix for the error state of the i-th IMU across the time interval $[t_k,\ t_{k+1}]$ and $\mathbf{Q}_i$ is the corresponding noise covariance, while $\mathbf{Diag}\left(\cdot,\cdot,\cdot\right)$ places the argument matrix entries on the block diagonals of an otherwise zero matrix. Each of these matrices are computed *per IMU* using its measurements. Note that in the above, the identity matrix $\mathbf{I}$ in $\mathbf{\Phi}_k$ and the right-bottom zero matrix $\mathbf{0}$ in $\mathbf{Q}_k$ correspond to the static states [see (4.24) and (4.25)].

4.3.3 mi-VINS Update

To limit the navigation drift accumulated over propagation, we utilize both multi-IMU rigid constraints and camera visual measurements to perform efficient EKF updates.

4.3.3.1 Enforcing Multi-IMU Constraints

As all IMUs are rigidly connected, at any time t we have the following relative transformation between the base and non-base IMUs:

$$\prescript{I_i}{I_b}{\bar{q}} = \prescript{I_i(t)}{G}{\bar{q}} \otimes \prescript{I_b(t)}{G}{\bar{q}}^{-1}, \quad \prescript{I_b}{}{\mathbf{p}}_{I_i} = \prescript{I_b(t)}{G}{\mathbf{R}} \left(\prescript{G}{}{\mathbf{p}}_{I_i(t)} - \prescript{G}{}{\mathbf{p}}_{I_b(t)}\right)$$

where $\prescript{I_i}{I_b}{\bar{q}}$ and $\prescript{I_b}{}{\mathbf{p}}_{I_i}$ are the *fixed* relative pose between the base IMU b and the i-th IMU. The residual associated with this constraint for each IMU can be written as:

$$\begin{cases} 2\mathbf{vec}\left(\prescript{I_i(t)}{G}{\bar{q}} \otimes \prescript{I_b(t)}{G}{\bar{q}}^{-1} \otimes \prescript{I_i}{I_b}{\bar{q}}^{-1}\right) & = \mathbf{0} \\ \prescript{G}{}{\mathbf{p}}_{I_i}(t) - \prescript{G}{}{\mathbf{p}}_{I_b}(t) - \prescript{G}{I_b(t)}{\mathbf{R}} \prescript{I_b}{}{\mathbf{p}}_{I_i} & = \mathbf{0} \end{cases} \Rightarrow \mathbf{r}_{I_i}(\mathbf{x}) = \mathbf{0} \tag{4.33}$$

where $\text{vec}(\bar{q}) = \mathbf{q}$ returns the vector portion of the argument quaternion $\bar{q}$. We stack this constraint for each auxiliary IMU to form a system of residuals, $\mathbf{r}_I(\mathbf{x}) = \mathbf{0}$. Linearization of these residuals at the current state estimate yields:

$$\mathbf{r}_I(\hat{\mathbf{x}}) + \frac{\partial \mathbf{r}_I}{\partial \tilde{\mathbf{x}}}\tilde{\mathbf{x}} = \mathbf{0} \;\Rightarrow\; \mathbf{0} - \mathbf{r}_I(\hat{\mathbf{x}}) = \frac{\partial \mathbf{r}_I}{\partial \tilde{\mathbf{x}}}\tilde{\mathbf{x}} \tag{4.34}$$

Note that this is a *hard* constraint that acts as a measurement with *zero* noise. Alternatively, we may loosen this constraint by adding a small, synthetic noise to this measurement. Such a treatment may be useful to handle unmodeled errors such as flexible calibration. In practice, such constraints may quickly degrade performance due to inaccuracies in the calibrated transforms between sensors, which motivates us to perform online calibration of these parameters. It should be noted that such relative-pose constraints have been used in previous multi-IMU systems [7], along with a constraint on the relationship between the IMUs' velocities. However, transferring velocities from one frame to another requires angular velocity measurements, which have already been used in the propagation step, thereby introducing unmodeled correlations. As such, we choose to forgo this constraint to ensure consistency. In addition, as explained in Section 4.3.4, we utilize these relative-pose constraints to perform both spatial and temporal calibration of the sensors. Note that while in this work we assume the spatial calibration parameters remain static, i.e., that the sensors are rigidly mounted, they can also be modeled as random walks when dealing with a more flexible mount. Such a treatment can also be extended to the temporal parameters.

4.3.4 Online Spatial/Temporal Sensor Calibration

As in the multi-camera scenario, we wish to account for the fact that the multi-IMU system may not be perfectly precalibrated, and to therefore refine the spatial and temporal parameters online. In the case of a multi-IMU system that consists of *asynchronous* independent inertial sensors with non-negligible time offsets relative to the base IMU clock, in addition to errors in the rigid transformation estimates, inaccurate time offsets can greatly impact the multi-IMU constraints, Eq. (4.33), and thus the

estimation, which motivates us to perform online calibration of these parameters. We model each IMU's offset from the clock of the base IMU in the standard manner:

$$ {}^{I_b}t = {}^{I_i}t + {}^{I_i}t_{I_b} \tag{4.35} $$

To determine this time offset ${}^{I_i}t_{I_b}$, we estimate it online as an additional random variable included in our state.

In order to correctly enforce the asynchronous multi-IMU relative pose constraint (4.33), each IMU must be expressed at the *same* time, and therefore this time offset must be compensated for. To this end, we propagate the i-th IMU's state up to the *estimate* of the current base IMU time as expressed in the i-th clock, ${}^{I_i}\hat{t} = {}^{I_b}t - {}^{I_i}\hat{t}_{I_b}$. Explicitly, the IMU state after propagation at time step $k+1$ can be written as [see (4.22)]:

$$ \mathbf{x}_I({}^{I_b}t_{k+1}) = \begin{bmatrix} \mathbf{x}_{I_0}({}^{I_0}\hat{t}_{k+1})^\top & \cdots & \mathbf{x}_{I_N}({}^{I_N}\hat{t}_{k+1})^\top \end{bmatrix}^\top $$

where $\mathbf{x}_{I_i}({}^{I_i}\hat{t}_{k+1})$ is the true value of the i-th IMU at the estimated time ${}^{I_i}\hat{t}_{k+1}$ in its own clock.

Using the first-order approximation for the motion of each IMU, the state of the i-th IMU at the time of the base IMU is a function of the state at the estimated time and the error in the time offset estimate, i.e.,

$$ \begin{aligned} {}^G\mathbf{p}_{I_i}({}^{I_i}t_{k+1}) &= {}^G\mathbf{p}_{I_i}({}^{I_b}t_{k+1} - {}^{I_i}\hat{t}_b - {}^{I_i}\tilde{t}_{I_b}) = {}^G\mathbf{p}_{I_i}({}^{I_i}\hat{t}_{k+1} - {}^{I_i}\tilde{t}_{I_b}) \\ &\approx {}^G\mathbf{p}_{I_i}({}^{I_i}\hat{t}_{k+1}) - {}^G\mathbf{v}_{I_i}({}^{I_i}\hat{t}_{k+1}){}^{I_i}\tilde{t}_{I_b} \end{aligned} \tag{4.36} $$

$$ {}^{I_i({}^{I_i}t_{k+1})}_{\quad G}\mathbf{R} \approx \mathrm{Exp}\left({}^{I_i}\boldsymbol{\omega}_{I_i}({}^{I_i}\hat{t}_{k+1}){}^{I_i}\tilde{t}_{I_b}\right) {}^{I_i({}^{I_i}\hat{t}_{k+1})}_{\quad G}\mathbf{R} \tag{4.37} $$

where $\mathrm{Exp}(\cdot)$ is the matrix exponential mapping an axis-angle to a rotation matrix and $\boldsymbol{\omega}_i({}^{I_i}\hat{t}_{k+1})$ is the angular velocity of the i-th IMU. With that, we rewrite (4.33) in a residual form (for simplicity time index $k+1$ is dropped):

$$ \mathbf{r}_{\theta_i} = 2\mathbf{vec}\left(\begin{bmatrix} -\frac{1}{2}{}^{I_i}\boldsymbol{\omega}_{I_i}({}^{I_i}\hat{t}){}^{I_i}\tilde{t}_{I_b} \\ 1 \end{bmatrix} \otimes {}^{I_i({}^{I_i}\hat{t})}_{\quad G}\bar{q} \otimes {}^{I_b({}^{I_b}t)}_{\quad G}\bar{q}^{-1} \otimes {}^{I_i}_{I_b}\bar{q}^{-1} \right) \tag{4.38} $$

$$\mathbf{r}_{p_i} = {}^{G}\mathbf{p}_{I_i}({}^{I_i}\hat{t}) - {}^{G}\mathbf{v}_{I_i}({}^{I_i}\hat{t}){}^{I_i}\tilde{t}_{I_b} - {}^{G}\mathbf{p}_{I_b}({}^{I_b}t) - {}^{G}_{I_b({}^{I_b}t)}\mathbf{R}\,{}^{I_b}\mathbf{p}_{I_i} \tag{4.39}$$

The Jacobians of these constraints can be computed as:

$$\begin{aligned}
&\frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_i}\tilde{t}_{I_b}} = -\mathbf{A}\left(\boldsymbol{\omega}_{mi} - \hat{\mathbf{b}}_{i\omega}\right), \quad \frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_i}\tilde{\boldsymbol{\theta}}_{I_b}} = -\mathbf{A}^{\top}, \quad \frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_i({}^{I_i}\hat{t})}\tilde{\boldsymbol{\theta}}_{G}} = \mathbf{A} \\
&\frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_b({}^{I_b}t)}_{G}\tilde{\boldsymbol{\theta}}} = -\Big((q_{l,4}\mathbf{I}_3 - \lfloor \mathbf{q}_l \times \rfloor)(q_{r,4}\mathbf{I}_3 + \lfloor \mathbf{q}_r \times \rfloor) - \mathbf{q}_l\mathbf{q}_r^{\top} \Big) \\
&\frac{\partial \mathbf{r}_{p_i}}{\partial {}^{G}\tilde{\mathbf{p}}_{I_i}({}^{I_i}\hat{t})} = \mathbf{I}_3, \quad \frac{\partial \mathbf{r}_{p_i}}{\partial {}^{G}\tilde{\mathbf{p}}_{I_b}({}^{I_b}t)} = -\mathbf{I}_3, \quad \frac{\partial \mathbf{r}_{p_i}}{\partial {}^{I_i}\tilde{t}_{I_b}} = -{}^{G}\hat{\mathbf{v}}_{I_i}({}^{I_i}\hat{t}) \\
&\frac{\partial \mathbf{r}_{p_i}}{\partial {}^{I_b({}^{I_b}t)}_{G}\tilde{\boldsymbol{\theta}}} = {}^{G}_{I_b({}^{I_b}t)}\hat{\mathbf{R}}\lfloor {}^{I_b}\hat{\mathbf{p}}_{I_i} \times \rfloor, \quad \frac{\mathbf{r}_{p_i}}{\partial {}^{I_b}\tilde{\mathbf{p}}_{I_i}} = -{}^{G}_{I_b({}^{I_b}t)}\hat{\mathbf{R}}
\end{aligned} \tag{4.40}$$

where we have used these definitions: $\mathbf{A} = (q_{res,4}\mathbf{I}_3 + \lfloor \mathbf{q}_{res} \times \rfloor)$, $\bar{q}_{res} = {}^{I_i({}^{I_i}\hat{t})}_{G}\hat{\bar{q}} \otimes {}^{I_b({}^{I_b}t)}_{G}\hat{\bar{q}}^{-1} \otimes {}^{I_i}_{I_b}\hat{\bar{q}}^{-1}$, $\bar{q}_l = {}^{I_i({}^{I_i}\hat{t})}_{G}\hat{\bar{q}} \otimes {}^{I_b({}^{I_b}t)}_{G}\hat{\bar{q}}^{-1}$ and $\bar{q}_r = {}^{I_i}_{I_b}\hat{\bar{q}}^{-1}$.

As in the camera-cloning, the value of the angular velocity used in the linearization [see (4.37)] comes from the i-th IMU's gyro measurements, i.e., $\boldsymbol{\omega}_{im}({}^{I_i}\hat{t}) - \hat{\mathbf{b}}_{i\omega}({}^{I_i}\hat{t})$. As before, errors in the linear and angular velocities are multiplied by ${}^{I_i}\tilde{t}_{I_b}$, and thus do not affect the measurement up to first order. It is also important to note that, after update with the multi-IMU relative-pose constraints, the state of the i-th non-base IMU will still be at the *prior* estimate at time ${}^{I_i}\hat{t}_{k+1|k}$ (i.e., the time we last propagated the IMU i to), while we will have an *updated* estimate for the time offset ${}^{I_i}\hat{t}_{I_b}$. To compensate for this, when we receive a new camera image at time ${}^{I_b}t_{k+2}$, propagation for each IMU is actually performed across the interval $[{}^{I_i}\hat{t}_{k+1|k}, {}^{I_i}\hat{t}_{k+2|k+1}]$. In summary, we propagate each of the IMU states and the joint covariance using measurements from each IMU. The states are propagated to the same estimated imaging time, after which we enforce the derived spatial-temporal multi-IMU relative-pose constraints (4.38)-(4.39) between the base and auxiliary IMUs through the EKF update.

4.3.5 Resilience to IMU Sensor Failures

In extreme environmental conditions, robustness to sensor failures is key to lifelong persistent localization. As compared to a failure of a stereo camera where the system can continue on monocular vision, failure of the IMU prevents VINS from

performing state estimation. In this section, we describe how the proposed mi-VINS is resilient to IMU sensor failures and allows for up to N IMU sensors to fail before the system completely stops working.

In particular, we consider the most challenging scenario where the base IMU sensor fails, as when a non-base IMU sensor fails it is trivial to just marginalize its navigation state. During base failure, we "promote" an auxiliary sensor to be the new base (which is denoted as the n-th IMU). We then transform the quantities in our state to be expressed with respect to this new base. Specifically, each IMU clone refers to the base IMU sensor frame at the true imaging time, t_j, and we can write the following transformations for each:

$$ {}^{I_n(t_j)}_{G}\mathbf{R} = {}^{I_n}_{I_b}\mathbf{R}\,{}^{I_b(t_j)}_{G}\mathbf{R} \tag{4.41} $$

$$ {}^{G}\mathbf{p}_{I_n}(t_j) = {}^{G}\mathbf{p}_{I_b}(t_j) + {}^{I_b(t_j)}_{G}\mathbf{R}^{\top}\,{}^{I_b}\mathbf{p}_{I_n} \tag{4.42} $$

In the case of online spatial calibration, we propagate them to the new base IMU as follows:

$$ {}^{I_i}_{I_n}\mathbf{R} = {}^{I_i}_{I_b}\mathbf{R}\,{}^{I_n}_{I_b}\mathbf{R}^{\top},\ {}^{I_n}\mathbf{p}_{I_i} = {}^{I_n}_{I_b}\mathbf{R}\left({}^{I_b}\mathbf{p}_{I_i} - {}^{I_b}\mathbf{p}_{I_n}\right) \tag{4.43} $$

$$ {}^{C_k}_{I_n}\mathbf{R} = {}^{C_k}_{I_b}\mathbf{R}\,{}^{I_n}_{I_b}\mathbf{R}^{\top},\ {}^{C_k}\mathbf{p}_{I_n} = {}^{C_k}\mathbf{p}_{I_b} + {}^{C_k}_{I_b}\mathbf{R}\,{}^{I_b}\mathbf{p}_{I_n} \tag{4.44} $$

Lastly, we note that the relationship between any clock, the base clock, and the new base clock takes the form ${}^{I_n}t + {}^{I_n}t_{I_b} = {}^{I_i}t + {}^{I_i}t_{I_b} \Rightarrow {}^{I_i}t_{I_n} = {}^{I_i}t_{I_b} - {}^{I_n}t_{I_b}$. Using these constraints, we can modify the estimates such that the n-th IMU serves as the new base, through a proper mean and covariance propagation. This procedure can be triggered at any point, such as when base sensor failure is detected, allowing for *continuous, uninterrupted* estimation. For clarity, the main steps of the proposed mi-VINS with online sensor calibration are outlined in Algorithm 1.

4.3.6 Multi-IMU Experimental Results

To validate the proposed mi-VINS we have performed both Monte-Carlo simulations and real-world tests, while focusing on evaluating the resilience to IMU sensor failures and the accuracy of online spatial and temporal calibration.

Algorithm 1 mi-VINS with online sensor calibration

INPUT: New image available at time ${}^{C}t_m$
if IMU failure detected **then**
  if Failed IMU is base **then**
    Switch to new IMU base.
  end if
  Marginalize failed IMU state.
end if
Propagate each IMU state estimates up to time ${}^{I_i}\hat{t}_m = {}^{C}t_m + {}^{C}\hat{t}_{I_b} - {}^{I_i}\hat{t}_{I_b}$ along with the joint covariance.
Clone the base IMU pose at the true imaging time ${}^{C}t_m$(4.8).
Perform feature tracking
Perform EKF update with the spatial-temporal multi-IMU relative-pose constraints (4.39) and MSCKF measurements.

4.3.6.1 Monte-Carlo Simulations

In simulations, we consider a stereo visual-inertial (VI) system, while in our real-world tests shown later a monocular VI system is validated. Specifically, we simulated an Asctec Firefly UAV equipped with a stereo-VI sensor using the Gazebo simulator [71] traveling in a dynamic 3D trajectory (see Figure 4.6). A series of 7 IMUs were placed 1.5 meters radially around the body frame of the UAV with random orientations. Ground-truth inertial measurements were collected at 400Hz and corrupted using the characteristics of an ADIS16448 IMU using the standard discrete-time simulation method of [44] (along with simulating biases), while image measurements were corrupted by one pixel noise at a rate of 20Hz. Constant (static) time offsets were subtracted from each IMU and camera readings introducing the time errors between measurements. The mi-VINS was initialized with perturbed ground-truth calibration of varying magnitudes, and 30 Monte-Carlo runs were performed, where each run represented a different realization of the measurement noises. The system was initialized from rest such that each IMU's starting velocity was zero, while the base IMU was initialized with the ground-truth pose. After this pose of the base IMU was determined, the IMU-IMU calibration parameters were used in a covariance propagation to initialize the pose of each auxiliary IMU. For this experiment, a sliding window size of

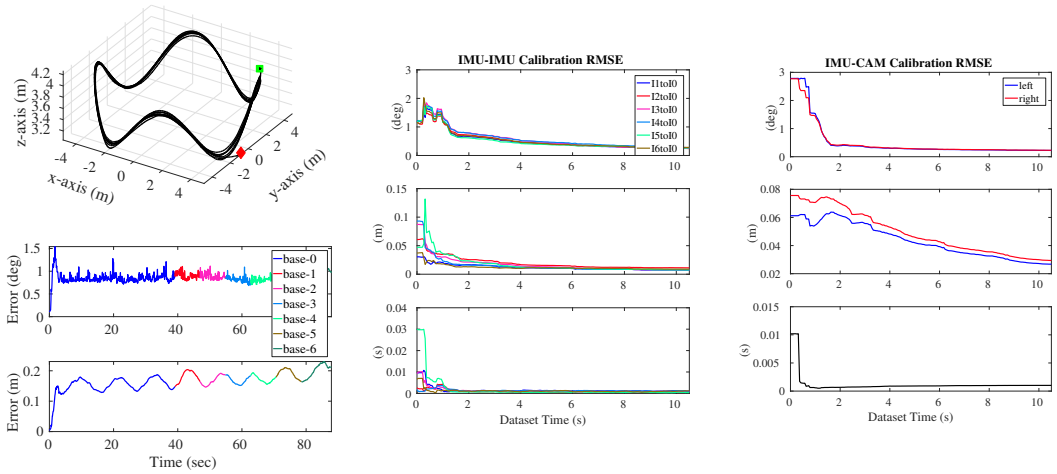


Figure 4.6: Monte-Carlo simulation results: (top-left) 250 meter long 3D trajectory with the start and end locations denoted as green square and red diamond, respectively. (bottom-left) RMSE of pose estimates, color-coded segments based on which base IMU was active. (center) IMU-IMU calibration RMSE of the first 10 seconds of the dataset (as it converges after that). (right) IMU-CAM calibration RMSE for the first 10 seconds for the stereo pair transformation and time offset.

15 poses was used in the MSCKF.

After waiting 40 seconds to ensure convergence of calibration parameters, the base IMU was switched off every 6 seconds until only a single IMU sensor remained to mimic repeated sensor failures. Figure 4.6 (bottom-left) shows the root mean squared errors (RMSE) of the trajectory estimates, where the color-coded different segments are obtained by using different base IMUs. For continuous trajectory estimates, we recorded the poses of IMU 0 at every timestep using the current estimates for the base IMU and the spatial calibration parameters between the base IMU and IMU 0. From the IMU-IMU calibration results shown Figure 4.6 (center), we clearly see that the temporal and spatial calibrations quickly converge to the ground-truth. Finally, the IMU-CAM calibration shown in Figure 4.6 (right) converges towards the ground-truth from poor initial guesses. These results show the ability to both perform accurate online localization with resilience to multiple sensor failures (in this case 6 sensors failed in total).

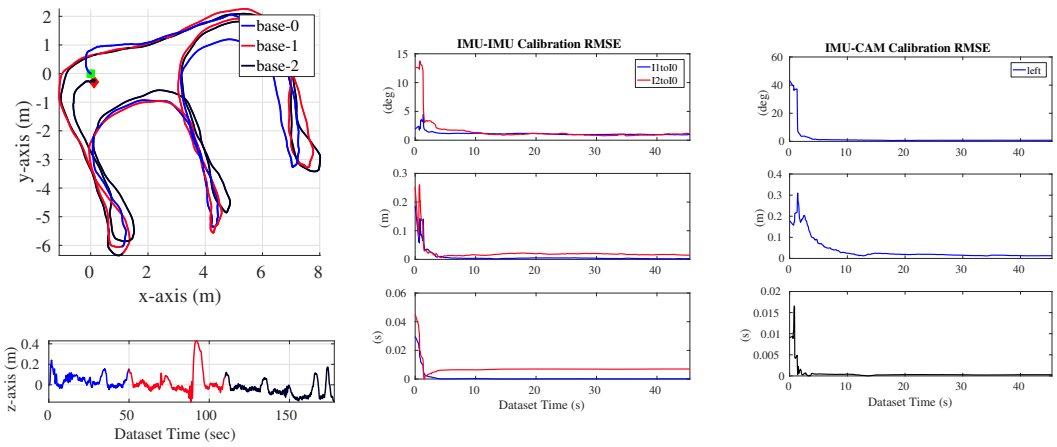


Figure 4.7: Experimental results: (left) Top-down view and z-axis trajectory estimates (about 143-meter long) with different base IMUs. (center) IMU-IMU spatial and temporal calibration results and (right) IMU-camera calibration results.

4.3.6.2 Real-World Tests

In our real-world tests, we assembled the multi-IMU VI system shown in Figure 4.8, which consists of a Pointgrey Blackfly camera with a wide angle lens, MTI-100, MTI-g710, and VI-sensor [98] with an ADIS16448 IMU. We used the Blackfly camera as a monocular image feed, while the MTI-g710, MTI-100, and VI-sensor ADIS16448 IMU were used as the three inertial sensors. The MTI-g710 acted as the base IMU until 50 seconds into the dataset when switching to the MTI-100 sensor was triggered, followed by a switching to the ADIS16448 IMU 100 seconds in. It should be pointed out that this order of IMU failures is arbitrary as the proposed mi-VINS can handle the failure of any of the IMUs (base or non-base), while we here focus only on the most challenging case of *repeated* base IMU failures. KLT tracking [6] using the implementation from OpenCV [11] was performed on incoming image measurements, while outliers were rejected through 8-point RANSAC. When features reached the sliding window size (10 poses in this experiment) or were lost, they were used in the MSCKF update.

We evaluate our sensor calibration performance in comparison to the results obtained from the `Kalibr` calibration toolbox [44] which is an offline process and is

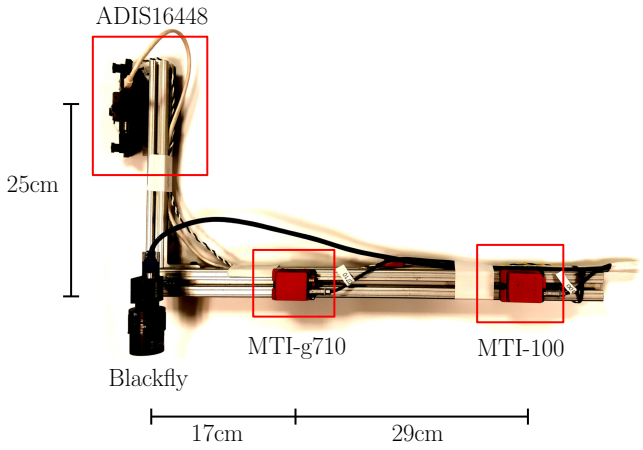


Figure 4.8: Multi-IMU VI-sensor used in our real-world experiments.

expected to be close to the ground-truth. These parameters were manually perturbed to form very poor initial guesses for the system in order to demonstrate robustness. The results in Figure 4.7 clearly show that the proposed mi-VINS achieves impressive localization and calibration performance. Two of the three IMUs "failed" during the trajectory but did not impact the estimation performance. As no ground-truth was available for this test, we returned to the starting location and computed the accumulated position error which is 0.35 meters, only about 0.24% of distance traveled. In addition the proposed system remained real-time, with an average processing time of 0.023 seconds (43 Hz) per image, compared to the 20 Hz rate of the camera.

4.4 MiMc: Multi-IMU Multi-Camera VINS

Having previously introduced both multi-IMU and multi-camera VIO systems, we implemented both functionalities within our previously developed OpenVINS framework [49]. OpenVINS is an open-sourced visual-inertial estimation framework with particular application to extending the basic MSCKF, and offers a large amount of simulation and evaluations tools. Additionally its standard single IMU-camera system has been shown to outperform existing open sourced state-of-the-art methods, and therefore we focus on improvements compared to this base. The resulting system, which is able to perform estimation and online calibration of an arbitrary numbers of

IMUs and cameras, is termed **M**ulti-**I**MU, **M**ulti-**C**amera VINS (MiMc). To achieve this fusion, we perform the multi-IMU propagation and update (Section 4.3) when receiving *base* camera images, and utilize the multi-camera interpolation-based update to process all other images (Section 4.2). We next present several extensions to the previous mc-VINS and mi-VINS systems that MiMc leverages for improved generality and performance.

4.4.1 Higher Order Interpolation

We note that the previously discussed multi-camera method utilizes an assumption of *linear* evolution in the orientation and position of the platform. While this may be adequate in many scenarios, in the case of highly-dynamic motion, this model does not hold. In particular, if we perform update using this improper model, this assumption *adds information* to the estimator that may be inconsistent, thereby hurting possible performance gains from adding additional sensors.

To address this issue, we propose to utilize higher-order representations to capture some of the more complex motion profiles seen in practice. In particular, we consider fitting a polynomial of order n to a set of n+1 poses. Explicitly (note we simplify ${}^{C_b}t := t$)

$$ {}^{I(t)}_{G}\mathbf{R} = \text{Exp}\left(\sum_{i=1}^{n} \mathbf{a}_{\theta i}\Delta t^i\right) {}^{I(t_0)}_{G}\mathbf{R} \tag{4.45} $$

$$ {}^{G}\mathbf{p}_{I(t)} = {}^{G}\mathbf{p}_{I(t_0)} + \sum_{i=1}^{n} \mathbf{a}_{pi}\Delta t^i \tag{4.46} $$

Here t_0 represents the time of the oldest pose being fitted to, and $\Delta t = t - t_0$ (note that due to representing this polynomial as the "change from t_0", we trivially recover the pose at t_0 when plugging in $\Delta t = 0$).

Fitting this polynomial involves estimating the parameters $\mathbf{a}_{\theta i}$ and $\mathbf{a}_{pi}$. To do so, we require n additional poses that, at each corresponding time, the polynomial should fit exactly:

$$ {}^{I(t_k)}_{G}\mathbf{R} = \text{Exp}\left(\sum_{i=1}^{n} \mathbf{a}_{\theta i}\Delta t_k^i\right) {}^{I(t_0)}_{G}\mathbf{R} \tag{4.47} $$

$$\Rightarrow \sum_{i=1}^{n} \mathbf{a}_{\theta i} \Delta t_k^i = \text{Log}\left({}^{I(t_k)}_{G}\mathbf{R} {}^{I(t_0)}_{G}\mathbf{R}^\top \right) := \Delta \boldsymbol{\phi}_k \tag{4.48}$$

$${}^{G}\mathbf{p}_{I(t_k)} = {}^{G}\mathbf{p}_{I(t_0)} + \sum_{i=1}^{n} \mathbf{a}_{pi} \Delta t_k^i \tag{4.49}$$

$$\Rightarrow \sum_{i=1}^{n} \mathbf{a}_{pi} \Delta t_k^i = {}^{G}\mathbf{p}_{I(t_k)} - {}^{G}\mathbf{p}_{I(t_0)} := \Delta \mathbf{p}_k \tag{4.50}$$

These constraints can be stacked in order to solve for the stacked vectors of coefficients:

$$\begin{bmatrix} \Delta \boldsymbol{\phi}_1 \\ \Delta \boldsymbol{\phi}_2 \\ \vdots \\ \Delta \boldsymbol{\phi}_n \end{bmatrix} = \begin{bmatrix} \Delta t_1 & \Delta t_1^2 & \cdots & \Delta t_1^n \\ \Delta t_2 & \Delta t_2^2 & \cdots & \Delta t_2^n \\ \vdots & \vdots & \ddots & \vdots \\ \Delta t_n & \Delta t_n^2 & \cdots & \Delta t_n^n \end{bmatrix} \begin{bmatrix} \mathbf{a}_{\theta 1} \\ \mathbf{a}_{\theta 2} \\ \vdots \\ \mathbf{a}_{\theta n} \end{bmatrix} \tag{4.51}$$

$$\Delta \boldsymbol{\phi} = \mathbf{V} \mathbf{a}_\theta \tag{4.52}$$

$$\mathbf{a}_\theta = \mathbf{V}^{-1} \Delta \boldsymbol{\phi} \tag{4.53}$$

$$\mathbf{a}_p = \mathbf{V}^{-1} \Delta \mathbf{p} \tag{4.54}$$

We note that when using order 1, we recover the linear solution used by the standard mc-VINS. These interpolation equations can then be used to solve for the corresponding measurement Jacobians, as shown in Appendix F.2. In order to implement this in our formulation, for each auxiliary camera measurement we query the poses which bound the estimated measurement time to form the polynomial for fitting. We attempt to make segments such that the measurement falls in the "middle" of the set of poses. For example, when forming an odd order polynomial, we try to ensure that we have $(n+1)/2$ poses earlier and newer than the measurement. If we do not have enough poses earlier or later, we simply grab the earliest/latest $n+1$ poses.

4.4.2 Rolling Shutter

Until this section we have assumed that the cameras utilized have been global shutter. That is, we have assumed that all measured pixel intensities were captured by the camera at the exact same time. However, certain low-cost cameras utilize a *rolling*

shutter, wherein each row of the image is captured sequentially. This leads to very large errors in the VIO if this effect is not taken into account. Let the total readout time of the ith camera be t_{ri}, and ${}^{C_i}t$ denote the nominal time that the camera reports for the image. Then the rolling shutter model measurement function for a pixel captured in the mth row (out of M total) of camera i is given by [80]:

$$\mathbf{z}_k = \mathbf{w}_i \left(\mathbf{\Pi} \left({}^{C_i(t)}\mathbf{p}_f \right), \boldsymbol{\zeta}_i \right) + \mathbf{n}_k \tag{4.55}$$

$$t = {}^{C_i}t + {}^{C_i}t_{C_b} + \frac{m}{M} t_{ri} \tag{4.56}$$

where ${}^{C_i}t$ is the nominal imaging start time in the ith camera clock and ${}^{C_i}t_{C_b}$ is the time offset from the base camera as previously discussed. Note that the rolling shutter effect simply adds a further offset into the time the measurement was captured. Thus, these measurements can be seamlessly incorporated into our multi-camera system with one slight modification: *all* measurements, even those of the base camera, must be expressed using interpolation. We note that now the cloning of the IMU occurs at the true *nominal* image time of the base camera. In the case we calibrate this readout time, we simply have an extra Jacobian with respect to this unknown parameter:

$$\frac{\partial \mathbf{z}_k}{\partial \tilde{t}_{ri}} = \frac{m}{M} \frac{\partial \mathbf{z}_k}{\partial {}^{C_i}\tilde{t}_{C_b}} \tag{4.57}$$

Note that certain works may define this nominal imaging time in different ways (start of image, end of image, middle of image), which simply requires minor modification of (4.56) [79, 53].

4.4.3 First Estimate Jacobians

In contrast to the previous mi-VINS and mc-VINS, MiMc leverages First Estimate Jacobians (FEJ) [64], wherein the linearization points for all variables (except calibration) used when computing Jacobians corresponds to the first estimate the filter ever had for that parameter to maintain consistency of the system. This is because while the "ideal" visual-inertial EKF leveraging the true values for linearization has 4 unobservable directions (corresponding to global position and yaw), numerical errors

introduced by linearizing about *different* estimates for the *same* quantity at different timesteps (due to updates) makes unobservable directions falsely observable. Thus the estimator thinks it is gaining information in spurious directions, destroying the consistency (and accuracy) of the filter. FEJ remedies this by only ever linearizing with a *single*, first estimate, ensuring the correct observability of the system. This has been shown to greatly increase the accuracy of VIO in the single IMU-camera pair scenario [79].

For the multi-IMU propagation, we utilize FEJ to evaluate the analytical aspects of each IMU's state-transition matrix, in a manner analogous to [79]. As compared to the standard single-IMU case, to maintain consistency across the multi-IMU system, we compute each auxiliary IMU's linearization point using the FEJ estimates for the base IMU pose along with the *best* estimate for the IMU-IMU calibration. Owing to the fact that we perform linearization in this described way, it can be shown that the standard multi-IMU relative pose update Jacobians take the simpler form (See Eq. (4.40)):

$$\frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_i}\tilde{t}_{I_b}} = -\left(\boldsymbol{\omega}_{mi} - \hat{\mathbf{b}}_{i\omega}\right) \tag{4.58}$$

$$\frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_i}_{I_b}\tilde{\boldsymbol{\theta}}} = -\mathbf{I}, \tag{4.59}$$

$$\frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_i}_{G}{}^{({}^{I_i}\hat{t})}\tilde{\boldsymbol{\theta}}} = \mathbf{I} \tag{4.60}$$

$$\frac{\partial \mathbf{r}_{\theta_i}}{\partial {}^{I_b}_{G}{}^{({}^{I_b}t)}\tilde{\boldsymbol{\theta}}} = -{}^{I_i}_{I_b}\hat{\mathbf{R}} \tag{4.61}$$

For the multi-camera interpolation, when evaluating the measurement Jacobians involved, we utilize FEJ for all the poses involved with fitting the polynomial, but use the method proposed by both [80] and [101] when computing the residual. In particular, when evaluating the pixel measurement residual, we use the saved IMU readings to propagate from the closest neighbor clone estimate to compute the pose at the pixel measurement time, as this offers a more accurate estimate for the interpolated pose

mean. However, enforcing FEJ on the Jacobians as described guarantees the correct observability of the system.

4.4.4 MiMc-VINS Simulaton Results

In this section we thoroughly evaluate the proposed MiMc-VINS using extensive simulation.

4.4.4.1 B-Spline Trajectory Representation

At the center of the OpenVINS simulator is an $\mathbb{SE}(3)$ B-spline which allows for the calculation of the pose, velocity, and accelerations at any given timestep along a trajectory. For all simulations, the input is a pose trajectory file which is uniformly sampled and a B-spline is fitted to. The three simulated trajectories we evaluate the system on are shown in Figure 4.9. The first simulated dataset is called "Gore" and is based on the output of OpenVINS operating on a 240 meter sequence collected in the University of Delaware's Gore Hall in which three floors are traversed. The second dataset is called "Outdoor" and was collected as the output of OpenVINS for a sequence collected from a moving car which travels along three levels of a parking garage before exiting and driving along the road, and has total length of approximately 1800 meters. The third scenario is highly dynamic and based on the groundtruth of the "Tum Corridor 1" dataset from the TUM VI benchmark datasets [113] which we simply term "Tum" and is approximately 295 meters.

4.4.4.2 Inertial Measurements

To compute the inertial measurements for each IMU, we use the B-spline representation to model the trajectory of the base IMU. We then leverage the B-spline to compute the following quantities: ${}^{G}_{I_b}\mathbf{R}$, ${}^{G}_{I_b}\dot{\mathbf{R}}$, ${}^{G}_{I_b}\ddot{\mathbf{R}}$, ${}^{G}\mathbf{p}_{I_b}$, ${}^{G}\dot{\mathbf{p}}_{I_b}$, and ${}^{G}\ddot{\mathbf{p}}_{I_b}$, where as usual we define ${}^{G}\dot{\mathbf{p}}_{I_b} := {}^{G}\mathbf{v}_{I_b}$, and ${}^{G}\ddot{\mathbf{p}}_{I_b} := {}^{G}\mathbf{a}_{I_b}$. The angular velocity, ${}^{I_b}\boldsymbol{\omega}_{I_b}$, and angular acceleration, ${}^{I_b}\boldsymbol{\alpha}_{I_b}$, can be computed using:

$$ {}^{G}_{I_b}\dot{\mathbf{R}} = {}^{G}_{I_b}\mathbf{R}\lfloor {}^{I_b}\boldsymbol{\omega}_{I_b}\times\rfloor \tag{4.62} $$

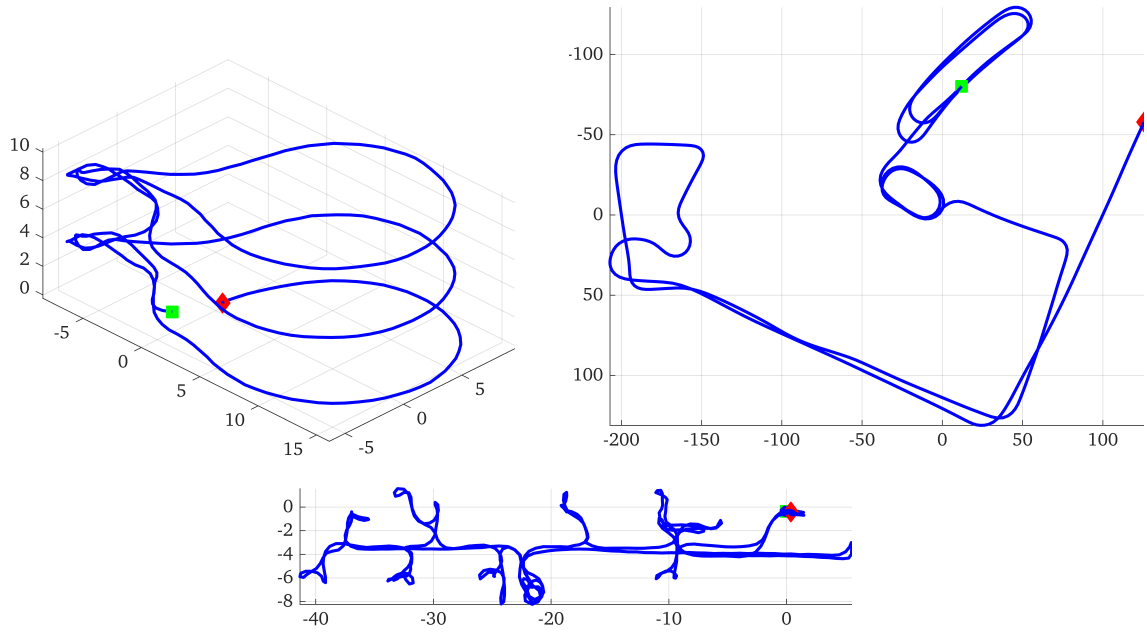


Figure 4.9: Simulated trajectories, axes are in units of meters. Going clockwise from top left: Gore, Outdoor, and Tum. Green square denotes the start and red diamond denotes the end.

$$\Rightarrow {}^{I_b}\boldsymbol{\omega}_{I_b} = \left({}^{G}_{I_b}\mathbf{R}^{\top}\,{}^{G}_{I_b}\dot{\mathbf{R}}\right)^{\vee} \tag{4.63}$$

$${}^{G}_{I_b}\ddot{\mathbf{R}} = {}^{G}_{I_b}\dot{\mathbf{R}}\lfloor {}^{I_b}\boldsymbol{\omega}_{I_b}\times\rfloor + {}^{G}_{I_b}\mathbf{R}\lfloor {}^{I_b}\boldsymbol{\alpha}_{I_b}\times\rfloor \tag{4.64}$$

$$\Rightarrow {}^{I_b}\boldsymbol{\alpha}_{I_b} = \left({}^{G}_{I_b}\mathbf{R}^{\top}\left({}^{G}_{I_b}\ddot{\mathbf{R}} - {}^{G}_{I_b}\dot{\mathbf{R}}\lfloor {}^{I_b}\boldsymbol{\omega}_{I_b}\times\rfloor\right)\right)^{\vee} \tag{4.65}$$

where the $(\cdot)^{\vee}$ extracts the vector portion of a skew symmetric matrix. With this, the global accelerations and local linear velocities can be generated using the rigid body constraints [107]:

$$\begin{aligned} {}^{G}\mathbf{a}_{I_i} &= {}^{G}\mathbf{a}_{I_b} + {}^{G}_{I_b}\mathbf{R}\left(\lfloor {}^{I_b}\boldsymbol{\omega}_{I_b}\times\rfloor\lfloor {}^{I_b}\boldsymbol{\omega}_{I_b}\times\rfloor + \lfloor {}^{I_b}\boldsymbol{\alpha}_{I_b}\times\rfloor\right) {}^{I_b}\mathbf{p}_{I_i} \\ {}^{I_i}\boldsymbol{\omega}_{I_i} &= {}^{I_i}_{I_b}\mathbf{R}\,{}^{I_b}\boldsymbol{\omega}_{I_b} \end{aligned} \tag{4.66}$$

In summary, we can use the above equations to generate the synthetic true IMU measurements for each IMU at any given time. Note that the time each IMU measurement is generated takes into account the true time offset between the base and auxiliary IMU. These true measurements are then corrupted using the random walk biases and corresponding white noises.

4.4.4.3 Visual-Bearing Measurements

After initializing the B-spline trajectory, environmental landmarks are generated using the procedure in OpenVINS. Specifically, a static map of features is generated by incrementing along the trajectory and if the number of projected features fall below the desired feature count threshold, random feature bearing rays are generated from each camera and assigned a random depth. These generated features are then appended to the environmental map and projected into future frames. To generate feature measurements in the case of a global shutter camera (which is the current model used by OpenVINS), environmental features would simply be projected onto the camera's image plane using its true pose and intrinsics.

In the experiments that follow, we focus on the case of rolling shutter cameras. Rolling shutter measurements were generated with a modification to the standard feature projection procedure used by OpenVINS using the method proposed in [80]. In particular, due to the rolling shutter effect one cannot simply utilize the groundtruth feature positions and camera poses to project onto the image plane, as each row was captured at a different time. Therefore, we start with the nominal image time, t_0, and perform standard global shutter projection for a given feature. This projection yields a row that the feature projected to, m_0. Based on this row, we then compute a *new* imaging time based on the rolling shutter effect by Equation (4.56):

$$t_1 = t_0 + \frac{m_0}{M} t_{ri} \tag{4.67}$$

Using this new time, we recompute the projection for the pose at t_1, and iterate until convergence (typically 2-3 iterations are needed). This ensures that a given feature measurement was captured at the pose time corresponding to that image row. These true raw pixel coordinates were then corrupted by drawing from a one-pixel standard deviation Gaussian describing the noise. For the six cameras considered, we used rolling shutter readout times of 10, 15, 20, 17, 8, and 10 milliseconds, respectively, while each image stream was simulated at 10, 11, 13, 23, 18, and 22 Hz in order to highlight the

asynchronicity of sensors that can be handled by the system. Lastly, each IMU was simulated at 400 Hz.

4.4.4.4 Consistency / Convergence of MiMc

To validate the consistency of the entire system as well as the convergence of all calibration parameters, we ran the proposed system on the Tum dataset using three IMUs and three cameras. For each calibrated parameter, we drew from a prior distribution, with these noises listed in Table 4.1. In order to initialize the system, we utilized the groundtruth to set the initial values of the pose of the base IMU, as well as the velocity of every IMU in the rig. Using the initial, uncertain IMU-IMU transforms, we then used a straightforward estimate and covariance propagation to initialize the poses of all auxiliary IMUs. For these experiments, a sliding window size of 10 was used.

Figure 4.10 shows the estimation errors of the system operating on the Tum dataset. As can be seen, the errors of the estimator offer good consistency (that is, they stay within the 3σ bounds reported by the covariance), and additionally all calibration parameters converge to their correct values. For these experiments we found that inflating the noise of the multi-IMU rigid-body constraints during the first few seconds of the run (5e-3 for these experiments) before switching to a very small value (5e-5, rather than purely 0) offered the best performance/stability. Thus the IMU calibration parameters can be seen to change from slow to fast convergence after this switch (in these experiments, we used the first 5 seconds). Overall, these results validate our system's calibration performance as well as the resulting consistency of the estimator.

4.4.4.5 Effect of Adding Cameras

We next investigated the performance gains offered by additional sensing. First, we looked at the benefit of adding extra cameras into the system. Sweeping through 1-6 rolling shutter cameras with a single IMU, 30 Monte-Carlo runs were simulated for each

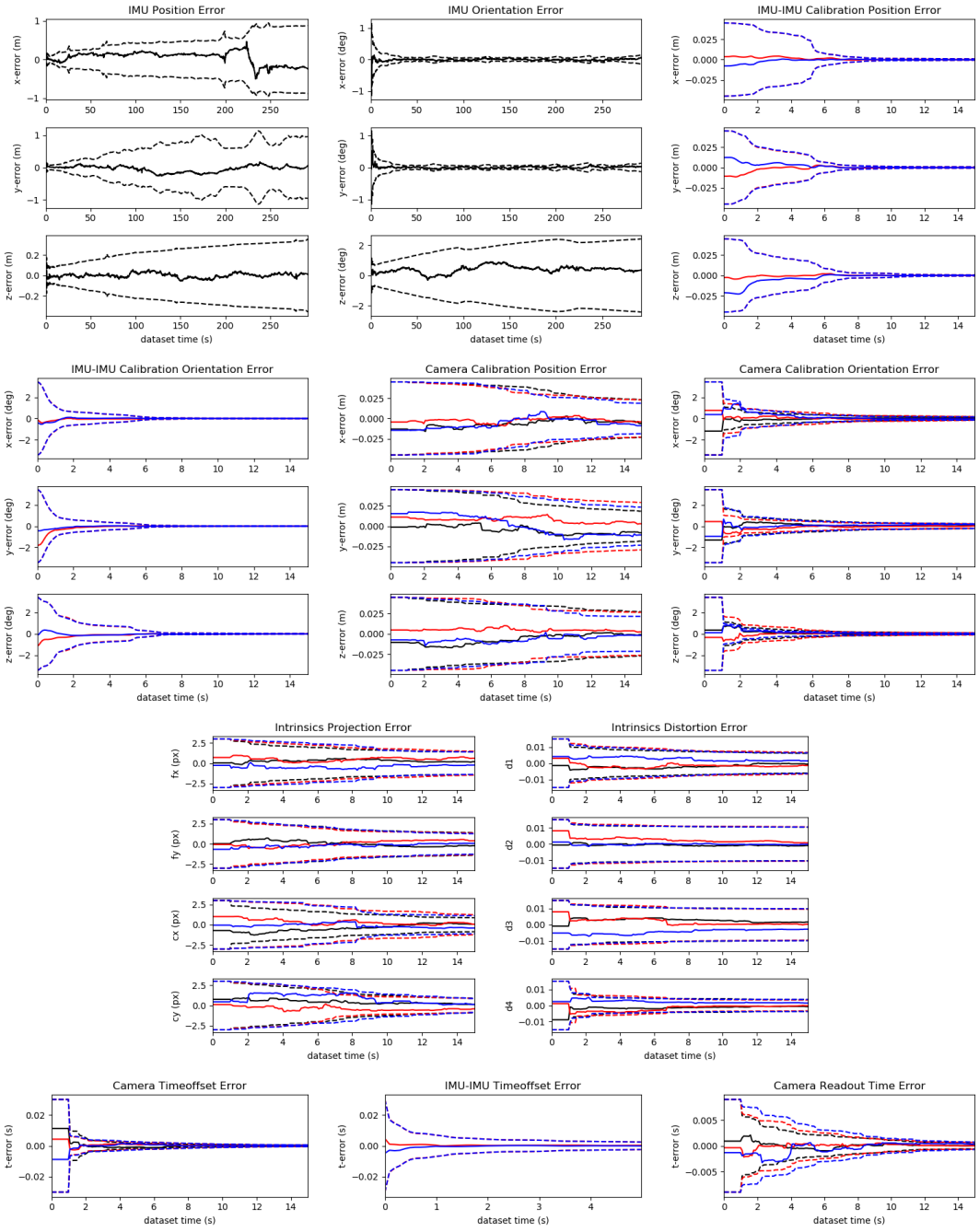


Figure 4.10: Multi-Sensor Calibration for three IMUs and three cameras on the simulated Tum dataset. Black, blue, and red respectively denote parameters relating to the base, second, and third sensor respectively (IMU or camera). Solid lines denote errors, while dotted lines of the corresponding color refer to the 3σ bounds reported by the estimator's covariance. Note that for the camera timeoffset plot, the base camera's timeoffset is with respect to the base *IMU*, while all others are with respect to the base *camera*. For calibration parameters, only the first few seconds are shown.

Table 4.1: Noise distributions used in the simulations.

Parameter	Value	Parameter	Value
Gyro bias turn-on noise	0.01	Accel bias turn-on noise	0.01
Gyro white noise	1.6968e-04	Gyro random walk	1.9393e-05
Accel white noise	2.0000e-3	Accel bias random walk	3.0000e-3
Focal length / camera center noise	1.0	Distortion noise	0.01
Relative orientation noise (rad)	0.017	Relative position noise (m)	0.01
Time offset noise (s)	0.01	Readout time noise (ms)	3.0

dataset and each sensor number. Each run represented a different realization of the measurement noises and initial calibration errors. To mimic the effects of measurement depletion due to low amounts of texture, each camera was only allowed to track 25 features in the environment at a time. In addition, each camera was given a non-overlapping field of view with the others. We stress again that for all experiments, the initial estimates for the calibration parameters were perturbed from their true values by drawing from a prior distribution. The exact noise parameters leveraged during simulation can be found in Table 4.1.

For these runs we report both the Absolute Trajectory Error (ATE), which is simply the average RMSE across the dataset, and the Relative Pose Error (RPE) [131]. The results for the system when using perturbed calibration parameters are shown in Figure 4.13. For both metrics, adding more camera information greatly improved estimation performance for every tested trajectory. In particular, moving from 1 camera to 6 reduced the ATE from 1.091 degrees and 0.983 meters across all datasets to just 0.331 degrees and 0.425 meters. For RPE, at every segment length tested, adding more cameras yielded lower relative pose errors. These results are to be expected, due to the fact that we are incorporating more information into our estimator in the form of additional measurements provided by the auxiliary cameras. In addition, because we introduced errors to our initial calibration estimates, these results validate that performance gains can be seen even if the system is not well calibrated beforehand.

Figure 4.11: ATE Errors in degrees/meters for Outdoor, Tum, and Gore datasets when performing online calibration with multiple cameras.

Number of Cameras	Outdoor	Tum	Gore	Average
1	0.277 / 1.479	1.173 / 0.829	1.824 / 0.640	1.091 / 0.983
2	0.250 / 1.133	0.891 / 0.444	1.057 / 0.450	0.733 / 0.676
3	0.231 / 1.017	0.649 / 0.374	0.752 / 0.375	0.544 / 0.589
4	0.228 / 0.945	0.468 / 0.270	0.557 / 0.297	0.418 / 0.504
5	0.214 / 0.852	0.451 / 0.240	0.518 / 0.252	0.394 / 0.448
6	**0.205 / 0.816**	**0.329 / 0.221**	**0.460 / 0.238**	**0.331 / 0.425**

Figure 4.12: RPE Errors in degrees/meters for the simulated datasets using different numbers of cameras. The lowest errors are highlighted in bold.

Number of Cameras	8m	16m	24m	32m	40m	48m
1	0.117 / 0.138	0.156 / 0.192	0.181 / 0.237	0.204 / 0.277	0.218 / 0.309	0.233 / 0.342
2	0.097 / 0.092	0.131 / 0.130	0.149 / 0.161	0.166 / 0.187	0.179 / 0.208	0.192 / 0.227
3	0.083 / 0.077	0.111 / 0.110	0.127 / 0.137	0.142 / 0.159	0.151 / 0.179	0.159 / 0.195
4	0.071 / 0.064	0.096 / 0.090	0.109 / 0.112	0.118 / 0.131	0.128 / 0.148	0.137 / 0.162
5	0.066 / 0.057	0.087 / 0.080	0.100 / 0.099	0.109 / 0.116	0.116 / 0.131	0.126 / 0.145
6	**0.062 / 0.051**	**0.082 / 0.073**	**0.094 / 0.091**	**0.103 / 0.108**	**0.110 / 0.123**	**0.117 / 0.136**

Figure 4.13: Boxplots of the RPE using different number of cameras. The lowest errors are highlighted in bold.

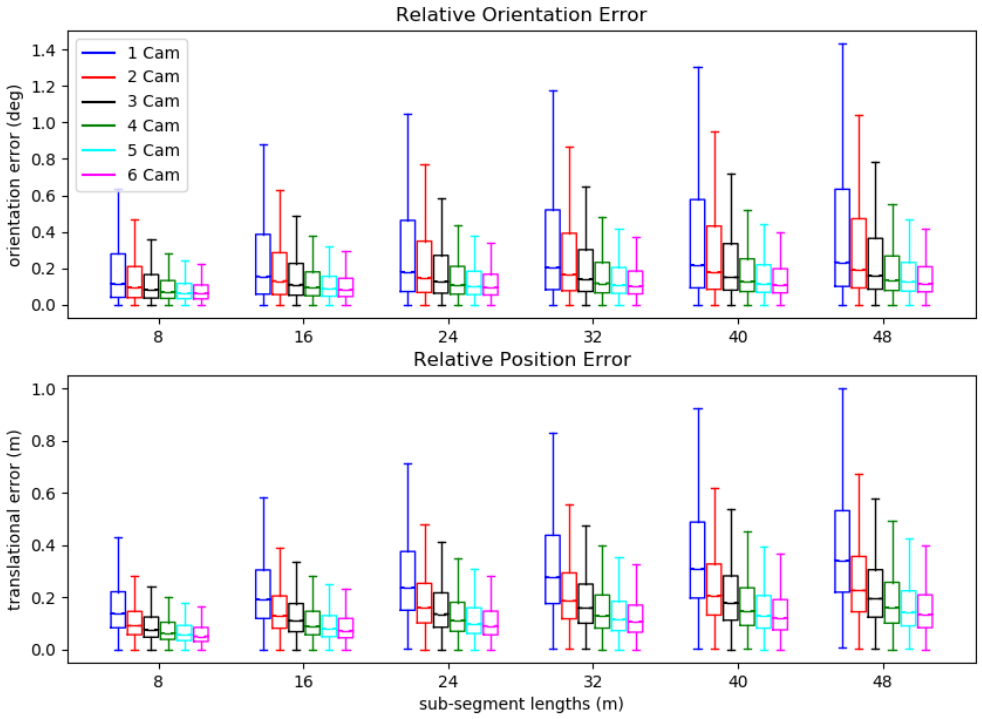


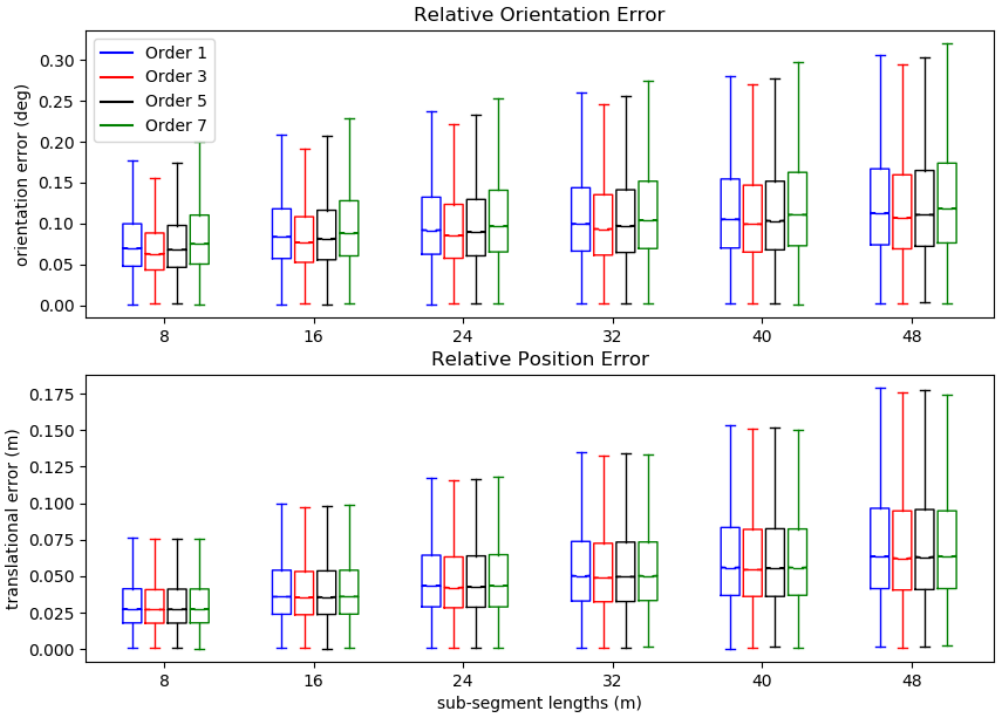
Figure 4.14: ATE Errors in degrees/meters for the Tum dataset when using different interpolation orders for six cameras. The lowest errors are highlighted in bold.

Interpolation Order	**ATE**
1	0.274 / 0.180
3	**0.270** / **0.179**
5	0.273 / 0.180
7	0.302 / 0.181

Figure 4.15: RPE Errors in degrees/meters for the simulated datasets using different interpolation orders. The lowest errors are highlighted in bold.

Interpolation Order	**8m**	**16m**	**24m**	**32m**	**40m**	**48m**
1	0.070 / 0.027	0.084 / 0.036	0.092 / 0.043	0.099 / 0.050	0.105 / 0.056	0.113 / 0.064
3	**0.063** / **0.027**	**0.077** / **0.035**	**0.085** / **0.042**	**0.093** / **0.049**	**0.099** / **0.054**	**0.107** / **0.062**
5	0.068 / 0.027	0.082 / 0.036	0.090 / 0.043	0.097 / 0.050	0.103 / 0.055	0.111 / 0.063
7	0.075 / 0.027	0.088 / 0.036	0.097 / 0.043	0.104 / 0.050	0.111 / 0.056	0.118 / 0.064

Figure 4.16: Boxplots of the RPE using different interpolation orders



4.4.4.6 Choice of Interpolation Order

A free choice in MiMc is what order polynomial to utilize for asynchronous camera fusion. As such, we next looked at how this choice of interpolation order affects performance. To do so, we tested on the most dynamic of the three datasets, Tum, and used an extremely low frequency of 5 Hz for the base camera. This was done in order to maximize the effect that order choice would have on the resulting estimate. Six cameras were utilized in total, and calibration was performed online. The RPE and ATE results are shown in Figure 4.16.

Interestingly, while accuracy (in terms of both metrics) is improved when moving from order one to order three, adding additional complexity to the polynomials tended to cause a *decrease* in accuracy. We conjecture that this may be due to the fact that there may be overfitting in the polynomial regression as the system "interprets" small errors in the estimates as higher-order motion. This is especially problematic as we leverage First Estimate Jacobians, and thus the system attempts to fit a polynomial to a series of suboptimal, *initial* estimates, rather than a smoothed window of the current *full* estimates. Overall, we found that order three polynomials gave the best performance, albeit by a small margin. We also note that one could alternatively utilize different orders for the position and orientation independently.

4.4.4.7 Effect of Adding IMUs

We next investigated the gains in accuracy possible when adding additional IMUs into the system. For these experiments, six IMUs were simulated with varying spatial offsets, while only a single rolling shutter camera was used. As in the last experiment, we ran 30 Monte-Carlo simulations of the estimator for each number of sensors used and for each dataset. The RPE and ATE metrics are shown in Figure 4.19. As evident, adding more IMUs always improves the RPE for every segment length tested, as well as the ATE for every trajectory. In particular moving from 1 IMU to 6 reduced the average ATE across all datasets from 1.091 degrees 0.983 meters to just 0.657 degrees and 0.630 meters. Overall, these results show that adding IMUs can dramatically

improve performance, and therefore validates our desire to add additional sensors into the system, even if a perfect prior calibration between sensors is unavailable.

4.4.4.8 Effect of FEJ on MiMc

Having validated the overall system's performance, we then investigated the improvements offered by enforcing First Estimate Jacobians on MiMc. We ran the same Monte Carlo analysis over the three datasets using different numbers of sensors, as well as *with and without* FEJ. The ATE and RPE results for this experiment are shown in Tables 4.20 and 4.21. For these experiments, we utilized up to 50 features per camera, and allowed up to 25 total of them to be added to the state as SLAM landmarks. As previously indicated in the individual multi-IMU and multi-camera sweeps, increasing the number of sensors greatly increases the accuracy of the system in terms of both ATE and RPE. This effect is seen whether or not FEJ is enforced. However, as is well-known is the single IMU-camera case [79], guaranteeing the correct observability *always* led to improved performance for all sensor combinations. In fact, this effect was so apparent that in some cases a lower number of sensors using FEJ could *outperform* a FEJ-less counterpart using *more* sensors (in terms of ATE). This can be seen in the case of 2 IMU, 2 Camera with FEJ, which achieves an ATE of 0.288 degrees and 0.285 meters, versus the "naive" 3 IMU, 3 Camera, with 0.462 degrees and 0.394 meters. Overall, these results indicate that FEJ plays an important role in the accuracy of the total system.

4.4.5 Real World Experimental Results

To evaluate the proposed system on real world data, we constructed a sensor platform consisting of three 640x480 ELP-960P2CAM-V90-VC USB 2.0 rolling shutter stereo cameras operating at 30 Hz. The platform has two IMUs consisting of an XSENS MT-100, Microstrain 3DM-GX-25 and can be seen in Figure 4.22. The three stereo pairs were placed in a semi-circular pattern giving an overall large field of view greater

Figure 4.17: ATE Errors in degrees/meters for Outdoor, Tum, and Gore datasets when performing online calibration with multiple IMUs. The lowest errors are highlighted in bold.

Number of IMU	Outdoor	Tum	Gore	Average
1	0.277 / 1.479	1.173 / 0.829	1.824 / 0.640	1.091 / 0.983
2	0.217 / 1.197	1.112 / 0.713	1.548 / 0.515	0.959 / 0.808
3	0.190 / 1.078	1.018 / 0.655	1.418 / 0.465	0.875 / 0.733
4	0.180 / 1.014	0.773 / 0.593	1.332 / 0.436	0.762 / 0.681
5	0.176 / 0.986	0.703 / 0.587	1.222 / 0.415	0.700 / 0.662
6	**0.166 / 0.922**	**0.663 / 0.571**	**1.143 / 0.398**	**0.657 / 0.630**

Figure 4.18: RPE Errors in degrees/meters for the large-scale simulated dataset using different numbers of IMUs. The lowest errors are highlighted in bold.

Number of IMU	8m	16m	24m	32m	40m	48m
1	0.117 / 0.138	0.156 / 0.192	0.181 / 0.237	0.204 / 0.277	0.218 / 0.309	0.233 / 0.342
2	0.098 / 0.126	0.137 / 0.176	0.160 / 0.217	0.184 / 0.251	0.194 / 0.280	0.203 / 0.306
3	0.091 / 0.120	0.123 / 0.167	0.144 / 0.206	0.163 / 0.236	0.177 / 0.264	0.182 / 0.289
4	0.081 / 0.115	0.111 / 0.160	0.131 / 0.196	0.147 / 0.225	0.161 / 0.251	0.168 / 0.273
5	0.078 / 0.112	0.106 / 0.156	0.126 / 0.191	0.140 / 0.219	0.153 / 0.244	0.162 / 0.264
6	**0.075 / 0.110**	**0.101 / 0.152**	**0.121 / 0.185**	**0.134 / 0.213**	**0.145 / 0.237**	**0.153 / 0.257**

Figure 4.19: Boxplots of the RPE using different number of IMU

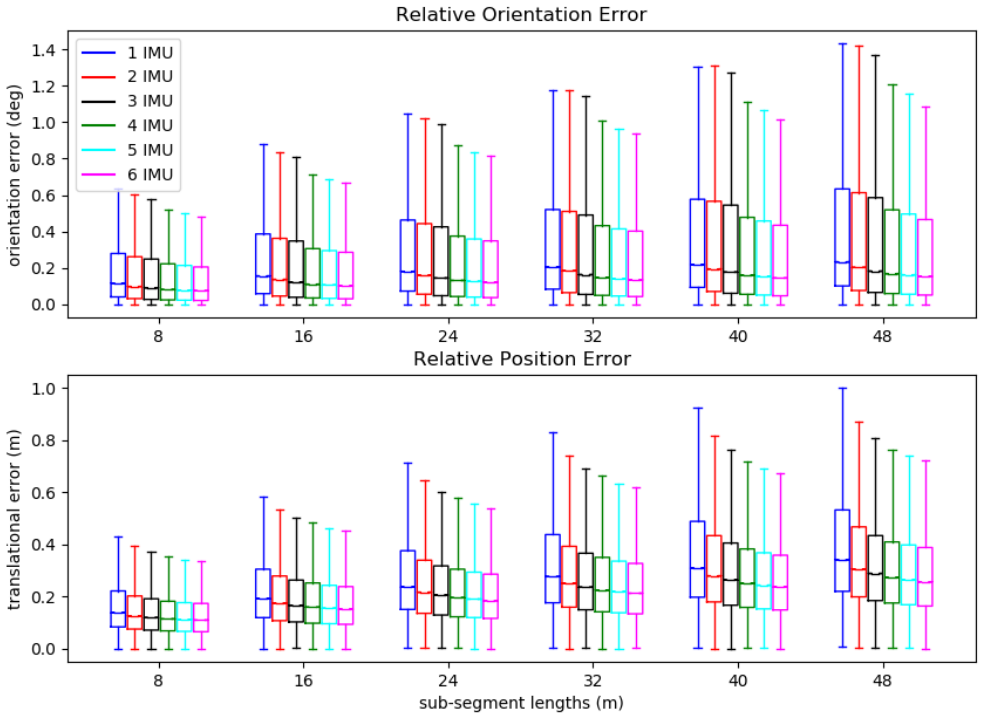


Figure 4.20: ATE Errors in degrees/meters for Outdoor, Tum, and Gore datasets for different amounts and sensors and with/without using FEJ.

Num IMUs, Num Cameras, FEJ	Outdoor	Tum	Gore	Average
1, 1, false	0.351 / 1.048	0.369 / 0.194	2.345 / 0.428	1.021 / 0.557
1, 1, true	0.239 / 0.886	0.237 / 0.153	0.768 / 0.223	0.415 / 0.421
2, 2, false	0.257 / 0.767	0.325 / 0.151	1.165 / 0.236	0.583 / 0.385
2, 2, true	0.174 / 0.588	0.224 / 0.115	0.466 / 0.153	0.288 / 0.285
3, 3, false	0.297 / 0.859	0.330 / 0.151	0.760 / 0.173	0.462 / 0.394
3, 3, true	0.145 / 0.525	0.177 / 0.113	0.307 / 0.112	0.210 / 0.250
4, 4, false	0.218 / 0.639	0.283 / 0.126	0.325 / 0.079	0.275 / 0.282
4, 4, true	0.133 / 0.468	0.167 / 0.089	0.226 / 0.064	0.175 / 0.207

Figure 4.21: ATE Errors in degrees/meters for Outdoor, Tum, and Gore datasets for different amounts and sensors and with/without using FEJ.

Num IMUs, Num Cameras, FEJ	8m	16m	24m	32m	40m	48m
1, 1, false	0.057 / 0.046	0.075 / 0.064	0.087 / 0.081	0.098 / 0.096	0.106 / 0.107	0.114 / 0.119
1, 1, true	0.055 / 0.045	0.071 / 0.064	0.081 / 0.080	0.090 / 0.093	0.096 / 0.103	0.103 / 0.111
2, 2, false	0.044 / 0.033	0.058 / 0.047	0.068 / 0.058	0.077 / 0.068	0.084 / 0.075	0.090 / 0.082
2, 2, true	0.042 / 0.032	0.054 / 0.045	0.062 / 0.055	0.069 / 0.064	0.073 / 0.071	0.078 / 0.076
3, 3, false	0.037 / 0.027	0.048 / 0.039	0.057 / 0.049	0.064 / 0.058	0.068 / 0.064	0.073 / 0.069
3, 3, true	0.036 / 0.027	0.046 / 0.038	0.053 / 0.047	0.058 / 0.055	0.062 / 0.061	0.065 / 0.065
4, 4, false	0.033 / 0.022	0.043 / 0.031	0.050 / 0.040	0.057 / 0.046	0.062 / 0.051	0.067 / 0.055
4, 4, true	0.030 / 0.021	0.039 / 0.030	0.045 / 0.037	0.049 / 0.042	0.052 / 0.047	0.056 / 0.050

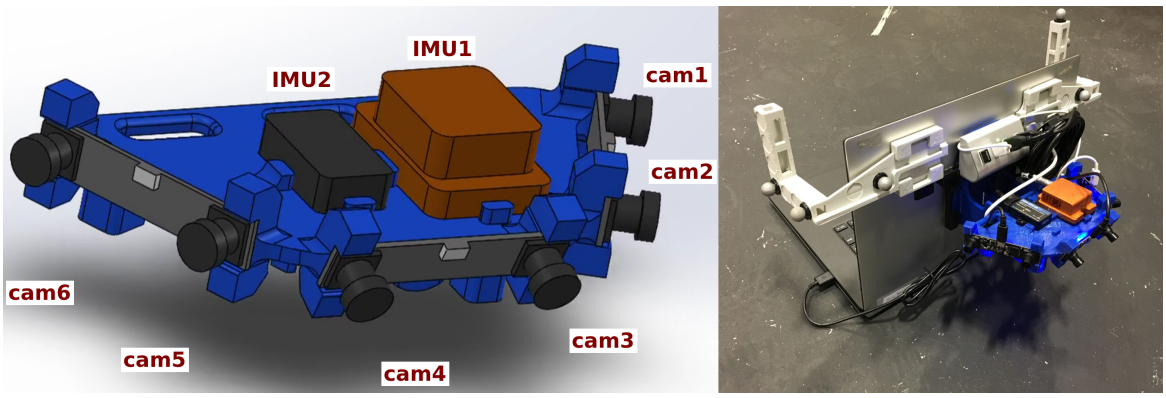


Figure 4.22: Constructed sensor platform with three rolling shutter pairs, XSENS MT-100, and Microstrain 3DM-GX-25 IMUs.

then 180°. Only the left image of each stereo pair was leveraged in the following experiments. Multiple datasets were collected in a large VICON warehouse, providing highly accurate groundtruth for comparison. The dataset trajectories (labeled as multicam 1-4) are 74, 91, 185, and 108 meters in total length, respectively. The groundtruth plots of these trajectories are shown in Figure 4.23.

The system was initialized from a stationary position. In particular, the base IMU collected accelerometer data during this period to estimate the direction of gravity, while the initial velocity estimate of each IMU was set to zero. Once a sufficient excitation of the base IMU was detected, the system began by forming estimates of the auxiliary IMU poses using the uncertain calibration as described in the simulation section. Finally, propagation and update began as normal. All system parameters were calibrated online with the initial guesses being based on offline calibration performed with the Kalibr toolbox [44].

Features were extracted uniformly using FAST [109] and tracked independently, for each camera's image stream using KLT [6], with outliers rejected via 8-point RANSAC. Each camera tracked at most 100 features in its image stream. Up to 25 features that were tracked longer than the sliding window period were added as SLAM features into the state vector. For all others, the MSCKF update was performed if the feature's first measurement was collected before the second oldest sliding window clone (that is, if it has data that will become "too old" after the next marginalization phase). When processing pixel measurement updates, we used a slightly inflated assumed noise sigma of 2 pixels in order to account for the unmodeled effect of image patch warping due to the rolling shutter cameras. We found experimentally that this led to improved accuracy of our system. For these experiments, we used a sliding window size of 12. Lastly, the system was forced to run in realtime, with new images being dropped if the previous frame was still undergoing processing. Thus increasing the number of sensors may lead to more dropped frames in certain instances.

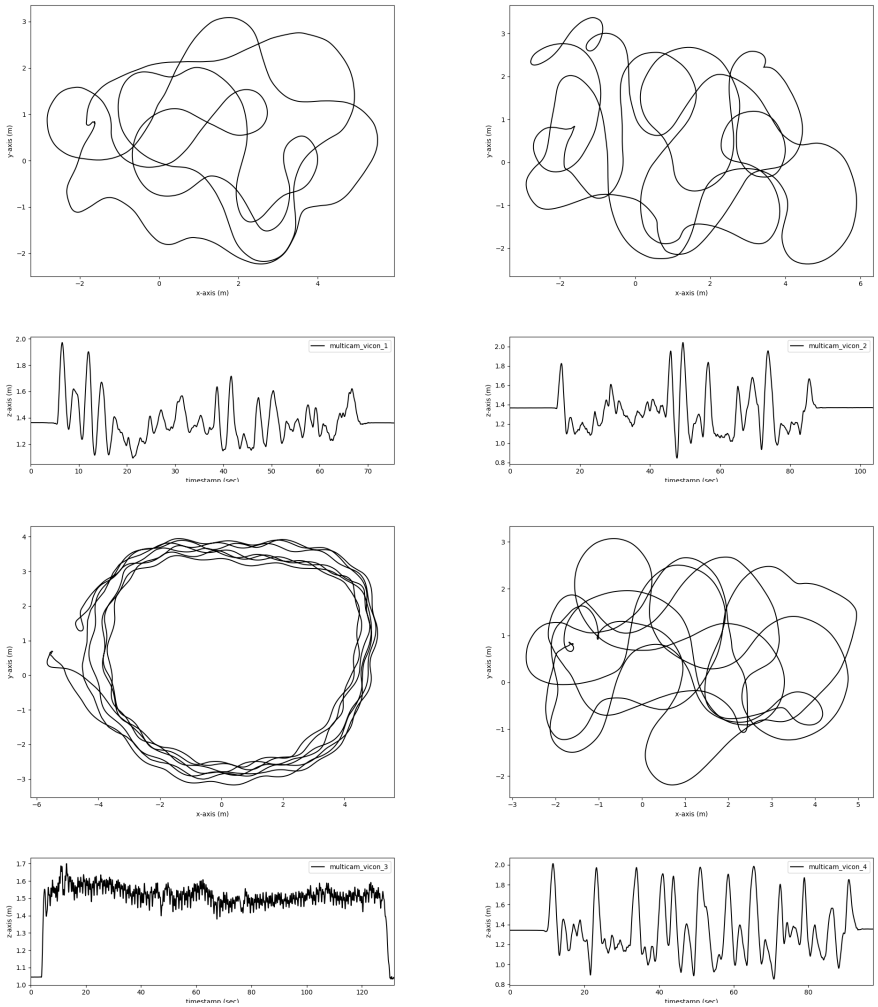


Figure 4.23: Trajectories of the collected multi-sensor datasets. For clarity, the top-down views and z-axis trajectory are plotted separately. Note that the z-axis plot for each trajectory is placed under its top-down counterpart.

4.4.5.1 Accuracy Validation

To further validate our choice of increased sensors, we processed each of the datasets using different numbers of IMUs and cameras. To handle the randomness caused by RANSAC-based frontends and possible dropped frames, we performed 30 runs for each sensor combination on each dataset. Because these results do not leverage the groundtruth to initialize, a 4 DOF alignment using the first estimated and true poses was performed for proper error evaluation [132]. The ATE and RPE results are shown in Table 4.25. As can be seen, incorporating additional sensing tends to greatly improve both metrics of estimation accuracy. In particular, the 3-camera, 2-IMU average ATE of 1.248 degrees and 0.193 meters was a great improvement over the single-camera, single-IMU which had an ATE of 2.716 degrees and 0.302 meters. We do note, however, that this improvement is not seen in all cases such as multicam_1, where 2 IMU and 1 camera outperformed 2 IMU 3 cameras in terms of ATE. This may be due to suboptimal tuning of the noise characteristics, especially for the IMU noises. In addition, we note that because the system was run in realtime, increasing the number of sensors may lead to more dropped frames in certain instances. This can be handled, for example, by more intelligent selection that spreads measurements *across* cameras. For RPE, adding more sensors led to improved performance in this metric at every tested segment length. Overall, these results clearly motivate our desire to add additional sensors into the system, and demonstrate that real world accuracy gains can be achieved even without requiring synchronized or accurately calibrated sensors.

4.5 Conclusion

In this chapter we have introduced a MSCKF-based VINS system that is able to fuse the information from an arbitrary number of asynchronous cameras and inertial measurement units. In particular, to limit the increase in computational burden from adding additional cameras, we only performed stochastic cloning at times corresponding to one of the cameras, and represented the state at intermediate times through efficient pose interpolation. In order to utilize multiple IMUs, we propagated

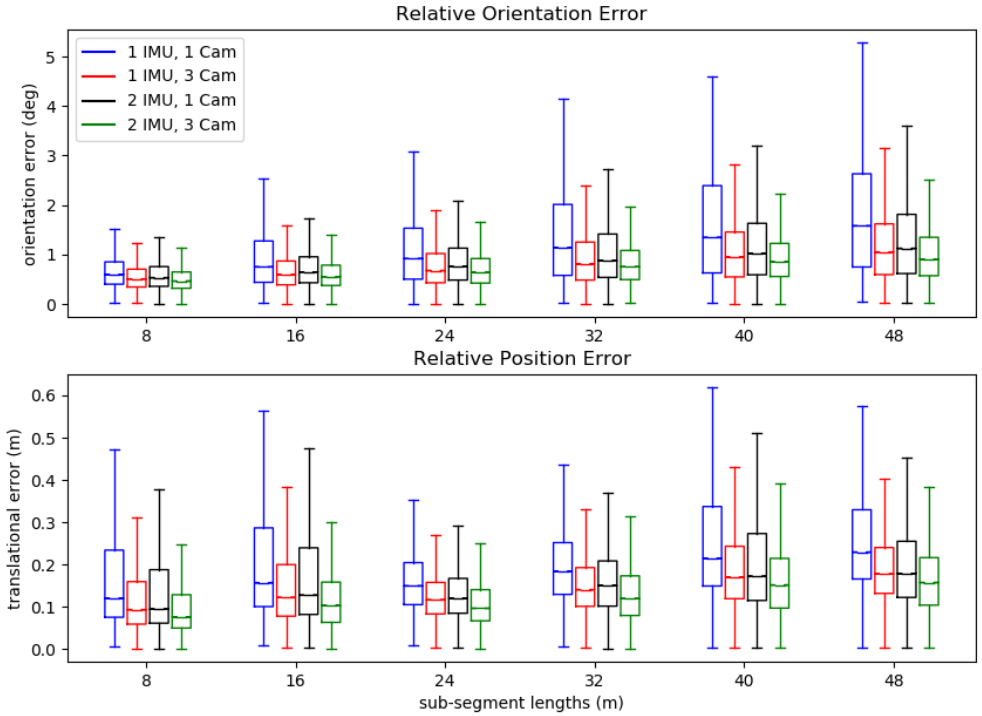
Figure 4.24: ATE errors in degrees/meters on the datasets for MIMC-VINS. Average of 30 runs. The lowest errors are highlighted in bold.

Num IMU, Num Camera	**multicam_1**	**multicam_2**	**multicam_3**	**multicam_4**	**Average**
1, 1	1.539 / 0.158	1.215 / 0.161	4.433 / 0.651	3.676 / 0.237	2.716 / 0.302
1, 3	1.249 / 0.180	1.027 / 0.191	2.215 / 0.472	2.102 / 0.162	1.648 / 0.251
2, 1	**1.186** / **0.139**	1.096 / 0.170	3.039 / 0.569	1.348 / 0.136	1.667 / 0.254
2, 3	1.350 / 0.156	**0.799** / **0.153**	**1.739** / **0.341**	**1.101** / **0.123**	**1.248** / **0.193**

Figure 4.25: RPE errors in degrees/meters on the datasets for the proposes algorithm. Average of 30 runs. The lowest errors are highlighted in bold.

Num IMU, Num Camera	**8m**	**16m**	**24m**	**32m**	**40m**	**48m**
1, 1	0.719 / 0.153	0.872 / 0.199	1.037 / 0.169	1.246 / 0.208	1.403 / 0.259	1.491 / 0.259
1, 3	0.646 / 0.119	0.796 / 0.149	0.917 / 0.125	1.040 / 0.154	1.145 / 0.190	1.183 / 0.186
2, 1	0.691 / 0.141	0.850 / 0.188	0.994 / 0.153	1.149 / 0.191	1.266 / 0.240	1.298 / 0.226
2, 3	**0.606** / **0.097**	**0.705** / **0.120**	**0.779** / **0.103**	**0.881** / **0.120**	**0.955** / **0.151**	**1.018** / **0.142**

Figure 4.26: Boxplots of the RPE using different interpolation orders



a joint system consisting of each IMU's navigation state and utilized an update based on the rigid-body constraints between each IMU to fuse their information. To achieve robust performance, online calibration was performed for the spatial and temporal calibration parameters between all sensors, as well as for the intrinsics of each camera (including rolling shutter readout time). Both functionalities were implemented within our previously developed OpenVINS codebase to formulate the MiMc system, and the consistency and performance gains of adding additional sensors were validated.

Chapter 5

VISUAL-INERTIAL LOCALIZATION AND TARGET TRACKING

Until this point we have only considered the problem of estimating the motion of a sensor platform using its onboard visual-inertial sensing equipment. However, in many robotics applications, not only is the user interested in estimating their own motion, but also the tracking of external bodies (targets) whose states are only indirectly observed through exteroceptive sensors mounted on the tracking robot. External object tracking may be necessary for safe navigation in dynamic environments, as in autonomous driving [123], or may even be the overall goal of the sensor deployment, as in military surveillance. For these reasons, the problem of simultaneous localization, mapping, and moving object tracking (SLAMMOT) is important but challenging [122]. However, estimating the poses and motion of external bodies using visual-inertial sensing has received less attention, with a few notable exceptions [36, 18].

In this chapter, we first present a Visual-Inertial Localization and Target Tracking (VILTT) system that tightly-couples the estimation of the motions of a sensor platform and an external, dynamic 3D rigid body [36] within the computationally efficient MSCKF framework. In this way, we can actually improve the performance of the VINS alone by leveraging information provided by target measurements. We offer several motion models for various targets seen in practice whose parameters are then estimated along with those of the moving sensor platform, and perform an extensive observability analysis of the combined system. The proposed tightly-coupled VILTT is then validated in both simulated real-world experiments.

Owing to the fact that we must pick a motion model for the target, a danger remains that this could be *inconsistently* chosen, which may actually degrade the VINS performance. To handle these scenarios, we propose a system that leverages

the Schmidt-Kalman Filter (SKF), an extension of the standard EKF which does not update a portion of the state while still consistently tracking all correlations. In this way, we do not allow the target measurements to update the navigation states, while at the same time we remain conservative about the uncertainty of the combined system. We show in both simulated and real-world experiments that this representation, along with a robot-centric representation of the target pose, leads to robust estimation performance even in the case of inconsistently chosen target models.

5.1 Related Work

5.1.1 Target Tracking

While SLAMMOT has seen study in other works [122, 116], few leverage visual-inertial sensors coupled with a pose representation of the target. In particular, many target tracking systems treat the target as a single 3D point [2, 18, 21]. With this, only measurements of this representative feature can be used to update the target, thereby ignoring the additional information provided by other features that typically exist on the object's body. Another limitation of a point particle model is that it requires continuous observation of this single point in order to update. However, in many scenarios the tracking robot may view the target from varied viewing angles thereby occluding this feature despite the object remaining in view of the sensor. Treating the target as a pose with a rigidly connected point cloud relaxes this assumption as we can gain information of the target state when *any* of its features are viewed.

Works that do perform full target-body pose estimation often leverage more advanced sensors such as LIDAR [5] or RGB-D cameras [3], limiting application for lightweight Micro-Aerial Vehicles (MAVs) due to weight and cost constraints. Other target pose-based estimation methods require additional prior knowledge to handle ambiguity of the estimation problem, such as the target's scale in monocular vision. For example, Li et al. [83] used a dimensional prior of the target and conditioned on the output of the ego-motion estimation (a decoupled approach), and thus did not account for the uncertainty in these estimates.

Henein et al. [55] presented a SLAMMOT framework for estimation in the presence of moving dynamic bodies. Each dynamic object's global feature point cloud and relative pose change is estimated at each timestep, while constraining that the evolution of these points followed rigid body motion. However, due to the large number of state variables introduced by estimating the target's point cloud at each timestep, this solution is offline. By contrast, in this work we estimate the target's point cloud in its local frame, while also leveraging a sliding window filtering framework, and estimate in real-time both the tracking robot and target poses.

The work closest to ours, introduced by Qiu et al. [104], utilized a robust monocular visual-inertial batch-based estimator [103] which was first used to track the motion of the sensor platform. The target object was detected using the learning-based object recognition system YOLO [106], and a vision-only structure-from-motion problem was then used to estimate (up-to-scale) the point cloud of the target as well as the relative pose between the camera and target, while metric scale was estimated using trace correlation. While this system was shown to offer robust monocular pose-tracking performance, due to the lack of explicitly estimating the target motion parameters, it is unclear whether this method can be used for active tracking purposes as future target states are not predicted. By contrast, our system fuses observations of the target in a tightly-coupled probabilistic formulation, and is able to improve overall trajectory accuracy through target observation information, its assumed motion model, and proper modeling of its uncertainty. As we additionally estimate motion parameters, we are also able to both predict the target's pose at future timesteps and provide an associated uncertainty of this prediction, which can be useful for active tracking scenarios [124].

5.1.2 Handling Poor Models

Previous methods handle model uncertainty by using multiple estimators, each using a different model, though these lead to a large computational increase. We refer the reader to [84] for a survey on such methods. In this work, we present an alternative

solution which can handle incorrect motion assumptions by leveraging the Schmidt-Kalman Filter [112] formulation, allowing for consistent estimation of the target while preventing corruption of the tracking robot's VINS.

The SKF formulation updates estimates for only a certain subset of variables, keeping the estimates of a set of "nuisance" parameters fixed, while still consistently tracking all correlations. Because the marginal covariance corresponding to these nuisance parameters is not updated, this leads to computational gains, which has been investigated within the VINS community to reduce complexity [26, 47, 48], here we leverage the SKF to prevent inconsistent target models from corrupting the trajectory estimates of the tracking robot. We note the SKF has been previously investigated in the context of target tracking to address this issue or to avoid estimating navigation errors, however these works do not consider the visual-inertial domain or object pose tracking as in this work [12, 99, 127].

5.2 Tightly-Coupled Visual-Inertial Localization and Target Tracking

In this section, we propose a *tightly-coupled* estimator for visual-inertial localization and target tracking (VILTT) by building upon the computationally efficient MSCKF-based VINS framework and generalizing to incorporate 6DOF rigid-body target tracking of a 3D moving object. In particular, the main contributions of this section are the following [36]:

- We represent the target object as a rigid structure built from features and incorporate this representation into the MSCKF using different motion models. This representation allows for *robust* estimation of the target state even if the same feature is not continuously seen over a trajectory due to changing viewing angles or occlusions.
- We offer an extensive observability analysis of the system with the three proposed target motion models, and show that, besides the four unobservable directions inherited from VINS [58], there will be additional unobservable directions related to the target state, whose geometric interpretations are also provided.
- The proposed tightly-coupled VILTT estimator, with each of the three target motion models, is validated in Monte-Carlo simulations and real-world experiments.

5.2.1 Target State Representation

Leveraging the lightweight MSCKF framework, we now rigorously incorporate the tracking of an external 3D moving object into the same estimation thread. The first problem that needs to be addressed is *how* to represent the rigid-body target. That is, we need to define what parameters must be estimated to fully define the target and its motion.

One of the simplest representations is that of a point particle, which involves estimating a single position and its derivatives [21]. In reality, however, we often wish to track the motion of an arbitrarily large rigid *object*. Naive use of the point particle model requires picking a single point on the target to serve as a representation of the entire object, losing higher-level geometric understanding. For example, when target tracking using vision sensors, computer vision algorithms will possibly yield many features (corners) on the body of the target. A point particle model requires that we can only use one of these measurements, i.e., the one corresponding to the representative feature. If this feature becomes occluded, for example if the rigid body undergoes a large rotation, we can no longer measure the target, despite the fact that other features on the object can be still be observed. Therefore, we are motivated to instead represent a target as a set of structured points along with a pose.

In particular, we assume that the target evolves as a rigid body. That is, the relative positions of all points that reside on it, as expressed in a local body frame, remain fixed. We pick a representative point to serve as the origin of a local body frame (see Fig. 5.1). The pose of the target is then defined by the position of this representative point, ${}^{G}\mathbf{p}_{T}$, along with an orientation, ${}^{T}_{G}\bar{q}$. In practice this representative point will often correspond to the first seen target feature that can be reliably extracted over time.

5.2.2 Target Measurement Update

Similar to the standard static features, image measurements of points on the target's rigid body are collected. By abuse of notation, those measurements corresponding

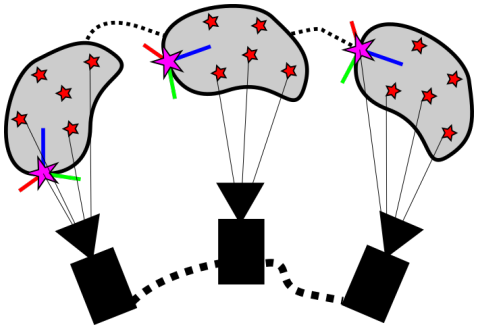


Figure 5.1: Illustration of the rigid body target tracking problem. As the sensor suite moves through the environment, bearing measurements to features on the evolving target's body, shown as stars, are collected and tracked through the sequence of images. To represent the pose of the target, a coordinate system is chosen and attached to a representative feature (pink star).

to the representative feature are written as:

$$\mathbf{z} = \mathbf{\Pi}({}^{C}\mathbf{p}_{T}) + \mathbf{n}_{f} \tag{5.1}$$

$${}^{C}\mathbf{p}_{T} = {}^{C}_{I}\mathbf{R}\,{}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T} - {}^{G}\mathbf{p}_{I}\right) + {}^{C}\mathbf{p}_{I} \tag{5.2}$$

This measurement function has the following Jacobians:

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{T}}{\partial {}^{G}\tilde{\mathbf{p}}_{T}} = {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} \tag{5.3}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{T}}{\partial {}^{G}\tilde{\mathbf{p}}_{I}} = -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} \tag{5.4}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{T}}{\partial {}^{I}_{G}\tilde{\boldsymbol{\theta}}} = {}^{C}_{I}\hat{\mathbf{R}}\lfloor {}^{I}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_{T} - {}^{G}\hat{\mathbf{p}}_{I}\right)\times\rfloor \tag{5.5}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{T}}{\partial {}^{C}_{I}\tilde{\boldsymbol{\theta}}} = \lfloor {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_{T} - {}^{G}\hat{\mathbf{p}}_{I}\right)\times\rfloor \tag{5.6}$$

$$\frac{\partial {}^{C_i}\tilde{\mathbf{p}}_{T}}{\partial {}^{C}\tilde{\mathbf{p}}_{I}} = \mathbf{I} \tag{5.7}$$

It is important to note that because the representative point's position is being estimated in this framework, this measurement can be used directly in the EKF update.

Since the number of tracked features that do *not* correspond to the representative point could be very large, we leverage the MSCKF's nullspace projection (i.e., linear marginalization [129]) to limit the state size. In particular, we concurrently maintain a sliding window of *target* poses $\mathbf{x}_{T_i} = [{}^{T_i}_{G}\bar{q}^{\top}\ {}^{G}\mathbf{p}_{T_i}^{\top}]^{\top}$, where for convenience

we let $\mathbf{T}_i$ denote the target pose clone associated with IMU clone i corresponding to the same imaging time. Non-representative feature measurements are given by:

$$\mathbf{z} = \mathbf{\Pi}({}^{C}\mathbf{p}_{ft}) + \mathbf{n}_f \tag{5.8}$$

$${}^{C}\mathbf{p}_{ft} = {}^{C}_{I}\mathbf{R}\ {}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_T + {}^{T}_{G}\mathbf{R}^{\top T}\mathbf{p}_{ft} - {}^{G}\mathbf{p}_I\right) + {}^{C}\mathbf{p}_I \tag{5.9}$$

which has the following Jacobians:

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{G}\tilde{\mathbf{p}}_T} = {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} \tag{5.10}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{G}\tilde{\mathbf{p}}_I} = -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}} \tag{5.11}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{I}_{G}\tilde{\boldsymbol{\theta}}} = {}^{C}_{I}\hat{\mathbf{R}}\lfloor {}^{I}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_T + {}^{G}_{T}\hat{\mathbf{R}}^{T}\hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_I\right)\times\rfloor \tag{5.12}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_T}{\partial {}^{T}_{G}\tilde{\boldsymbol{\theta}}} = -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}}\,{}^{G}_{T}\hat{\mathbf{R}}\lfloor {}^{T}\hat{\mathbf{p}}_{ft}\times\rfloor \tag{5.13}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{G}\tilde{\mathbf{p}}_{ft}} = {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}}\,{}^{G}_{T}\hat{\mathbf{R}} \tag{5.14}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_T}{\partial {}^{C}_{I}\tilde{\boldsymbol{\theta}}} = \lfloor {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_T + {}^{G}_{T}\hat{\mathbf{R}}^{T}\hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_I\right)\times\rfloor \tag{5.15}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_T}{\partial {}^{C}\tilde{\mathbf{p}}_I} = \mathbf{I} \tag{5.16}$$

In this case, the unknown feature state is the position of the feature expressed in the frame of the target, ${}^{T}\mathbf{p}_{ft}$. We then perform the same nullspace projection as the standard MSCKF with all tracks of this dynamic feature allowing for an efficient update that does not depend on the feature state. Alternatively, we can choose to add a small subset of the additional features into the state vector depending on the available resources, which we have found yields a substantial improvement in the accuracy of the target orientation estimate due to reobservations. In fact, we have found experimentally that while these MSCKF-like target features provide short-term orientation information, relying on them solely will yield large orientation drift over a long period of time, thus motivating us to keep a small, sparse set of features in the state.

As in the standard MSCKF, an initial estimate of the marginal parameter (non-representative feature) must be obtained in order to perform the nullspace projection,

as well as variable initialization. To this end, we use the following geometric constraint about the unknown non-representative feature position, ${}^{T}\mathbf{p}_{ft}$, (see Fig. 5.1):

$$ {}^{T}\mathbf{p}_{ft} = {}^{T_i}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{C_i} - {}^{G}\mathbf{p}_{T_i}\right) + d_i {}^{T_i}_{G}\mathbf{R}\,{}^{G}_{C_i}\mathbf{R}\,{}^{C_i}\mathbf{r}_i \tag{5.17} $$

$$ \Rightarrow \left\lfloor {}^{T_i}_{G}\mathbf{R}\,{}^{G}_{C_i}\mathbf{R}\,{}^{C_i}\mathbf{r}_i \right\rfloor {}^{T}\mathbf{p}_{ft} = \left\lfloor {}^{T_i}_{G}\mathbf{R}\,{}^{G}_{C_i}\mathbf{R}\,{}^{C_i}\mathbf{r}_i \right\rfloor {}^{T_i}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{C_i} - {}^{G}\mathbf{p}_{T_i}\right) \tag{5.18} $$

where d_i is the unknown depth of the feature in the i-th image, and ${}^{C_i}\mathbf{r}_i$ is the corresponding measured bearing, and ${}^{G}\mathbf{p}_{C_i}$ is the position of the measuring camera. Stacking all measurements (5.18) for the feature taken over the interval, we build a *linear system* that can be solved efficiently to obtain an estimate of the local feature position. This estimate is then refined by a local BA over the target feature utilizing the collected bearing measurements. Finally we utilize delayed initialization [76] to add these new points to our state (see Appendix E for details).

5.2.3 Target Stochastic Motion Models

To incorporate target tracking (a dynamic system) into the EKF framework, a motion model for propagation is needed. Unlike the tracking robot, we have *no* access to proprioceptive sensors for prediction of the target's state. As such, we assume a stochastic motion model and jointly estimate its parameters alongside the target's pose. In the following, we advocate three possible motion models that can capture a large class of realistic target tracking scenarios.

5.2.3.1 Model 1: Constant Global Linear Velocity

We first assume constant global linear velocity, ${}^{G}\mathbf{v}_T$, and constant angular velocity, ${}^{T}\boldsymbol{\omega}_T$, which yields the total target state and its dynamics as:

$$ \mathbf{x}_T^{(1)} = \begin{bmatrix} {}^{T}_{G}\bar{q}^{\top} & {}^{T}\boldsymbol{\omega}_T^{\top} & {}^{G}\mathbf{p}_T^{\top} & {}^{G}\mathbf{v}_T^{\top} \end{bmatrix}^{\top} $$

$$ {}^{T}_{G}\dot{\bar{q}} = \frac{1}{2}\boldsymbol{\Omega}\left({}^{T}\boldsymbol{\omega}_T\right){}^{T}_{G}\bar{q},\ {}^{G}\dot{\mathbf{p}}_T = {}^{G}\mathbf{v}_T,\ {}^{G}\dot{\mathbf{v}}_T = \mathbf{n}_{tv},\ {}^{T}\dot{\boldsymbol{\omega}} = \mathbf{n}_{t\omega} $$

In particular, we treat both the linear and angular velocities as continuous-time random walks driven by noises $\mathbf{n}_{tv}$ and $\mathbf{n}_{t\omega}$. The strength of the noise values can be used to

capture the predictability of the target based on its assumed motion model. This model is ideal for scenarios in which the evolution of the target's orientation and position are decoupled. For example, UAVs can move with full position and yaw control, and thus the orientation is not strictly coupled with position. The error state of this model evolves according to:

$$\begin{bmatrix} {}^{T}_{G}\dot{\tilde{\boldsymbol{\theta}}} \\ {}^{T}\dot{\tilde{\boldsymbol{\omega}}}_{T} \\ {}^{G}\dot{\tilde{\mathbf{p}}}_{T} \\ {}^{G}\dot{\tilde{\mathbf{v}}}_{T} \end{bmatrix} = \begin{bmatrix} -\lfloor {}^{T}\hat{\boldsymbol{\omega}}_{T} \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \begin{bmatrix} {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T}\tilde{\boldsymbol{\omega}}_{T} \\ {}^{G}\tilde{\mathbf{p}}_{T} \\ {}^{G}\tilde{\mathbf{v}}_{T} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \mathbf{n}_{t\omega} \\ \mathbf{n}_{tv} \end{bmatrix}$$

5.2.3.2 Model 2: Constant Local Linear Velocity

This model assumes that the target exhibits constant velocity as seen from its *local* frame. We therefore replace the global linear velocity in the previous model with local velocity ${}^{T}\mathbf{v}_{T}$. Such a model can be used for example, for tracking ground vehicles or fixed-wing aircrafts. The target state and its dynamics are given by:

$$\begin{aligned} \mathbf{x}_{T}^{(2)} &= \begin{bmatrix} {}^{T}_{G}\bar{q}^{\top} & {}^{T}\boldsymbol{\omega}_{T}^{\top} & {}^{G}\mathbf{p}_{T}^{\top} & {}^{T}\mathbf{v}_{T}^{\top} \end{bmatrix}^{\top} \\ {}^{T}_{G}\dot{\bar{q}} &= \frac{1}{2}\boldsymbol{\Omega}\left({}^{T}\boldsymbol{\omega}_{T}\right){}^{T}_{G}\bar{q},\ {}^{G}\dot{\mathbf{p}}_{T} = {}^{G}_{T}\mathbf{R}\,{}^{T}\mathbf{v}_{T},\ {}^{T}\dot{\mathbf{v}}_{T} = \mathbf{n}_{tv},\ {}^{T}\dot{\boldsymbol{\omega}}_{T} = \mathbf{n}_{tw} \end{aligned} \tag{5.19}$$

with the corresponding error state dynamics:

$$\begin{bmatrix} {}^{T}_{G}\dot{\tilde{\boldsymbol{\theta}}} \\ {}^{T}\dot{\tilde{\boldsymbol{\omega}}}_{T} \\ {}^{G}\dot{\tilde{\mathbf{p}}}_{T} \\ {}^{T}\dot{\tilde{\mathbf{v}}}_{T} \end{bmatrix} = \begin{bmatrix} -\lfloor {}^{T}\hat{\boldsymbol{\omega}}_{T} \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -{}^{G}_{T}\hat{\mathbf{R}}\lfloor {}^{T}\hat{\mathbf{v}}_{T} \rfloor & \mathbf{0}_3 & \mathbf{0}_3 & {}^{G}_{T}\hat{\mathbf{R}} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \end{bmatrix} \begin{bmatrix} {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T}\tilde{\boldsymbol{\omega}}_{T} \\ {}^{G}\tilde{\mathbf{p}}_{T} \\ {}^{T}\tilde{\mathbf{v}}_{T} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \mathbf{n}_{t\omega} \\ \mathbf{n}_{tv} \end{bmatrix}$$

5.2.3.3 Model 3: Local Planar Velocity

In many applications, the target is known to navigate in a locally planar environment (e.g., when tracking a ground vehicle). Rather than assuming pure 2D scenarios, we allow for *changing* ground planes, for example, when a vehicle goes up a ramp before coming to a new elevation (see Fig. 5.2). A method to handle this plane

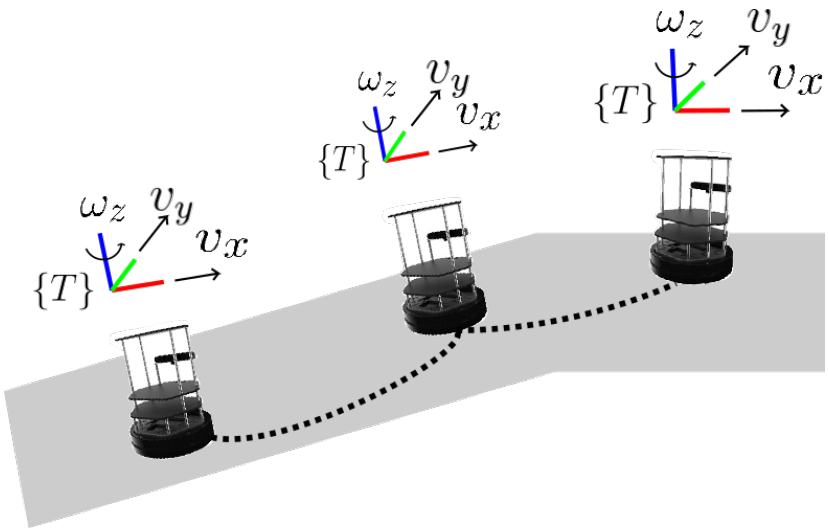


Figure 5.2: Illustration of the planar motion model. The target maintains a constant yaw rate and local planar velocity. The noise injected into the model can be used to handle changes in the ground plane.

change is to also estimate the current ground plane and add pseudo-measurements that the target should operate in this plane [126]. We forgo directly estimating the plane by proposing the following stochastic local planar model:

$$\mathbf{x}_T^{(3)} = \begin{bmatrix} {}^T_G\bar{q}^\top & \omega_z & {}^G\mathbf{p}_T^\top & v_x & v_y \end{bmatrix}^\top$$

$${}^T_G\dot{\bar{q}} = \frac{1}{2}\mathbf{\Omega}\left(\begin{bmatrix} n_{\omega x} \\ n_{\omega y} \\ \omega_z \end{bmatrix}\right){}^T_G\bar{q}, \quad {}^G\dot{\mathbf{p}}_T = {}^G_T\mathbf{R}\begin{bmatrix} v_x \\ v_y \\ n_{vz} \end{bmatrix}$$

$$\dot{v}_x = n_{vx}, \quad \dot{v}_y = n_{vy}, \quad \dot{\omega}_z = n_{\omega z}$$

In particular, along with the target pose we maintain the scalars v_x, v_y, and ω_z which describe the linear velocity in the local frame and the angular velocity about the z-axis (yaw). This motion model therefore constrains the evolution of the target state to act with constant local velocity in a plane whose normal lies along the axis of rotation. The noises in the other directions allow us to model such effects as uneven roads and the changing of the current ground plane (such as going up a hill). The error state

Table 5.1: Summary of unobservable subspace of the proposed VILTT with the three target motion models considered in this work. Note that the numbers in front of the unobservable directions indicate the corresponding dimensions.

Motion models	**If all measurements**	**If no measurements of representative point**
Model #1: Constant ${}^{G}\mathbf{v}_{T}$ Constant ${}^{T}\boldsymbol{\omega}_{T}$	1: Global yaw 3: Global IMU position 3: Target orientation ${}^{T}_{G}\mathbf{R}$ $\Rightarrow \dim(\textit{unobservable subspace}) = 7$	1: Global yaw 3: Global IMU sensor position 3: Target orientation ${}^{T}_{G}\mathbf{R}$ 1: Representative point position along rotation axis $\Rightarrow \dim(\textit{unobservable subspace}) = 8$
Model #2: Constant ${}^{T}\mathbf{v}_{T}$ Constant ${}^{T}\boldsymbol{\omega}_{T}$	1: Global yaw 3: Global IMU position 3: Target orientation ${}^{T}_{G}\mathbf{R}$ $\Rightarrow \dim(\textit{unobservable subspace}) = 7$	1: Global yaw 3: Global IMU position 3: Target orientation ${}^{T}_{G}\mathbf{R}$ 3: Representative point position $\Rightarrow \dim(\textit{unobservable subspace}) = 10$
Model #3: Planar motion Constant v_x, v_y Constant ω_z	1: Global yaw 3: Global IMU position 1: Target orientation yaw $\Rightarrow \dim(\textit{unobservable subspace}) = 5$	1: Global yaw 3: Global IMU position 1: Target orientation yaw 3: Representative point position $\Rightarrow \dim(\textit{unobservable subspace}) = 8$

evolves according to:

$$\begin{bmatrix} {}^{T}_{G}\dot{\tilde{\boldsymbol{\theta}}} \\ \dot{\tilde{\omega}}_z \\ {}^{G}\dot{\tilde{\mathbf{p}}}_{T} \\ \dot{\tilde{v}}_x \\ \dot{\tilde{v}}_y \end{bmatrix} = \begin{bmatrix} -\lfloor {}^{T}\hat{\boldsymbol{\omega}}_{T} \rfloor & \mathbf{e}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 2} \\ \mathbf{0}_3 & \mathbf{0}_{3\times 1} & \mathbf{0}_3 & \mathbf{0}_{3\times 2} \\ -{}^{G}_{T}\hat{\mathbf{R}}\lfloor {}^{T}\hat{\mathbf{v}}_{T} \rfloor & \mathbf{0}_{3\times 2} & \mathbf{0}_3 & {}^{G}_{T}\hat{\mathbf{R}}\begin{bmatrix}\mathbf{e}_1 & \mathbf{e}_2\end{bmatrix} \\ \mathbf{0}_{2\times 3} & \mathbf{0}_{2\times 1} & \mathbf{0}_{2\times 3} & \mathbf{0}_{2\times 2} \end{bmatrix} \begin{bmatrix} {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}_z \\ {}^{G}\tilde{\mathbf{p}}_{T} \\ \tilde{v}_x \\ \tilde{v}_y \end{bmatrix} + \begin{bmatrix} \mathbf{L} & \mathbf{0}_3 \\ \mathbf{e}_3^{\top} & \mathbf{0}_{1\times 3} \\ \mathbf{0}_3 & {}^{G}_{T}\hat{\mathbf{R}}\mathbf{K} \\ \mathbf{0}_{2\times 3} & \mathbf{J} \end{bmatrix} \begin{bmatrix} n_{\omega x} \\ n_{\omega y} \\ n_{\omega z} \\ n_{vx} \\ n_{vy} \\ n_{vz} \end{bmatrix} \tag{5.20}$$

where $\mathbf{L} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \mathbf{K} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{J} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, and $\mathbf{e}_i$ is the 3×1 unit vector in the i-th axis direction.

5.2.4 Observability Analysis

In this section, we perform an in-depth observability analysis for the linearized VILTT system with the three target motion models. The key reasons for observability analysis include: (i) it provides a deep insight about the system's geometrical

properties [61, 58, 90] and determines the minimum measurement modalities or state parameters needed to initialize the estimator, (ii) it can be used to identify degenerate motions [128, 126] which cause additional unobservable directions and should be avoided in real applications whenever possible, and (iii) the observability constrained (OC)-based methodology as in OC-EKF [61] and OC-VINS [58], that enforce the correct observability properties, can be adopted to improve consistency.

For simplicity, we consider the case where the state vector contains the IMU state $\mathbf{x}_I$, target state $\mathbf{x}_T$ (from Section 5.2.3), one static (environmental) feature ${}^G\mathbf{p}_f$ and one non-representative target feature ${}^T\mathbf{p}_{ft}$:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_I^\top & {}^G\mathbf{p}_f^\top & \mathbf{x}_T^\top & {}^T\mathbf{p}_{ft}^\top \end{bmatrix}^\top \tag{5.21}$$

In analogy to [58], we construct the observability matrix for the linearized VILTT system whose right nullspace spans the unobservable directions. Intuitively, these unobservable directions correspond to state variables that cannot be recovered from the measurement constraints. In the following, we present the main results of our observability analysis, while the detailed analysis can be found in Appendix G. These results are also summarized in Table 5.1.

5.2.4.1 Model 1

Given model 1 (constant ${}^G\mathbf{v}_T$ and constant ${}^T\boldsymbol{\omega}_T$), if all measurements to the static feature, target feature, and representative point are available, the VILTT system will have at least 7 unobservable directions corresponding to the global yaw, global IMU position ${}^G\mathbf{p}_I$, and the target orientation ${}^T_G\mathbf{R}$. Clearly, the first 4 unobservable directions are inherited from VINS [58]. Interestingly, if the measurements of the target's representative point are *unavailable* (e.g., due to occlusion), the system will have one more unobservable direction corresponding to the representative point position along the rotation axis of ${}^T\boldsymbol{\omega}_T$. In both cases, the unobservable directions related to the target state are the following:

$$\mathbf{N}^{(1)}_{{}^T_G\mathbf{R}} = \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{I}_3 & \left(\lfloor {}^T\hat{\boldsymbol{\omega}}_T \rfloor\right)^\top & \mathbf{0}_3 & \mathbf{0}_3 & \left(\lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor\right)^\top \end{bmatrix}^\top \tag{5.22}$$

$$\mathbf{N}^{(1)}_{{}^{G}\mathbf{p}_T} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \left({}^{G}_{T_0}\hat{\mathbf{R}}\,{}^{T}\hat{\boldsymbol{\omega}}_T\right)^\top & \mathbf{0}_{1\times 3} & \left(-{}^{T}\hat{\boldsymbol{\omega}}_T\right)^\top \end{bmatrix}^\top \tag{5.23}$$

5.2.4.2 Model 2

Given model 2 (constant ${}^{T}\mathbf{v}_T$ and constant ${}^{T}\boldsymbol{\omega}_T$), if all measurements available, the system will have at least 7 unobservable directions as in model 1. Similarly, if no representative-point measurements are available, the system will have 3 extra unobservable directions that correspond to the full 3D position of the representative point. In both cases, the unobservable directions related to the target state can be respectively found as follows:

$$\begin{aligned} \mathbf{N}^{(2)}_{{}^{T}_{G}\mathbf{R}} &= \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{I}_3 & \left(\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor\right)^\top & \mathbf{0}_3 & \left(\lfloor {}^{T}\hat{\mathbf{v}}_T \rfloor\right)^\top & \left(\lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor\right)^\top \end{bmatrix}^\top \\ \mathbf{N}^{(2)}_{{}^{G}\mathbf{p}_T} &= \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\right)^\top & \left(\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor\right)^\top & -\mathbf{I}_3 \end{bmatrix}^\top \end{aligned} \tag{5.24}$$

5.2.4.3 Model 3

Given model 3 (planar motion with constant ω_z, v_x, and v_y), if all measurements are available, unlike the above two models, the target's roll and pitch will become observable and thus the system has at least 5 unobservable directions, among which 4 are inherited from VINS and 1 corresponds to target orientation yaw. If no measurements of the representative point are available, as in the case of model 2, the system will also gain 3 extra unobservable directions corresponding to the full 3D position of the representative point. In both cases, the unobservable directions related to the target state are given by:

$$\mathbf{N}^{(3)}_{{}^{T}_{G}\mathbf{R}} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{e}_3^\top & \mathbf{0}_1 & \mathbf{0}_{1\times 3} & \hat{v}_y & -\hat{v}_x & \left(\lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{e}_3\right)^\top \end{bmatrix}^\top \tag{5.25}$$

$$\mathbf{N}^{(3)}_{{}^{G}\mathbf{p}_T} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_3\right)^\top & \mathbf{0}_1 & \mathbf{0}_1 & -\mathbf{e}_3^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_2\right)^\top & -\hat{\omega}_z & \mathbf{0}_1 & -\mathbf{e}_2^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_1\right)^\top & \mathbf{0}_1 & \hat{\omega}_z & -\mathbf{e}_1^\top \end{bmatrix}^\top$$

It can be seen from these results that the VILTT systems will have at least 4 unobservable directions inherited from VINS which correspond to the initial global

yaw and global IMU position (see [58, 128]), while the extra unobservable directions depend on the target motion model selected. Note that it is not trivial to find a best representative point that will be frequently measured, as a representative point that is occluded or cannot be tracked reliably will make the system suffer from the introduction of additional unobservable directions. On the other hand, an unobservable parameter can be initialized arbitrarily (unless some prior information is available). For example, in models 1 and 2, the initial target orientation can be freely chosen due to its unobservability, while in model 3, the orientation initialization procedure needs to be carefully addressed with available measurements.

5.2.5 Initializing the Target

We note that during the initial phase of the VILTT system, the robot has no information about the state of the target. Therefore the system must *initialize* these variables once the moving object begins to be observed. In our formulation, we first initialize the target as a point particle before transforming into a full pose. In particular, we consider the simplified target dynamics:

$$ {}^{G}\dot{\mathbf{p}}_T = {}^{G}\mathbf{v}_T \tag{5.26} $$

$$ {}^{G}\dot{\mathbf{v}}_T = \mathbf{n}_T \tag{5.27} $$

To achieve an initial estimate, we represent the target as a position and velocity at some initial time, t_0, where defining $\Delta t_k = t_k - t_0$:

$$ {}^{G}\mathbf{p}_{T_k} = {}^{G}\mathbf{p}_{T_0} + \Delta t_k {}^{G}\mathbf{v}_{T_0} \tag{5.28} $$

In addition, given the camera pose, $\{{}^{G}\mathbf{p}_{C_k}, {}^{C_k}_{G}\mathbf{R}\}$ and a bearing measurement, ${}^{C_k}\mathbf{r}_k$, from the camera center to the representative feature, we have the following constraint:

$$ {}^{G}\mathbf{p}_{T(t_k)} = {}^{G}\mathbf{p}_{C_k} + d_k {}^{G}_{C_k}\mathbf{R} {}^{C_k}\mathbf{r}_k \tag{5.29} $$

Where d_k depth of the feature in the corresponding image frame. As this quantity is unknown, we remove this dependency by choosing two vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ which are orthogonal to ${}^{G}_{C_k}\mathbf{R}{}^{C}\mathbf{r}_k$. Multiplying Eq. 5.29 by $\mathbf{A}_k = \begin{bmatrix}\mathbf{a}_1 & \mathbf{a}_2\end{bmatrix}^\top$, we have:

$$\mathbf{A}_k {}^{G}\mathbf{p}_{C_k} = \begin{bmatrix}\mathbf{A}_k & \mathbf{A}_k \Delta t_k\end{bmatrix} \begin{bmatrix}{}^{G}\mathbf{p}_{T_0} \\ {}^{G}\mathbf{v}_{T_0}\end{bmatrix} \tag{5.30}$$

Thus each measurement gives us two constraints about the unknown 6 degrees of freedom. Collecting all such constraints (we need at least three measurements) then allows us to solve for an estimate for the initial position and estimate, which can then be propagated to get an estimate at each time the target was measured.

While this gives us an initial guess for the motion parameters of the representative feature, we need to properly add this new variable into our covariance by considering all information involving the target, which involve both visual measurements and motion constraints. We note that due to the simplified state, we have the following target propagation across a viewing time interval $[t_k, t_{k+1}]$:

$$\begin{bmatrix}{}^{G}\mathbf{p}_{T_{k+1}} \\ {}^{G}\mathbf{v}_{T_{k+1}}\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \Delta t\mathbf{I} \\ \mathbf{0} & \mathbf{I}\end{bmatrix} \begin{bmatrix}{}^{G}\mathbf{p}_{T_k} \\ {}^{G}\mathbf{v}_{T_k}\end{bmatrix} + \begin{bmatrix}\mathbf{n}_{dp} \\ \mathbf{n}_{dv}\end{bmatrix} \tag{5.31}$$

$$\mathbf{x}_{T_{k+1}} = \mathbf{\Psi}_k \mathbf{x}_{T_k} + \mathbf{n}_k \tag{5.32}$$

where $\mathbf{n}_k = \begin{bmatrix}\mathbf{n}_{dp}^\top & \mathbf{n}_{dv}^\top\end{bmatrix}^\top$ is the discrete-time noise with covariance $\mathbf{Q}_k$. Note that this constraint acts as a measurement between the state at the beginning and end of the interval:

$$\mathbf{0}_6 = -\mathbf{x}_{T_{k+1}} + \mathbf{\Psi}_k \mathbf{x}_{T_k} + \mathbf{n}_k \tag{5.33}$$

We next perform a local bundle adjustment over the collected target measurements and the propagation constraints to formulate an estimate for the position and velocity of the target at each measurement time:

$$\hat{\mathbf{x}}_T = \arg\min_{\mathbf{x}_T} \sum_{k=0}^{N-1} \left|\left|\mathbf{\Psi}_k \mathbf{x}_{T_k} - \mathbf{x}_{T_{k+1}}\right|\right|^2_{\mathbf{Q}_k^{-1}} + \sum_i ||\mathbf{h}_i(\mathbf{x}_{T_i}) - \mathbf{z}_i||^2_{\mathbf{R}_i^{-1}} \tag{5.34}$$

where the first summation contains all motion constraints, while the second refers to all visual measurements collected over the interval (5.1) (note that we may have more than one measurement associated with each imaging time if multiple cameras are utilized). For computational reasons, as is typical for variable initialization, we fix the estimates for the rest of the state during this bundle adjustment. Therefore, the problem size is very small and does not require inversion of our dense covariance to formulate an information prior.

After obtaining a good initial estimate through this procedure, we then use both sets of measurements to initialize each velocity and position as in Appendix E, before marginalizing all but the current ones. To assign a notion of a full pose, we choose the final global orientation as an arbitrary value with $\mathbf{0}$ uncertainty for most of the representations (as this is arbitrary). For the planar model, we set up an initial plane whose x-axis aligns with the estimated velocity direction and with 0 roll, as this will correspond to most ground vehicle trajectories. To do so, we find the global representation of the local x-axis:

$$^{G}\mathbf{e}_{Lx} = \frac{^{G}\hat{\mathbf{v}}_{T}}{||^{G}\hat{\mathbf{v}}_{T}||_{2}} \tag{5.35}$$

We then find a local z-axis with no roll by performing orthonormalization with the direction of gravity (i.e., $^{G}\mathbf{e}_{3} = [0\ 0\ 1]^{\top}$).

$$^{G}\mathbf{e}_{Lz} = \frac{^{G}\mathbf{e}_{3} - {}^{G}\mathbf{e}_{Lx}\left({}^{G}\mathbf{e}_{Lx}^{\top}\,{}^{G}\mathbf{e}_{3}\right)}{||^{G}\mathbf{e}_{3} - {}^{G}\mathbf{e}_{Lx}\left({}^{G}\mathbf{e}_{Lx}^{\top}\,{}^{G}\mathbf{e}_{3}\right)||_{2}} \tag{5.36}$$

This gives a final rotation matrix as:

$${}^{T}_{G}\mathbf{R} = \begin{bmatrix} {}^{G}\mathbf{e}_{Lx} & \lfloor {}^{G}\mathbf{e}_{Lz}\times\rfloor\,{}^{G}\mathbf{e}_{Lx} & {}^{G}\mathbf{e}_{Lz} \end{bmatrix}^{\top} \tag{5.37}$$

We then give this initial uncertainty a non-zero value (except in the yaw) according to the previous analysis. For all models, we finally initialize the angular velocity as $\mathbf{0}$ with a large uncertainty. Lastly, a covariance propagation is performed to transform the target from the global pose/velocity representation into the final one.

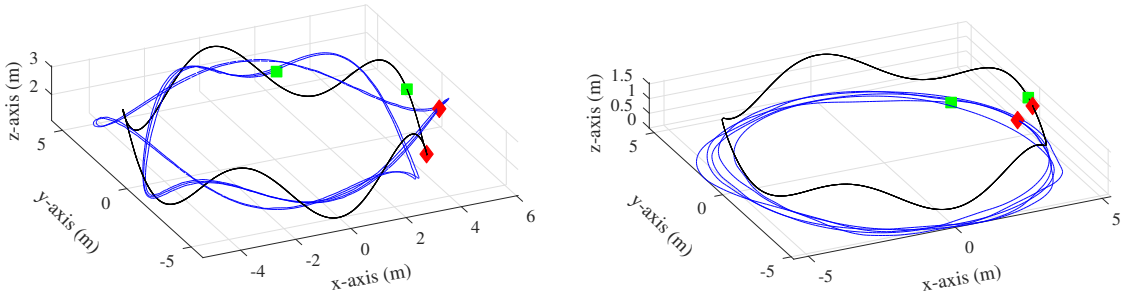


Figure 5.3: 3D simulation trajectories of the tracking robot (black) and target (blue): (a) general 3D target motion, and (b) constrained target motion. The total tracking robot's path length is 186 and 165 meters, respectively. The green square and red diamond denote the start and end of the tracking robot and target trajectories, respectively.

5.2.6 Simulation Results

In this work we used a stereo visual-inertial system, but note that the proposed method can also be deployed in a monocular setting. The RotorS simulator [45], which leverages Gazebo [71], was used to simulate an Asctec Firefly UAV equipped with a stereo visual-inertial sensor as the active tracking robot, while another simulated robot acted as the passive target. Two scenarios were created: (i) both the tracking robot and a target Firefly move with 3D motion, and (ii) a Turtlebot target is constrained to a 2D planar motion while the tracking robot moves through 3D space. The ground-truth IMU readings were corrupted using the realistic sensor characteristics of an ADIS16448 IMU, while image measurements were corrupted by one pixel noise. The rigid-body target was treated as a one meter box with features lying around the surface of the boundary, while static features were simulated around the workspace.

For both static and target feature measurements, occlusions were simulated by checking whether the projection vector intersected the boundary of the target's box, ensuring that blocked features were not used. Fourteen of the target features were maintained in the state vector to prevent orientation drift in the VILTT. We performed 30 Monte-Carlo simulations wherein a single ground truth target and IMU trajectory was collected for each scenario, and each Monte-Carlo run represented a different realization of noise corrupting the corresponding measurements (IMU and

bearing). The performance metrics used are the root mean squared errors (RMSE) of: (i) the 6DOF absolute (global) pose (position and orientation) estimates of both the tracking robot and target, and (ii) the relative position estimates between the tracking robot and the target. We note that the relative position error may be of more importance than the absolute error for certain scenarios such as autonomous target following.

We first evaluated the performance of the proposed VILTT with the motion models 1 and 2, where both the tracking robot and target moved along 3D trajectories as depicted in Fig. 5.3. The Monte-Carlo results are shown in Table 5.2. As evident, both systems were able to achieve high accuracy and recover the target's 6DOF motion (both position and orientation). In this experiment the orientation error for model 1 was much smaller than that for model 2, which is most likely due to the fact that the UAV had mostly decoupled orientation and position control. Interestingly both models had very similar performance in the position estimates for the target and the pose estimates of the IMU. Note that although the estimated target trajectory did *not* exhibit constant global or local velocity exactly, the proposed models were still able to handle these imperfections due to modeling the target's velocities as random walks.

In the second simulation, we validated the performance of the proposed VILTT with the three target models where the target exhibited constrained motion. The Monte-Carlo results of RMSE values are shown in Table 5.3. Clearly, all models were able to generate accurate trajectories for the target. We note that for these results the orientation error of model 3 (planar motion) cannot be directly comparable to that of the other two models, as the planar model is attempting to estimate a partially observable target orientation (only the yaw of target orientation is unobservable), while the other models estimate the change in orientation from the arbitrarily initial value. It is interesting to note that even though the target did not move with ideal constant velocity, as it was driven by hand, all models were able to handle this deviation from the assumed motion model. Surprisingly, for this scenario the planar model actually gave the worst result of the three models in terms of target position. This is most likely due

Table 5.2: Averaged RMSE results of the proposed VILTT in the case of general 3D target motion, showing both the absolute and relative accuracy of the realtime performance.

	Global Velocity		**Local Velocity**	
Units	m	deg	m	deg
IMU	0.309	2.669	0.302	2.571
Target	0.319	3.559	0.318	5.625
Relative	0.011	-	0.028	-

Table 5.3: Averaged RMSE results of the proposed VILTT in the case of constrained 3D target motion, showing both the absolute and relative accuracy of the realtime performance.

	Global Velocity		**Local Velocity**		**Local Planar**	
Units	m	deg	m	deg	m	deg
IMU	0.196	1.180	0.196	1.178	0.204	1.039
Target	0.207	1.616	0.207	1.610	0.214	7.567
Relative	0.010	-	0.010	-	0.015	-

to the fact that the imperfections in each model are being captured by the propagation noise whose characterization may greatly impact the accuracy of estimation. It is expected that better characterization of these noise levels would yield better results, which we investigate in Section 5.3. We note that in this work, the noises were chosen to attempt to capture the motions being executed by the simulations, rather than directly simulating target motion noise by drawing from a known distribution.

5.2.7 Experimental Results

In our real-world tests, to prove the concept of the proposed VILTT estimator, we simplified the target segmentation through the use of attached fiducial markers (see Fig. 5.4) around the target that can be reliably detected so that we can focus on the evaluation of the target estimation accuracy. Of course, more sophisticated target detection (e.g., based on deep learning) can be leveraged for a given application domain, as is done in Section 5.3. We calculated a bounding box of the target as the min/max image coordinates of the extracted fiducial markers, and updated the bounding box over a small sliding window to robustify it to failures in tag extraction.

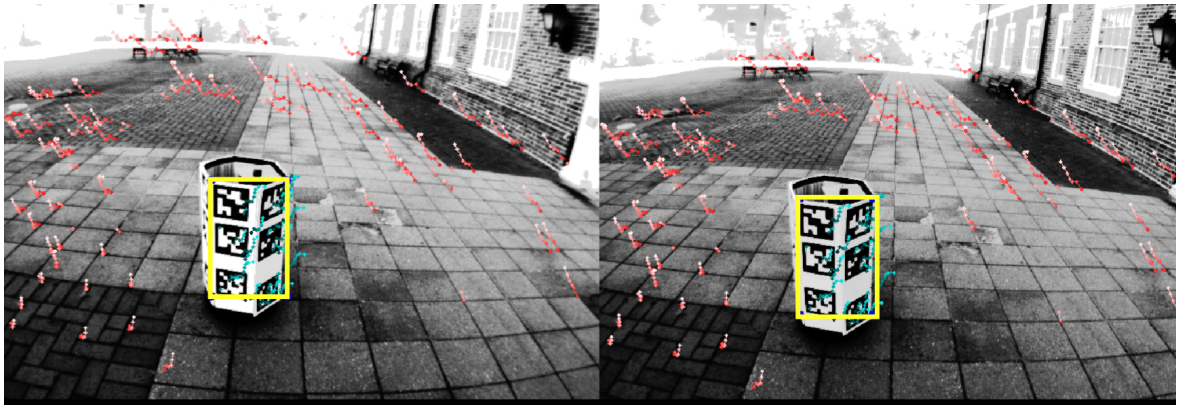


Figure 5.4: Example image taken during the second experiment. A Turtlebot equipped with fiducial tags acted as the target. These tags were used to distinguish features both on (light-blue) and off (red) the target by forming a bounding box (yellow).

This information was used to ensure that features were properly classified as either on or off the moving target. We found in this experimental setup that the target had to be within two or three meters to the camera in order to provide reliable tag extraction, otherwise, the system was not able to reliably detect the target. After extraction and computation of the target bounding box, we performed KLT tracking of sparse feature points [87]. Specifically, we initialized new features using FAST [109] feature extraction in a uniform grid pattern. KLT tracking was performed from both left-left and right-right camera images temporally, as well as left-right for stereo matches. We performed 8-point RANSAC on the static features and rejected measurements if they failed any of the three tracks. When features were lost or reached their maximum track length, they were processed using the MSCKF update step (either static or target-based). In this experiment, we maintained the features corresponding to the center and top left corner of each fiducial tag in the state vector. We should point out that while we used fiducial tag corners in our filter, we did *not* use any knowledge of the tag's size in our estimation, rather we relied on them *only* for target detection. Images were collected at a rate of 20 Hz using a VI-Sensor [98], while our system was able to process these images at a faster rate (on average 30 Hz) on an Intel i7-4700MQ CPU @ 2.40GHz. For comparison, the standard MSCKF takes 0.017 seconds per frame, while the proposed method takes an average of 0.031 seconds per frame (approximate 1.8× computational

increase).

We evaluated the proposed VILTT estimator on one of datasets that we collected outdoor on the University of Delaware campus, where an RTK-GPS was attached to the target robot (Turtlebot) to allow for groundtruth comparison and the target traveled on a semi-planar brick surface. The tracking robot (IMU/camera platform) was first initialized without the target in view, and after 31 seconds of motion, the target entered the view of the moving platform and was successfully initialized. Multiple loops around the target were made to showcase the ability to still localize the target without observation of the representative point. The tracking robot, target, and RTK-GPS paths are overlaid onto satellite imagery as shown in Fig. 5.5. The first 50 seconds of the target and RTK-GPS trajectories were used to compute a "best fit" transformation to align the two frame of references. For clarity, we present only the local velocity model in Figure 5.5, but all three models were able to successfully track the target. Clearly, the target position estimates closely follow the path of the RTK-GPS. We also utilized the fiducial tags to compute the error in the estimated relative position between the IMU and target during times when the representative tag is visible. Although we note that "ground-truth" relative positions from tag extraction might not be very accurate, we still found that our estimator yielded an RMSE of 0.044m.

In addition, we performed a second experiment wherein the target executed a mostly straight trajectory with small sinusoidal perturbations, followed by a sharp turn and continued motion. In this scenario, loss of sight of the target across a few seconds occurred due to actively facing the camera away from the target multiple times along the trajectory, with the maximum time of lost tracking being 7.4 seconds. While this sometimes led to large target updates upon reobserving the Turtlebot after a long period of pure propagation, the proposed VILTT was still able to perform *continuous* estimation of the target, with the resulting trajectory from using the local planar model shown in Fig. 5.5. However, we do note that during our experiments we found that viewing the target from a new angle (no previously seen target features are detected) upon reobservation after a period of tracking loss such that new target

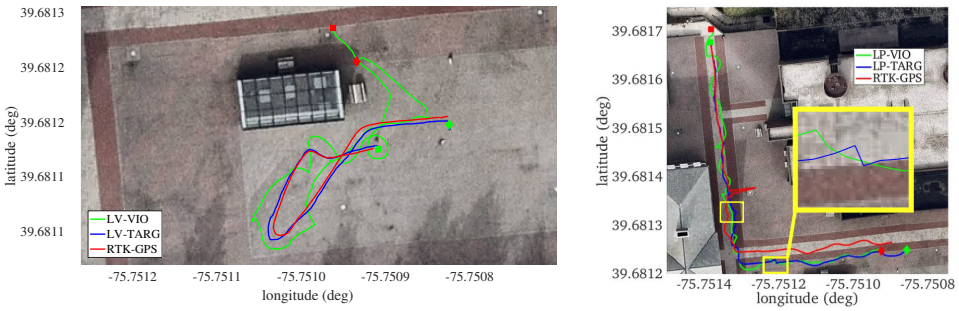


Figure 5.5: Top-down views of the trajectories generated by the proposed VILTT estimator in two real-world experiments. Note that only the RTK-GPS measurements (red) of the target were available. (a) The VIO (green) traveled a total distance of 85 meters, with the target (blue) moving 47 meters. The start and end positions are denoted by a square and diamond of opposite colors, respectively. (b) In this scenario, the jump in the RTK groundtruth is due to GPS multipath errors from nearby buildings. The proposed VILTT successfully tracks the target over its 81 meter long trajectory, despite losing sight multiple times (areas within the yellow boundary boxes). This re-observance typically causes a (possibly) large target correction (see blown-up box).

features are initialized with extremely poor estimates, could lead to divergence of the filter upon reobservation. If this behavior is detected (such as through Mahalanobis distance testing of incoming target measurements), we recommend that the target be reinitialized. For this trajectory, the RMSE for the relative position from the estimator referenced to the output value from tag extraction was 0.06m.

5.3 Robot-Centric, SKF-based Target Tracking

When performing SLAMMOT, a question remains open till date whether to use a tightly-coupled or loosely-coupled approach [122]. In the former, the ego-motion and object tracking is formulated as a single joint estimation problem [36], while in the latter the processes are performed separately, often conditioning the target tracking on the output of the localization module [83]. While this theoretically leads to decreased performance compared to tightly-coupled system, the separation often leads to substantial computational savings. In this section we investigate the effect that target motion modeling has on overall estimation performance. In particular, we show that while tightly-coupling the systems leads to improved accuracy of both processes (that is, the localization performance is improved when co-estimating the target rather than

simply discarding these measurements), this benefit is only seen if the target motion noise values are "properly" chosen. If they are overconfident, this will actually lead to a degrading of the VINS performance. This can be particularly catastrophic in cases where an autonomous vehicle performs the tracking and relies on the accuracy of its VINS to maintain operation.

Therefore, we propose to leverage Schmidt-Kalman Filtering (SKF) [112], an extension to the standard EKF, and employ a local (robot-centric) representation of the target, in particular, when there is low confidence in the chosen target model. This allows for the navigation estimates to be provably the same as if target measurements were discarded, however all correlations between the target and IMU are still properly modeled. A summary of this section's contributions are the following [35]:

- We investigate the choice of dynamic object motion noises in a joint visual-inertial localization and target tracking filter, and show through simulation that the selection of these values can lead to either improved or degraded accuracy of the VINS.
- We offer a new SKF formulation which does not allow target measurements to update navigation states while still consistently tracking all correlations, and show that this system can lead to robust estimation accuracy even with inconsistent model selection.
- We advocate for a robot-centric representation of the target's pose which is shown to offer improved performance for both the EKF and SKF formulations.
- The proposed system is validated in a real-world experiment where it is shown to offer accurate localization and dynamic object pose tracking estimation.

5.3.1 Local Target Estimation

In the previous section, while different motion models were utilized, each represented the target in the *global* frame of reference. By contrast, we now present the local formulation, where the target pose is represented relative to the IMU:

$$\mathbf{x}_T = \begin{bmatrix} {}^T_I\bar{q}^\top & {}^T\boldsymbol{\omega}_T^\top & {}^I\mathbf{p}_T^\top & {}^T\mathbf{v}_T^\top \end{bmatrix}^\top$$

As before, in order to properly add object features using delayed initialization, we also maintain a set of robot-centric target pose clones:

$$\mathbf{x}_{TC} = \begin{bmatrix} {}^{T_{k-1}}_{I_{k-1}}\bar{q}^\top & {}^{I_{k-1}}\mathbf{p}^\top_{T_{k-1}} & \cdots & {}^{T_{k-N+1}}_{I_{k-N+1}}\bar{q}^\top & {}^{I_{k-N+1}}\mathbf{p}^\top_{T_{k-N+1}} \end{bmatrix}^\top$$

Note that the target clones in the local formulation are expressed with respect to the corresponding IMU clone, rather than the evolving IMU state. While we use the term "robot-centric" for this system, only the target poses are represented in the robot frame, as compared to other robot-centric VIO systems which estimate the IMU/poses in the local frame [9, 60].

5.3.2 Robot-Centric Dynamic Model

In the local representation the evolution of the target is coupled with that of the IMU:

$$\begin{aligned} {}^T_I\dot{\mathbf{R}} &= {}^T_G\dot{\mathbf{R}}\,{}^G_I\mathbf{R} + {}^T_G\mathbf{R}\,{}^G_I\dot{\mathbf{R}} \\ &= -\lfloor {}^T\boldsymbol{\omega}_T\times\rfloor {}^T_G\mathbf{R}\,{}^G_I\mathbf{R} + {}^T_G\mathbf{R}\,{}^G_I\mathbf{R}\lfloor {}^I\boldsymbol{\omega}_I\times\rfloor \\ &= -\lfloor {}^T\boldsymbol{\omega}_T\times\rfloor {}^T_I\mathbf{R} + {}^T_I\mathbf{R}\lfloor {}^I\boldsymbol{\omega}_I\times\rfloor \qquad (5.38) \\ {}^I\mathbf{p}_T &= {}^I_G\mathbf{R}\left({}^G\mathbf{p}_T - {}^G\mathbf{p}_I\right) \\ \Rightarrow {}^I\dot{\mathbf{p}}_T &= -\lfloor {}^I\boldsymbol{\omega}_I\times\rfloor {}^I_G\mathbf{R}\left({}^G\mathbf{p}_T - {}^G\mathbf{p}_I\right) + {}^I_G\mathbf{R}\left({}^G\mathbf{v}_T - {}^G\mathbf{v}_I\right) \\ &= -\lfloor {}^I\boldsymbol{\omega}_I\times\rfloor {}^I\mathbf{p}_T + {}^I_T\mathbf{R}\,{}^T\mathbf{v}_T - {}^I_G\mathbf{R}\,{}^G\mathbf{v}_I \qquad (5.39) \end{aligned}$$

As in the global representation, we model the local linear and angular velocities as random walks:

$${}^T\dot{\mathbf{v}}_T = \mathbf{n}_{vt}\ ,\ {}^T\dot{\boldsymbol{\omega}}_T = \mathbf{n}_{\omega t} \qquad (5.40)$$

5.3.3 Error State Dynamics

We now compute the corresponding error state dynamics. Starting with orientation, we have:

$$\frac{d}{dt}\left(\left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right){}^T_I\hat{\mathbf{R}}\right) = -\lfloor {}^T\boldsymbol{\omega}_T\times\rfloor {}^T_I\mathbf{R} + {}^T_I\mathbf{R}\lfloor {}^I\boldsymbol{\omega}_I\times\rfloor \qquad (5.41)$$

First we expand the left-hand side (LHS) of (5.41):

$$\begin{aligned}\text{LHS} &= \frac{d}{dt}\left(\left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right){}^T_I\hat{\mathbf{R}}\right) \\ &= -\lfloor {}^T_I\dot{\tilde{\boldsymbol{\theta}}}\times\rfloor {}^T_I\hat{\mathbf{R}} + \left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right){}^T_I\dot{\hat{\mathbf{R}}} \\ &= -\lfloor {}^T_I\dot{\tilde{\boldsymbol{\theta}}}\times\rfloor {}^T_I\hat{\mathbf{R}} + \left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right)\left(-\lfloor {}^T\hat{\boldsymbol{\omega}}_T\times\rfloor {}^T_I\hat{\mathbf{R}} + {}^T_I\hat{\mathbf{R}}\lfloor {}^I\hat{\boldsymbol{\omega}}_I\times\rfloor\right)\end{aligned} \tag{5.42}$$

Expanding the right-hand side (RHS) of (5.41):

$$\begin{aligned}\text{RHS} &= -\lfloor {}^T\boldsymbol{\omega}_T\times\rfloor {}^T_I\mathbf{R} + {}^T_I\mathbf{R}\lfloor {}^I\boldsymbol{\omega}_I\times\rfloor \\ &= -\lfloor {}^T\hat{\boldsymbol{\omega}}_T + {}^T\tilde{\boldsymbol{\omega}}_T\times\rfloor\left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right){}^T_I\hat{\mathbf{R}} \\ &\quad + \left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right){}^T_I\hat{\mathbf{R}}\lfloor {}^I\hat{\boldsymbol{\omega}}_I + {}^I\tilde{\boldsymbol{\omega}}_I\times\rfloor\end{aligned} \tag{5.43}$$

Equating (5.42) and (5.43), we have:

$$\begin{aligned}-\lfloor {}^T_I\dot{\tilde{\boldsymbol{\theta}}}\times\rfloor {}^T_I\hat{\mathbf{R}} + \left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right)\left(-\lfloor {}^T\hat{\boldsymbol{\omega}}_T\times\rfloor {}^T_I\hat{\mathbf{R}}\right) &= -\lfloor {}^T\hat{\boldsymbol{\omega}}_T + {}^T\tilde{\boldsymbol{\omega}}_T\times\rfloor\left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right){}^T_I\hat{\mathbf{R}} \\ &\quad + \left(\mathbf{I} - \lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\right){}^T_I\hat{\mathbf{R}}\lfloor {}^I\tilde{\boldsymbol{\omega}}_I\times\rfloor\end{aligned}$$

We now solve for the evolution of the orientation error:

$$\begin{aligned}-\lfloor {}^T_I\dot{\tilde{\boldsymbol{\theta}}}\times\rfloor {}^T_I\hat{\mathbf{R}} &= -\lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor\lfloor {}^T\hat{\boldsymbol{\omega}}_T\times\rfloor {}^T_I\hat{\mathbf{R}} + \lfloor {}^T\hat{\boldsymbol{\omega}}_T\times\rfloor\lfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor {}^T_I\hat{\mathbf{R}} - \lfloor {}^T\tilde{\boldsymbol{\omega}}_T\times\rfloor {}^T_I\hat{\mathbf{R}} + \lfloor {}^T_I\hat{\mathbf{R}}\,{}^I\tilde{\boldsymbol{\omega}}_I\times\rfloor {}^T_I\hat{\mathbf{R}} \\ &= \lfloor\lfloor {}^T\hat{\boldsymbol{\omega}}_T\times\rfloor {}^T_I\tilde{\boldsymbol{\theta}}\times\rfloor - \lfloor {}^T\tilde{\boldsymbol{\omega}}_T\times\rfloor {}^T_I\hat{\mathbf{R}} + \lfloor {}^T_I\hat{\mathbf{R}}\,{}^I\tilde{\boldsymbol{\omega}}_I\times\rfloor {}^T_I\hat{\mathbf{R}}\end{aligned} \tag{5.44}$$

Noting that ${}^I\boldsymbol{\omega}_I = \boldsymbol{\omega}_m - \mathbf{b}_\omega - \mathbf{n}_\omega$, we have:

$${}^T_I\dot{\tilde{\boldsymbol{\theta}}} = -\lfloor {}^T\hat{\boldsymbol{\omega}}_T\times\rfloor {}^T_I\tilde{\boldsymbol{\theta}} + {}^T\tilde{\boldsymbol{\omega}}_T - {}^T_I\hat{\mathbf{R}}\left(-\tilde{\mathbf{b}}_\omega - \mathbf{n}_\omega\right) \tag{5.45}$$

Leaving us with the Jacobians:

$$\frac{\partial {}^T_I\dot{\tilde{\boldsymbol{\theta}}}}{\partial {}^T_I\tilde{\boldsymbol{\theta}}} = -\lfloor {}^T\hat{\boldsymbol{\omega}}_T\times\rfloor \tag{5.46}$$

$$\frac{\partial {}^T_I\dot{\tilde{\boldsymbol{\theta}}}}{\partial {}^T\tilde{\boldsymbol{\omega}}_T} = \mathbf{I} \tag{5.47}$$

$$\frac{\partial {}^T_I\dot{\tilde{\boldsymbol{\theta}}}}{\partial \tilde{\mathbf{b}}_\omega} = {}^T_I\hat{\mathbf{R}} \tag{5.48}$$

$$\frac{\partial_I^T \dot{\tilde{\boldsymbol{\theta}}}}{\partial \mathbf{n}_\omega} = {}_I^T\hat{\mathbf{R}} \tag{5.49}$$

For position we have:

$$\begin{aligned}
{}^I\dot{\hat{\mathbf{p}}}_T + {}^I\dot{\tilde{\mathbf{p}}}_T &= -\lfloor {}^I\boldsymbol{\omega}_I \times \rfloor {}^I\mathbf{p}_T + {}_T^I\mathbf{R}^T\mathbf{v}_T - {}_G^I\mathbf{R}^G\mathbf{v}_I \\
&= -\lfloor {}^I\hat{\boldsymbol{\omega}}_I + {}^I\tilde{\boldsymbol{\omega}}_I \times \rfloor \left({}^I\hat{\mathbf{p}}_T + {}^I\tilde{\mathbf{p}}_T\right) + {}_T^I\hat{\mathbf{R}}\left(\mathbf{I} + \lfloor {}_I^T\tilde{\boldsymbol{\theta}} \times \rfloor\right)\left({}^T\hat{\mathbf{v}}_T + {}^T\tilde{\mathbf{v}}_T\right) \\
&\quad - \left(\mathbf{I} - \lfloor {}_G^I\tilde{\boldsymbol{\theta}} \times \rfloor\right) {}_G^I\hat{\mathbf{R}}\left({}^G\hat{\mathbf{v}}_I + {}^G\tilde{\mathbf{v}}_I\right) \\
&\approx {}^I\dot{\hat{\mathbf{p}}}_T + \lfloor {}^I\hat{\mathbf{p}}_T \times \rfloor {}^I\tilde{\boldsymbol{\omega}}_I - \lfloor {}^I\hat{\boldsymbol{\omega}}_I \times \rfloor {}^I\tilde{\mathbf{p}}_T - {}_T^I\hat{\mathbf{R}}\lfloor {}^T\hat{\mathbf{v}}_T \times \rfloor {}_I^T\tilde{\boldsymbol{\theta}} \\
&\quad + {}_T^I\hat{\mathbf{R}}^T\tilde{\mathbf{v}}_T - \lfloor {}_G^I\hat{\mathbf{R}}{}^G\hat{\mathbf{v}}_I \times \rfloor {}_G^I\tilde{\boldsymbol{\theta}} - {}_G^I\hat{\mathbf{R}}{}^G\tilde{\mathbf{v}}_I
\end{aligned} \tag{5.50}$$

This yields the Jacobians:

$$\frac{\partial {}^I\dot{\tilde{\mathbf{p}}}_T}{\partial {}^I\tilde{\mathbf{p}}_T} = -\lfloor \boldsymbol{\omega}_m - \hat{\mathbf{b}}_\omega \times \rfloor \tag{5.51}$$

$$\frac{\partial {}^I\dot{\tilde{\mathbf{p}}}_T}{\partial {}^T\tilde{\mathbf{v}}_T} = {}_T^I\hat{\mathbf{R}} \tag{5.52}$$

$$\frac{\partial {}^I\dot{\tilde{\mathbf{p}}}_T}{\partial_I^T\tilde{\boldsymbol{\theta}}} = -{}_T^I\hat{\mathbf{R}}\lfloor {}^T\hat{\mathbf{v}}_T \times \rfloor \tag{5.53}$$

$$\frac{\partial {}^I\dot{\tilde{\mathbf{p}}}_T}{\partial {}^G\tilde{\mathbf{v}}_I} = -{}_G^I\hat{\mathbf{R}} \tag{5.54}$$

$$\frac{\partial {}^I\dot{\tilde{\mathbf{p}}}_T}{\partial_G^I\tilde{\boldsymbol{\theta}}} = -\lfloor {}_G^I\hat{\mathbf{R}}{}^G\hat{\mathbf{v}}_I \times \rfloor \tag{5.55}$$

$$\frac{\partial {}^I\dot{\tilde{\mathbf{p}}}_T}{\partial \tilde{\mathbf{b}}_\omega} = -\lfloor {}^I\hat{\mathbf{p}}_T \times \rfloor \tag{5.56}$$

$$\frac{\partial {}^I\dot{\tilde{\mathbf{p}}}_T}{\partial \mathbf{n}_\omega} = -\lfloor {}^I\hat{\mathbf{p}}_T \times \rfloor \tag{5.57}$$

5.3.3.1 Covariance Propagation

Based on the above tracking robot and target motion models, we can define the propagation of the error state of the stacked system. Let $\mathbf{x}_S$ denote the set of zero-dynamics static variables (whose true values do not evolve in time, such as static

environmental or target features, as well as possible fixed calibration parameters). The overall linearized error state evolution is then given by:

$$\begin{bmatrix} \dot{\tilde{\mathbf{x}}}_I \\ \dot{\tilde{\mathbf{x}}}_T \\ \dot{\tilde{\mathbf{x}}}_S \end{bmatrix} \approx \begin{bmatrix} \mathbf{F}_x & \mathbf{0} & \mathbf{0} \\ \mathbf{A}_x & \mathbf{A}_T & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}}_I \\ \tilde{\mathbf{x}}_T \\ \tilde{\mathbf{x}}_S \end{bmatrix} + \begin{bmatrix} \mathbf{G}_x & \mathbf{0} \\ \mathbf{\Gamma}_x & \mathbf{\Gamma}_T \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{n}_I \\ \mathbf{n}_T \end{bmatrix}$$

where $\mathbf{F}_x$ is the Jacobian of the IMU's error state evolution with respect to the IMU errors, $\mathbf{A}_x$ and $\mathbf{A}_T$ are the Jacobians of the target's error state evolution with respect to the IMU and target errors. Lastly $\mathbf{G}_x, \mathbf{\Gamma}_x$, and $\mathbf{\Gamma}_T$ are the Jacobians with respect to the IMU and target propagation noises ($\mathbf{n}_I$ and $\mathbf{n}_T$). From these continuous error-state dynamics, the standard EKF propagation can be performed [118].

5.3.4 Robot-Centric Target Update

The position of the feature in the camera frame that appears in the measurement equation using the local representation is given by:

$${}^{C}\mathbf{p}_{ft} = {}^{C}_{I}\mathbf{R}\left({}^{I}_{T}\mathbf{R}^{T}\mathbf{p}_{ft} + {}^{I}\mathbf{p}_{T}\right) + {}^{C}\mathbf{p}_{I} \tag{5.58}$$

Note that in the local formulation, the target measurements do not directly involve the IMU clones. The Jacobians are then given by:

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{T}_{I}\tilde{\boldsymbol{\theta}}} = -{}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{T}\hat{\mathbf{R}}\lfloor {}^{T}\hat{\mathbf{p}}_{ft}\times\rfloor \tag{5.59}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{C}_{I}\tilde{\boldsymbol{\theta}}} = \lfloor {}^{C}_{I}\hat{\mathbf{R}}\left({}^{I}_{T}\hat{\mathbf{R}}^{T}\hat{\mathbf{p}}_{ft} + {}^{I}\hat{\mathbf{p}}_{T}\right)\times\rfloor \tag{5.60}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} = {}^{C}_{I}\hat{\mathbf{R}}\,{}^{I}_{T}\hat{\mathbf{R}} \tag{5.61}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{I}\tilde{\mathbf{p}}_{T}} = {}^{C}_{I}\hat{\mathbf{R}} \tag{5.62}$$

$$\frac{\partial {}^{C}\tilde{\mathbf{p}}_{ft}}{\partial {}^{C}\tilde{\mathbf{p}}_{I}} = \mathbf{I} \tag{5.63}$$

5.3.5 Initialization of Robot-Centric Target Tracking

In our implementation, we utilize the same initialization procedure as in the global model. In particular, we first initialize the target as a global pose and velocity before performing a covariance propagation to bring it to the local form. This transformation takes the form:

$$ {}^{T}_{I}\mathbf{R} = {}^{T}_{G}\mathbf{R}\,{}^{G}_{I}\mathbf{R} \tag{5.64} $$

$$ {}^{T}\mathbf{v}_T = {}^{T}_{G}\mathbf{R}\,{}^{G}\mathbf{v}_T \tag{5.65} $$

$$ {}^{I}\mathbf{p}_T = {}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_T - {}^{G}\mathbf{p}_I\right) \tag{5.66} $$

The transformation Jacobians are given by:

$$ \frac{\partial^T_I \tilde{\boldsymbol{\theta}}}{\partial^T_G \tilde{\boldsymbol{\theta}}} = \mathbf{I} \tag{5.67} $$

$$ \frac{\partial^T_I \tilde{\boldsymbol{\theta}}}{\partial^I_G \tilde{\boldsymbol{\theta}}} = -{}^{T}_{I}\hat{\mathbf{R}} \tag{5.68} $$

$$ \frac{\partial^T \tilde{\mathbf{v}}_T}{\partial^T_G \tilde{\boldsymbol{\theta}}} = \lfloor {}^{T}_{G}\hat{\mathbf{R}}\,{}^{G}\hat{\mathbf{v}}_T \times \rfloor \tag{5.69} $$

$$ \frac{\partial^T \tilde{\mathbf{v}}_T}{\partial^G \tilde{\mathbf{v}}_T} = {}^{T}_{G}\hat{\mathbf{R}} \tag{5.70} $$

$$ \frac{\partial^I \tilde{\mathbf{p}}_T}{\partial^G \tilde{\mathbf{p}}_T} = {}^{I}_{G}\hat{\mathbf{R}} \tag{5.71} $$

$$ \frac{\partial^I \tilde{\mathbf{p}}_T}{\partial^G \tilde{\mathbf{p}}_I} = -{}^{I}_{G}\hat{\mathbf{R}} \tag{5.72} $$

$$ \frac{\partial^I \tilde{\mathbf{p}}_T}{\partial^I_G \tilde{\boldsymbol{\theta}}} = \lfloor {}^{I}\hat{\mathbf{p}}_T \times \rfloor \tag{5.73} $$

5.3.6 Numerical Analysis of Modeling Errors

While it has been shown that the co-estimation of target tracking and localization within a tightly-coupled filter can improve performance [91], we have found through simulations that this conclusion is more nuanced. At the root of the proposed framework is an assumption that the observed target will follow the selected motion model which we have in order to perform prediction of the target's future trajectory. In the case that the target model is not accurate, e.g., too small of noises are used

Table 5.4: Tracking robot and target average RMSE in meters / degrees of the global (G) and local (L) tightly-coupled estimator for different values of σ_t. We evaluate the target pose in the tracking robot's IMU frame ${}^{T}_{I}\mathbf{T}$, the target pose in the global frame, ${}^{T}_{G}\mathbf{T}$, and the IMU pose in the global frame ${}^{I}_{G}\mathbf{T}$.

σ_t	${}^{T}_{I}\mathbf{T}$ (G)	${}^{T}_{G}\mathbf{T}$ (G)	${}^{I}_{G}\mathbf{T}$ (G)	${}^{T}_{I}\mathbf{T}$ (L)	${}^{T}_{G}\mathbf{T}$ (L)	${}^{I}_{G}\mathbf{T}$ (L)
0.001	0.01172 / 1.069	2.149 / 12.84	2.030 / 12.95	0.01177 / 0.948	1.883 / 8.944	1.839 / 8.687
0.005	0.00720 / 0.545	0.277 / 1.519	0.271 / 1.678	0.00746 / 0.543	0.259 / 1.547	0.254 / 1.566
0.010	0.00697 / 0.499	0.241 / 1.525	0.236 / 1.690	0.00724 / 0.497	0.220 / 1.322	0.217 / 1.388
0.050	0.00646 / 0.319	0.211 / 1.478	0.207 / 1.504	0.00664 / 0.315	0.203 / 1.313	0.199 / 1.341
0.100	0.00657 / 0.339	0.211 / 1.500	0.207 / 1.502	0.00661 / 0.340	0.197 / 1.291	0.194 / 1.293
0.500	0.00835 / 0.687	0.284 / 1.719	0.279 / 1.779	0.00848 / 0.627	0.235 / 1.370	0.232 / 1.378

in (5.40), inconsistencies will be introduced as we are overconfident in how the target evolves and, in the worst case, negatively affect the accuracy of the tracking robot's trajectory. Conversely, if we pick too large of noise values, then the motion parameter estimation becomes unstable, and we do not gain much information to improve the tracking robot's estimate.

To demonstrate this, a numerical simulation scenario was created, where a UAV equipped with an IMU and a stereo camera follows a planar target robot which travels in a semi-circular pattern. A B-spline was fitted to the tracking robot's trajectory to allow for calculation of the groundtruth angular velocities and linear accelerations. From these, the noisy IMU measurements (and corresponding random-walk biases) were simulated at a rate of 200 Hz. A map of 3D points was simulated along the floor and borders of the workspace. Synthetic stereo images were generated at a frequency of 10 Hz by projecting the map points into the groundtruth camera poses and corrupted with one pixel noise. The target's point cloud was simulated by rigidly attaching a set of points lying on the surface of a one meter cube to the pose of the groundtruth target at each timestep. Both occlusions due to the target and randomly losing track of a given feature were simulated to model problems faced by real-world front-ends. As the target object operated on a plane, we modeled the linear velocity as a random walk only in the local x and y directions, while the angular velocity was a random walk along the local z-axis. We note that this type of target is very common in real-world scenarios, such as when tracking a terrestrial vehicle.

Table 5.5: Tracking robot and target average RMSE for the proposed SKF formulation. For very overconfident noise values, the global representation causes target estimate divergence.

σ_t	${}^{T}_{I}\mathbf{T}$ (G)	${}^{T}_{G}\mathbf{T}$ (G)	${}^{I}_{G}\mathbf{T}$ (G)	${}^{T}_{I}\mathbf{T}$ (L)	${}^{T}_{G}\mathbf{T}$ (L)	${}^{I}_{G}\mathbf{T}$ (L)
0.001	×	×	0.231 / 1.397	0.01092 / 0.879	0.234 / 1.720	0.231 / 1.397
0.005	×	×	0.231 / 1.397	0.00763 / 0.528	0.234 / 1.524	0.231 / 1.397
0.010	0.0408 / 0.555	0.231 / 1.472	0.231 / 1.397	0.00732 / 0.488	0.234 / 1.511	0.231 / 1.397
0.050	0.0410 / 0.411	0.229 / 1.393	0.231 / 1.397	0.00665 / 0.334	0.234 / 1.399	0.231 / 1.397
0.100	0.0432 / 0.430	0.229 / 1.390	0.231 / 1.397	0.00664 / 0.364	0.234 / 1.413	0.231 / 1.397
0.500	0.0623 / 0.744	0.233 / 1.633	0.231 / 1.397	0.00853 / 0.639	0.234 / 1.617	0.231 / 1.397

To evaluate the effect of different noise parameters on estimation, we chose a single noise standard deviation $\sigma_t = \sigma_{wz} = \sigma_{vx} = \sigma_{vy}$ to drive the *assumed* random walks, see Eq. (5.40), and performed a parameter sweep over a series of values. Note that for all trials we utilized the same groundtruth IMU and target trajectories, and only varied the noise levels utilized by the filter during target propagation. While a more sophisticated sweep may be performed that varies the noise in all directions, we found that a single parameter was sufficient to demonstrate the discussed issues. For each assumed noise strength, 30 Monte Carlo runs were simulated using the same groundtruth trajectory while each run had a different realization for the IMU and pixel measurement noises. The average Root Mean Squared Error (RMSE) values shown in Table 5.4 clearly illustrate the issues caused by inconsistencies when assuming incorrect model noises. The standard VIO without estimating the external target is able to achieve an average RMSE of 0.231 meters and 1.397 degrees for a trajectory of 165 meters. While localization performance improves when using "proper" noise values, the overconfidence introduced by choosing target propagation noises that are too small corrupts the overall system. In addition, we found that having too large of noises may also harm VIO accuracy. This is most likely because at these levels the motion parameter estimation becomes very unstable leading to poor target predictions that may cause large linearization errors.

Looking at a typical single run and its error bounds, as shown in Figure 5.6, tightly-coupling of the target tracking reduces the uncertainty compared to stand-alone VIO while remaining consistent (that is, the errors remain in the bounds provided by the covariance) in the case of "good" noise parameters. When the chosen noise is too

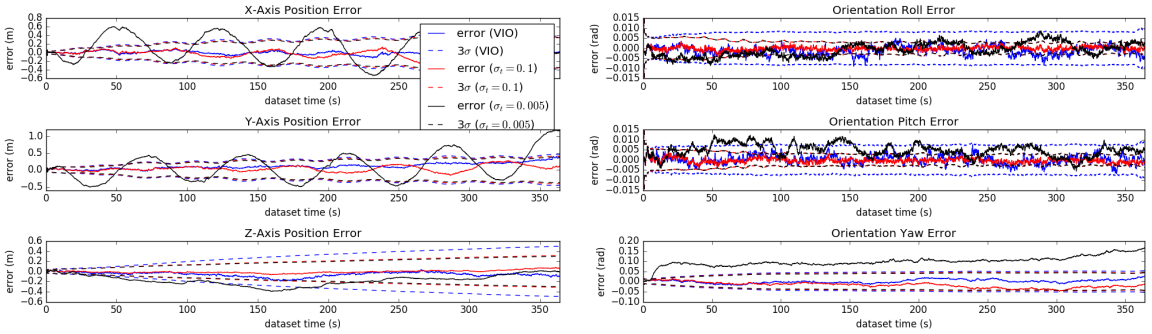


Figure 5.6: Estimation errors of IMU global pose when using VIO by itself, tightly-coupled tracking with a "good" target $\sigma_t = 0.1$, as well as an overconfident $\sigma_t = 0.005$. In the tightly-coupled system with "proper" noises, the error bounds are decreased and the estimate remains consistent, showing that the fusion of target information has improved localization performance. For an overconfident noise, the estimator becomes inconsistent.

small, the resulting trajectory error is inconsistent, thereby showing the corruption caused by incorrect modeling of the target motion. Thus, the effectiveness of tightly-coupling the estimation heavily depends on the choice of motion noise parameters. This has profound impact on the application of tightly-coupled target estimation algorithms in the real-world as the choice of correct noise values is difficult and can leave the system vulnerable to inaccuracies and in the worse case may even cause filter divergence. This motivates the proposed Schmidt-EKF formulation that can allow for proper modeling of the uncertainties of the joint system without the inconsistencies of the target model affecting the tracking robot's trajectory accuracy.

5.3.7 Schmidt EKF Update

In the case that we want to prevent corruption of the tracking robot's estimates, we leverage the SKF [112] to update the target states; that is, when processing measurements to the target, we "schmidt" the inertial state (and corresponding clones) and thus only update the target estimates and their correlation with the tracking robot's states. More explicitly, we partition the state into two, the tracking robot's states $\mathbf{x}_R$ and target parameters, $\mathbf{x}_T$, such that $\mathbf{x} = [\mathbf{x}_R^\top \ \mathbf{x}_T^\top]^\top$, and decompose the covariance as: $\mathbf{P} = \begin{bmatrix} \mathbf{P}_{RR} & \mathbf{P}_{RT} \\ \mathbf{P}_{TR} & \mathbf{P}_{TT} \end{bmatrix}$. Now, let us consider a measurement residual $\mathbf{r}$ that is being used to update the state. The SKF does not update the tracking robot's states by setting

their Kalman gain to zero (i.e. $\mathbf{K} = [\mathbf{K}_R^\top \ \mathbf{K}_T^\top]^\top \rightarrow [\mathbf{0}^\top \ \mathbf{K}_T^\top]^\top$). The update equations become the following:

$$\hat{\mathbf{x}}^\oplus = \hat{\mathbf{x}}^\ominus \boxplus \begin{bmatrix} \mathbf{0} \\ \mathbf{K}_T \end{bmatrix} \mathbf{r} = \hat{\mathbf{x}}^\ominus \boxplus \begin{bmatrix} \mathbf{0} \\ \mathbf{L}_T \end{bmatrix} \mathbf{S}^{-1}\mathbf{r} \tag{5.74}$$

$$\mathbf{P}^\oplus = \mathbf{P}^\ominus - \begin{bmatrix} \mathbf{0} & \mathbf{L}_R\mathbf{S}^{-1}\mathbf{L}_T^\top \\ \mathbf{L}_T\mathbf{S}^{-1}\mathbf{L}_R^\top & \mathbf{L}_T\mathbf{S}^{-1}\mathbf{L}_T^\top \end{bmatrix} \tag{5.75}$$

where $\begin{bmatrix} \mathbf{L}_R^\top & \mathbf{L}_T^\top \end{bmatrix}^\top = \mathbf{P}^\ominus\mathbf{H}^\top$. Clearly, the tracking robot states $\mathbf{x}_R$ and marginal covariance $\mathbf{P}_{RR}$ do not change in this update, preventing the corruption of these estimates in the case that the target model is inconsistent. Therefore the resulting VIO estimates are equivalent to if the target measurements had been discarded. It is important to note that the SKF still tracks correlations in the covariance which can be shown to be a conservative approximation of the EKF update [26]. We also highlight that the SKF is only used when updating using target measurements. During processing of the visual data corresponding to the static scene, the full state is updated normally. This allows corrections to the IMU to correct the target estimates, which is only possible because of the consistently tracked correlations.

Due to the fact that the SKF does not update a portion of the state, the representation of the estimated variables becomes crucial to the performance [12]. In the global model, the filter is free to fully update the global pose of the target, while in the local model, it fully updates the relative pose between the IMU and target. Intuitively, as target measurements are relative to the sensing platform, it makes more sense to fully update the relative relationship between the two. To investigate the effect of this robot-centric representation, we reran the noise sweep of the previous section using the local formulation, as shown in Table 5.4. These results illustrate that for the tightly-coupled system the robot-centric filter outperforms its global counterpart in terms of both target and VIO accuracy. This is most likely due to the fact that linearization errors tend to be much smaller in the local frame. However, this system still displayed the same problems when using a poor motion model.

We tested the proposed SKF formulation with both the global and local representations (Table 5.5). As expected, the systems cannot benefit from the tightly-coupled estimation for the "proper" noise values, while the VIO cannot be corrupted even when the target model is inconsistent. In addition, we experimentally found that the global target pose filter was more prone to target estimate divergence at the lower noise levels (typically following large target updates), while the robot-centric method was always able to provide accurate tracking performance, even when using an inconsistent motion model. This motivates our choice of a local representation for both the EKF and SKF formulations, and shows that our local SKF filter offers robustness to poor target models.

5.3.8 Extension to Real-World Tracking

We next evaluated the proposed local SKF-based localization and object tracking in a real-world scenario using a stereo visual-inertial rig. In what follows we first discuss the visual feature front-end needed to both segment the target and then track features on that target over its trajectory. From a high level, we performed segmentation on incoming images using a variation of UNet [108] that generates a mask of the target for each image. Using this mask, we then performed separate visual tracking for environmental and target features which can then be fused in the proposed estimator.

5.3.8.1 Deep Learning-based Target Detection

Due to the absence of an annotated visual-inertial target tracking dataset with groundtruth trajectories for the robot and target, we opted to collect and annotate our own dataset tracking a large remote controlled car. We collected a training, validation and testing dataset from the left camera at 1Hz, resulting in 279 images for training and validation (with 10 percent randomly chosen for validation), and 85 for testing. These images were labeled by hand to create the groundtruth masks. The camera used to collect the images had a fisheye lense, but the masks were generated without undistoring the images to avoid having to redistort later on. However, due to this, we

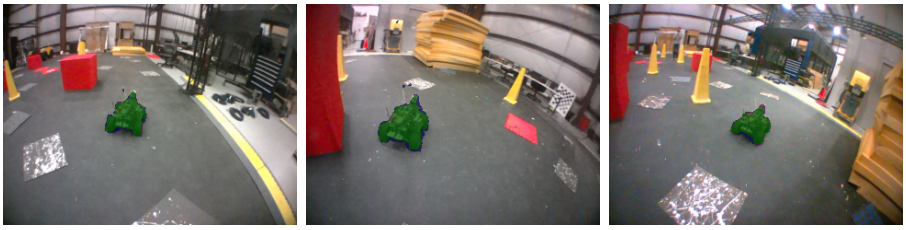


Figure 5.7: Visual results of the network on the testing dataset. True positive pixels have been marked green, false positives blue, and false negatives pink.

only used random left-right flipping to augment the data and not rotations and random cropping.

A variant of the popular UNet architecture was used to segment the target. In our test, we used sub-pixel convolution [115] in place of the original upsample layer proposed for efficiency. Additionally, we replaced the ReLU activation with ELU [23] to avoid the need for batch normalization – again, for efficiency. We also used zero padding to avoid the need for cropping the features used in skip connections. The network can predict on average in less than 20 milliseconds on a GTX 1080Ti graphics card using a 320×240 resolution. To test the network, we used the mean intersection over union (mIOU) and pixel-wise accuracy metrics. Pixel-wise accuracy in our case is defined by $\frac{TP+TN}{TP+TN+FP+FN}$, where TP, TN, FP, and FN are the true positive, true negative, false positive, and false negative predictions, respectively. The network achieved an mIOU of 0.628 and a pixel-wise accuracy of 99.1% on the testing dataset (see Fig. 5.7).

Since the target typically occupies a small portion of the overall pixels, we opted to extract target features from the bounding box of the segmentation mask instead of the entire image, while environmental features were extracted from the full image. To do this, we first removed the majority of false positives from the mask through a series of erosion and dilation operations. The bounding box was then taken as the rectangle about the blob with the largest area. A separate Kalman filter was used to track the bounding box, thereby smoothing the noisy masks.

5.3.8.2 Visual Feature Tracking

We utilized FAST detection [109] across multiple grids of the image to ensure that we achieved a uniform distribution of features. Only features that fell within the mask were labeled as target features. We performed KLT tracking [87] through the implementation available in OpenCV [11] for environmental features, and ORB descriptor tracking [111] for target features. This tracking was done both from the left-to-right images at the same timestamp as well as left-to-left and right-to-right tracking from the previous images. Outliers were rejected using 8-point RANSAC which was performed independently for the static and target features.

Target features that were tracked in this way for a certain number of frames were initialized as permanent object features. In addition, we performed ORB descriptor matching to the currently estimated target point cloud in order to determine whether new tracks corresponded to previously seen features. We have found that these target loop closing events are vital to the performance of the estimator as they greatly limit the target's drift relative to the IMU (and therefore globally). As mismatched features are common in real-world experiments, we utilized Mahalanobis distance tests to reject inconsistent measurements.

5.3.8.3 Experimental Results

In this experiment, a hand-held visual-inertial rig tracked a remote-controlled car. The network was trained to detect the vehicle using a separately collected dataset in the manner described previously. Note that different from the state of art [104], we do not require a high-resolution camera (which could certainly lead to higher accuracy). We allowed for a maximum of 150 features to be tracked from the point cloud. To limit visual-inertial drift, we added up to 10 SLAM features which were generated from visual tracks that reached the length of the window size (10 images in this test) [76], and were marginalized whenever they were lost. We also performed online estimation of the IMU-camera extrinsincs to handle the possibly poor prior calibration. As this was a (mostly) planar scenario, we chose the noises driving the local z-axis linear velocity

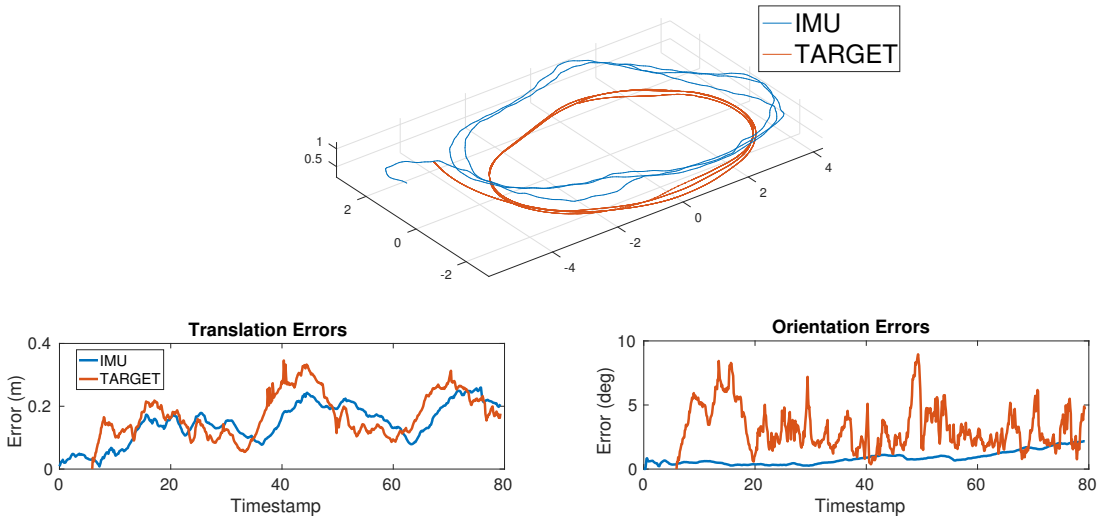


Figure 5.8: True trajectory and estimation errors of IMU and target global poses for the proposed SKF system. The total IMU and target trajectories were 62 meters and 52 meters, respectively.

and the pitch/roll angular velocities to be zero, as in the simulated example. For this experiment, images were collected at 10 Hz and the entire system ran in real time on a commercial laptop.

In order to evaluate the accuracy, the tracking scenario was performed in a large Vicon room, thus providing highly accurate pose estimates of both the sensor platform and the target. A purposefully overconfident target motion model was selected to illustrate the proposed method's robustness. Over 30 Monte Carlo runs, the SKF based target tracking had a VIO RMSE of 0.153 m / 1.091 degrees, while the global target errors were 0.183 m / 3.443 degrees. The error-vs-time plots for an example run are shown in Fig 5.8. By contrast, the tightly-coupled system achieved 0.409 m / 1.640 degrees for the IMU and 0.492 m / 3.037 degrees for the target, thus showing the degradation of the VIO due to the poor motion model. These results indicate that the proposed system is able to accurately estimate both the target and navigation states even when using an overconfident target model. In addition, the mask provided by the network failed for several sequential frames multiple times throughout the test, which our system was able to handle due to the estimation of the target motion parameters.

5.3.9 Conclusion

In this chapter we have proposed a tightly-coupled visual-inertial localization and target tracking system which is able to estimate the poses of both a moving sensor platform and an external rigid-body target. We offered three motion models which capture the behavior of many targets seen in practice, along with a comprehensive observability analysis using each model. The proposed system was then validated in both simulation and real-world experiments.

We next investigated the choice of target propagation *noises*, and showed that given "correct" noise levels, tightly-coupling the system leads to an improvement of the overall system, while in the case of inconsistently chosen noises the ego-motion estimation can be greatly corrupted. To handle this scenario, we have introduced a method based on the Schmidt-Kalman Filter for preventing incorrect motion models from corrupting the VINS, and have shown in both simulation and real-world experiments that this, along with a robot-centric representation of the target, leads to robust performance.

Chapter 6

CONCLUSION

6.1 Summary of Content

In this thesis, we have sought to design robust, efficient, and accurate visual-inertial estimation algorithms through three main thrusts. In the first thrust, we developed closed-form IMU preintegration (CPI) for improving graph-based VINS and VI-SLAM through more advanced modeling of the IMU dynamics between sample intervals. In particular, we offered two models for the evolution of these measurements which yield more accurate, closed-form expressions for the preintegration equations than those of the standard method. We successfully validated the CPI by incorporating its solutions into both direct and indirect VINS systems, showing competing accuracy to state-of-the-art methods.

In the second thrust, we developed an easy-to-use multi-IMU multi-camera (MiMc)-VINS which allows for an arbitrary number of asynchronous IMUs and (rolling shutter) cameras to be leveraged. This is enabled by the rigorous design of an estimator which online refines the estimates of the spatial and temporal relationships between all sensors as well as the intrinsics of each camera, in order to compensate for poor prior or time-varying calibration. As a result, our MiMC-VINS significantly improves estimation accuracy and robustness to sensor failure or measurement depletion while maintaining efficiency.

In the third thrust, we extended VINS from static environments to dynamic ones by incorporating the estimation of external rigid-body motion. In particular, we assigned motion models to moving objects, and estimated their motion parameters, pose, and local point-cloud. We showed that while proper choices of motion models

leads to overall improved accuracy of state estimation, poorly chosen ones can cause the estimates to degrade or even diverge. To handle these realistic scenarios where the properties of the motion model are not known, we offered a robust estimator based on the Schmidt-Kalman Filter which does not allow target measurements to update the VINS, while still properly tracking all correlations. We then showed that this approach, along with a robot-centric representation of the target, leads to robust estimation performance even in cases where the target model is poorly chosen.

6.2 Perspectives on Future Work

6.2.1 Multi-Camera Multi-IMU VINS

In this thesis we have focused on independent image streams for speed, however, it is known that cross-image matching leads to improved performance [102]. Thus a challenge remains on how to incorporate these matches into the proposed approach approach while remaining real time. In addition, more advanced interpolation schemes may prove even more effective than the currently presented approach [44], but may lead to increased computation.

While the proposed multi-IMU system has been greatly validated in this thesis, as noted in [130], adding navigation states increases the computational cost which may become infeasible for large numbers of IMUs. Thus reducing the state size while still allowing for online calibration remains challenging. In addition, incorporating online *initialization* of the calibration parameters as in [70] may lead to systems that are truly "plug-and-play", in the sense that the user does not even need to provide an initial estimate/uncertainty for these values. While we have also addressed the issue of recovering from sensor failures, research on reliably *detecting* these failures still remains open. Finally, the proposed MiMc may be modified to fit into a batch estimator such as those presented in Chapter 3, thereby leading to further improvements in accuracy and robustness at the cost of increased computation.

6.2.2 Target Tracking and Perception

While we have investigated how to robustify a VILTT system against improper model selection, we did not offer solutions on how to pick this model for a given application. Ideally, the system would be able to decide *online* what model/noise magnitudes to utilize, which may be addressed by leveraging deep-learning techniques. In addition, deciding online whether to use SKF or EKF updates could allow the system to automatically benefit from target measurements if these would help its estimate, while protecting itself when they become inconsistent, rather than deciding *a priori*. Scaling the proposed joint-estimation system to handle a large number of targets is also desirable for practical applications but remains challenging due to issues in computation and being able to reliably discern distinct objects. Lastly, we may not be just interested in how an external body might move, but also what actions it might perform (will this tracked target manipulate the environment in some way?). This level of perception is vital for allowing robots to robustly perform actions in real world scenarios.

6.2.3 Extensions to Cooperative VILTT

In this thesis we have focused on incorporating an arbitrary number of sensors mounted on a single agent for improved robustness. However, a promising direction of research lies in *cooperative* VINS [52]. That is, a future goal is the development of systems where multiple users can share data for improved accuracy across the entire fleet. This would have great application for cooperative VILTT, as having spatially diverse agents viewing the object from multiple angles would greatly improve tracking performance and robustify the system to loss of sight.

However, this scenario represents a generalization of the multi-sensor calibration problem tackled in Chapter 4, as we need the temporal relationship between *every* sensor in the fleet in order to perform the fusion. This is because each *user* might have its own base clock that it represents all measurements in. Thus properly fusing the information from multiple agents operating in a dynamic environment requires calibrating this offset, which remains a challenging problem.

Bibliography

[1] Sameer Agarwal, Keir Mierle, et al. *Ceres Solver*. http://ceres-solver.org.

[2] Aamir Ahmad et al. "Cooperative robot localization and target tracking based on least squares minimization". In: *International Conference on Robotics and Automation*. May 2013, pp. 5696–5701.

[3] Aitor Aldoma et al. "Multimodal cue integration through hypotheses verification for rgb-d object recognition and 6dof pose estimation". In: *2013 IEEE international conference on robotics and automation*. IEEE. 2013, pp. 2104–2111.

[4] Remus C Avram et al. "IMU sensor fault diagnosis and estimation for quadrotor UAVs". In: *IFAC-PapersOnLine* 48.21 (2015), pp. 380–385.

[5] Asma Azim and Olivier Aycard. "Detection, classification and tracking of moving objects in a 3D environment". In: *2012 IEEE Intelligent Vehicles Symposium*. IEEE. 2012, pp. 802–807.

[6] S. Baker and I. Matthews. "Lucas-Kanade 20 Years On: A Unifying Framework". In: *International Journal of Computer Vision* 56 (Mar. 2004), pp. 221–255.

[7] Jared B Bancroft. "Multiple IMU integration for vehicular navigation". In: *Proceedings of ION GNSS*. Vol. 1. 2009, pp. 1–13.

[8] Jared B Bancroft and Gérard Lachapelle. "Data fusion algorithms for multiple inertial measurement units". In: *Sensors* 11.7 (2011), pp. 6771–6798.

[9] Michael Bloesch et al. “Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback”. In: *The International Journal of Robotics Research* 36.10 (2017), pp. 1053–1072.

[10] Michael Bloesch et al. “Robust visual inertial odometry using a direct EKF-based approach”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 298–304.

[11] G. Bradski. “The OpenCV Library”. In: *Dr. Dobb’s Journal of Software Tools* (2000).

[12] Edmund F Brekke and Erik F Wilthil. “Suboptimal Kalman filters for target tracking with navigation uncertainty in one dimension”. In: *2017 IEEE Aerospace Conference*. IEEE. 2017, pp. 1–11.

[13] Martin Brossard, Silvere Bonnabel, and Jean-Philippe Condomines. “Unscented Kalman filtering on Lie groups”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 2485–2491.

[14] Michael Burri et al. “The EuRoC micro aerial vehicle datasets”. In: *The International Journal of Robotics Research* 35.10 (2016), pp. 1157–1163.

[15] Cesar Cadena et al. “Past, present, and future of simultaneous localization and mapping: toward the robust-perception age”. In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1309–1332.

[16] David Caruso, Jakob Engel, and Daniel Cremers. “Large-scale direct slam for omnidirectional cameras”. In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE. 2015, pp. 141–148.

[17] Averil B Chatfield. *Fundamentals of high accuracy inertial navigation*. American Institute of Aeronautics and Astronautics, 1997.

[18] Jing Chen, Tianbo Liu, and Shaojie Shen. “Tracking a moving target in cluttered environments using a quadrotor”. In: *International Conference on Intelligent Robots and Systems*. Deajeon,Korea, Oct. 2016, pp. 446–453.

[19] Jing Chen and Shaojie Shen. "Using a quadrotor to track a moving target with arbitrary relative motion patterns". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 5310–5317.

[20] Gregory Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Vol. 2. Springer Science & Business Media, 2011.

[21] Michael Chojnacki and Vadim Indelman. "Vision-based dynamic target trajectory and ego-motion estimation using incremental light bundle adjustment". In: *International Journal of Micro Air Vehicles* 10.2 (2018), pp. 157–170.

[22] Javier Civera, Andrew J Davison, and JM Martinez Montiel. "Inverse depth parametrization for monocular SLAM". In: *IEEE transactions on robotics* 24.5 (2008), pp. 932–945.

[23] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)". In: *arXiv preprint arXiv:1511.07289* (2015).

[24] Frank Dellaert. *Factor graphs and GTSAM: A hands-on introduction*. Tech. rep. Georgia Institute of Technology, 2012.

[25] Jeffrey Delmerico and Davide Scaramuzza. "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Brisbane, Australia, May 2018.

[26] Ryan C DuToit et al. "Consistent map-based 3D localization on mobile devices". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 6253–6260.

[27] Ethan Eade. "Gauss-Newton / Levenberg-Marquardt optimization". In: *Technical Report* (2013).

[28] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. *Closed-form Preintegration Methods for Graph-based Visual-Inertial Navigation*. Tech. rep. RPNG-2018-CPI. Available: http://udel.edu/~ghuang/papers/tr_cpi.pdf. University of Delaware, 2018.

[29] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "Closed-form preintegration methods for graph-based visual–inertial navigation". In: *The International Journal of Robotics Research* 38.5 (2019), pp. 563–586. DOI: 10.1177/0278364919835021.

[30] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "Direct visual-inertial navigation with analytical preintegration". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Singapore, May 2017, pp. 1429–1435.

[31] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "High-accuracy preintegration for visual-inertial navigation". In: *Proc. of the International Workshop on the Algorithmic Foundations of Robotics*. San Francisco, CA, Dec. 2016.

[32] Kevin Eckenhoff, Patrick Geneva, and Guoquan Huang. "Sensor-failure-resilient multi-imu visual-inertial navigation". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3542–3548.

[33] Kevin Eckenhoff, Liam Paull, and Guoquan Huang. "Decoupled, consistent node removal and edge sparsification for graph-based SLAM". In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Daejeon, Korea, Oct. 2016, pp. 3275–3282.

[34] Kevin Eckenhoff et al. "Multi-camera visual-inertial navigation with online intrinsic and extrinsic calibration". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 3158–3164.

[35] Kevin Eckenhoff et al. "Schmidt-EKF-based visual inertial moving object tracking". In: *2020 International Conference on Robotics and Automation (ICRA)*. June 2020.

[36] Kevin Eckenhoff et al. “Tightly-coupled visual-inertial localization and 3-D rigid-body target tracking”. In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1541–1548.

[37] Lars-Peter Ellekilde et al. “Dense 3D map construction for indoor search and rescue”. In: *Journal of Field Robotics* 24.1-2 (2007), pp. 71–89.

[38] Jakob Engel, Vladlen Koltun, and Daniel Cremers. “Direct sparse odometry”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.3 (2018), pp. 611–625.

[39] Jakob Engel, Thomas Schöps, and Daniel Cremers. “LSD-SLAM: Large-scale direct monocular SLAM”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 834–849.

[40] Jakob Engel, Jörg Stückler, and Daniel Cremers. “Large-scale direct SLAM with stereo cameras”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 1935–1942.

[41] Alessandro Filippeschi et al. “Survey of motion tracking methods based on inertial sensors: A focus on upper limb human motion”. In: *Sensors* 17.6 (2017), p. 1257.

[42] Christian Forster et al. “IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation”. In: Rome, Italy, July 2015.

[43] Christian Forster et al. “On-manifold preintegration for real-time visual–inertial odometry”. In: *IEEE Transactions on Robotics* 33.1 (2016), pp. 1–21.

[44] Paul Furgale, Joern Rehder, and Roland Siegwart. “Unified temporal and spatial calibration for multi-sensor systems”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 1280–1286.

[45] Fadri Furrer et al. “RotorS—A modular Gazebo MAV simulator framework”. In: *Robot Operating System (ROS)*. Springer, 2016, pp. 595–625.

[46] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition*. Providence, RI, June 2012.

[47] Patrick Geneva, Kevin Eckenhoff, and Guoquan Huang. "A linear-complexity EKF for visual-inertial navigation with loop closures". In: *Proc. International Conference on Robotics and Automation*. Montreal, Canada, May 2019.

[48] Patrick Geneva, James Maley, and Guoquan Huang. "An Efficient Schmidt-EKF for 3D Visual-Inertial SLAM". In: *Proc. Conference on Computer Vision and Pattern Recognition (CVPR)*. (accepted). Long Beach, CA, June 2019.

[49] Patrick Geneva et al. "OpenVINS: A Research Platform for Visual-Inertial Estimation". In: *IROS 2019 Workshop on Visual-Inertial Navigation: Challenges and Applications*. Macau, China, Nov. 2019. URL: https://github.com/rpng/open_vins.

[50] Giorgio Grisetti et al. "A tutorial on graph-based SLAM". In: *IEEE Intelligent Transportation Systems Magazine* 2.4 (2010), pp. 31–43.

[51] Chao X Guo and Stergios I Roumeliotis. "IMU-RGBD camera navigation using point and plane features". In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Tokyo, Japan, Nov. 2013, pp. 3164–3171.

[52] Chao X Guo et al. "Resource-aware large-scale cooperative three-dimensional mapping using multiple mobile devices". In: *IEEE Transactions on Robotics* 34.5 (2018), pp. 1349–1369.

[53] Chao Guo et al. "Efficient visual-inertial navigation using a rolling-shutter camera with inaccurate timestamps". In: *Proc. of the Robotics: Science and Systems Conference*. Berkeley, CA, July 2014.

[54] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[55] Mina Henein et al. "Simultaneous localization and mapping with dynamic rigid objects". In: *ArXiv* abs/1805.03800 (2018).

[56] Christoph Hertzberg et al. "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds". In: *Information Fusion* 14.1 (2013), pp. 57–77.

[57] Joel A Hesch et al. "Camera-IMU-based localization: Observability analysis and consistency improvement". In: *The International Journal of Robotics Research* 33.1 (2014), pp. 182–201.

[58] Joel A Hesch et al. "Consistency analysis and improvement of vision-aided inertial navigation". In: *IEEE Transactions on Robotics* 30.1 (2013), pp. 158–176.

[59] Sebastian Houben et al. "Efficient multi-camera visual-inertial SLAM for micro aerial vehicles". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 1616–1622.

[60] Zheng Huai and Guoquan Huang. "Robocentric visual-inertial odometry". In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. (in review). Madrid, Spain, Oct. 2018.

[61] Guoquan Huang. "Improving the Consistency of Nonlinear Estimators: Analysis, Algorithms, and Applications". PhD thesis. Department of Computer Science and Engineering, University of Minnesota, 2012. URL: http://people.csail.mit.edu/ghuang/paper/thesis.pdf.

[62] Guoquan Huang, Michael Kaess, and John Leonard. "Consistent Sparsification for Graph Optimization". In: *Proc. of the European Conference on Mobile Robots*. Barcelona, Spain, Sept. 2013, pp. 150–157.

[63] Guoquan Huang, Michael Kaess, and John Leonard. "Towards Consistent Visual-Inertial Navigation". In: *Proc. of the IEEE International Conference on Robotics and Automation*. Hong Kong, China, May 2014, pp. 4926–4933.

[64] Guoquan Huang, Anastasios I. Mourikis, and Stergios I. Roumeliotis. "A First-Estimates Jacobian EKF for Improving SLAM Consistency". In: *Proc. of the 11th International Symposium on Experimental Robotics*. Athens, Greece, July 2008.

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

[66] Adnane Jadid et al. *MultiIMU: Fusing Multiple Calibrated IMUs For Enhanced Mixed Reality Tracking*. Tech. rep. TUM-I1976. 2019.

[67] Andrew H Jazwinski. *Stochastic processes and filtering theory*. Courier Corporation, 2007.

[68] Mahesh K Jeerage. "Reliability analysis of fault-tolerant IMU architectures with redundant inertial sensors". In: *IEEE Symposium on Position Location and Navigation. A Decade of Excellence in the Navigation Sciences*. IEEE. 1990, pp. 587–592.

[69] Michael Kaess et al. "iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3281–3288.

[70] Dongshin Kim, Seunghak Shin, and In So Kweon. "On-Line Initialization and Extrinsic Calibration of an Inertial Navigation System With a Relative Preintegration Method on Manifold". In: *IEEE Transactions on Automation Science and Engineering* 15.3 (2017), pp. 1272–1285.

[71] Nathan Koenig and Andrew Howard. "Design and use paradigms for Gazebo, an open-source multi-robot simulator". In: *Proc. 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*. Vol. 3. IEEE. 2004, pp. 2149–2154.

[72] D.G. Kottas, K.J. Wu, and S.I. Roumeliotis. "Detecting and dealing with hovering maneuvers in vision-aided inertial navigation systems". In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. Nov. 2013, pp. 3172–3179.

[73] Dimitrios G Kottas and Stergios I Roumeliotis. "An iterative Kalman smoother for robust 3D localization on mobile and wearable devices". In: *Proc. of IEEE International Conference on Robotics and Automation*. Seattle, WA, May 2015, pp. 6336–6343.

[74] Rainer Kümmerle et al. "g 2 o: A general framework for graph optimization". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE. 2011, pp. 3607–3613.

[75] Stefan Leutenegger et al. "Keyframe-based visual–inertial odometry using nonlinear optimization". In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334.

[76] Mingyang Li. "Visual-inertial odometry on resource-constrained systems". PhD thesis. UC Riverside, 2014.

[77] Mingyang Li and Anastasios I Mourikis. "High-precision, consistent EKF-based visual-inertial odometry". In: *The International Journal of Robotics Research* 32.6 (2013), pp. 690–711.

[78] Mingyang Li and Anastasios I Mourikis. "Online temporal calibration for camera–IMU systems: Theory and algorithms". In: *The International Journal of Robotics Research* 33.7 (2014), pp. 947–964.

[79] Mingyang Li and Anastasios I Mourikis. "Optimization-based estimator design for vision-aided inertial navigation". In: *Robotics: Science and Systems*. Berlin Germany. 2013, pp. 241–248.

[80] Mingyang Li and Anastasios I. Mourikis. "Vision-aided inertial navigation with rolling-shutter cameras". In: *International Journal of Robotics Research* 33.11 (Sept. 2014), pp. 1490–1507.

[81] Mingyang Li and Anastasios I. Mourikis. "Vision-aided inertial navigation with rolling-shutter cameras". In: *International Journal of Robotics Research* 33.11 (Sept. 2014), pp. 1490–1507.

[82] Mingyang Li et al. "High-fidelity sensor modeling and self-calibration in vision-aided inertial navigation". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 409–416.

[83] Peiliang Li, Tong Qin, et al. "Stereo vision-based semantic 3d object and ego-motion tracking for autonomous driving". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 646–661.

[84] X Rong Li and Vesselin P Jilkov. "Survey of maneuvering target tracking. Part V. Multiple-model methods". In: *IEEE Transactions on Aerospace and Electronic Systems* 41.4 (2005), pp. 1255–1321.

[85] Yonggen Ling, Tianbo Liu, and Shaojie Shen. "Aggressive quadrotor flight using dense visual-inertial fusion". In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1499–1506.

[86] Haomin Liu et al. "Ice-ba: Incremental, consistent and efficient bundle adjustment for visual-inertial slam". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1974–1982.

[87] Bruce D. Lucas and Takeo Kanade. "An iterative image registration technique with an application to stereo vision". In: *International Joint Conference on Artificial Intelligence*. Vancouver, BC, Aug. 1981, pp. 674–679.

[88] Todd Lupton and Salah Sukkarieh. "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions". In: *IEEE Transactions on Robotics* 28.1 (2011), pp. 61–76.

[89] Jeremy Ma et al. “Real-time pose estimation of a dynamic quadruped in GPS-denied environments for 24-hour operation”. In: *The International Journal of Robotics Research* 35.6 (2016), pp. 631–653.

[90] Agostino Martinelli. “Visual-inertial structure from motion: Observability and resolvability”. In: *International Conference on Intelligent Robots and Systems*. Tokyo, Japan, Nov. 2013, pp. 4235–4242.

[91] F. M. Mirzaei and S. I. Roumeliotis. “A Kalman filter-based Algorithm for IMU-Camera Calibration”. In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. San Diego, CA, Oct. 2007, pp. 2427–2434.

[92] Anastasios I Mourikis et al. “Vision-aided inertial navigation for precise planetary landing: analysis and experiments.” In: *Robotics: Science and Systems*. Atlanta, GA. 2007.

[93] Anastasios I Mourikis et al. “Vision-aided inertial navigation for spacecraft entry, descent, and landing”. In: *IEEE Transactions on Robotics* 25.2 (2009), pp. 264–280.

[94] Anastasios Mourikis and Stergios I. Roumeliotis. “A multi-state constraint Kalman filter for vision-aided inertial navigation”. In: *IEEE International Conference on Robotics and Automation*. Rome, Italy, Apr. 2007, pp. 3565–3572.

[95] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. “ORB-SLAM: a versatile and accurate monocular SLAM system”. In: *IEEE Transactions on Robotics* 15.2 (2015), pp. 1147–1163.

[96] Raul Mur-Artal and Juan Tardós. “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras”. In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262.

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

[98] Janosch Nikolic et al. "A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM". In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE. 2014, pp. 431–437.

[99] Roman Y Novoselov et al. "Mitigating the effects of residual biases with Schmidt-Kalman filtering". In: *2005 7th International Conference on Information Fusion*. Vol. 1. IEEE. 2005, 8–pp.

[100] OpenCV Developers Team. *Open Source Computer Vision (OpenCV) Library*. Available: http://opencv.org.

[101] Mrinal K Paul and Stergios I Roumeliotis. "Alternating-stereo VINS: Observability analysis and performance evaluation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4729–4737.

[102] Mrinal K Paul et al. "A comparative analysis of tightly-coupled monocular, binocular, and stereo VINS". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 165–172.

[103] Tong Qin, Peiliang Li, and Shaojie Shen. "Vins-mono: A robust and versatile monocular visual-inertial state estimator". In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 1004–1020.

[104] Kejie Qiu et al. "Tracking 3-D Motion of Dynamic Objects Using Monocular Visual-Inertial Sensing". In: *IEEE Transactions on Robotics* 35.4 (2019), pp. 799–816.

[105] Raouf Rasoulzadeh and Alireza Mohammad Shahri. "Implementation of A low-cost multi-IMU hardware by using a homogenous multi-sensor fusion". In: *2016*

4th International Conference on Control, Instrumentation, and Automation (ICCIA). IEEE. 2016, pp. 451–456.

[106] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[107] Joern Rehder et al. "Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 4304–4311.

[108] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[109] Edward Rosten, Reid Porter, and Tom Drummond. "Faster and better: A machine learning approach to corner detection". In: *IEEE transactions on pattern analysis and machine intelligence* 32.1 (2010), pp. 105–119.

[110] Stergios I Roumeliotis and Joel W Burdick. "Stochastic cloning: A generalized framework for processing relative state measurements". In: *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*. Vol. 2. IEEE. 2002, pp. 1788–1795.

[111] Ethan Rublee et al. "ORB: An efficient alternative to SIFT or SURF". In: *2011 International conference on computer vision*. Ieee. 2011, pp. 2564–2571.

[112] Stanley F Schmidt. "Application of state-space methods to navigation problems". In: *Advances in control systems*. Vol. 3. Elsevier, 1966, pp. 293–340.

[113] David Schubert et al. "The TUM VI benchmark for evaluating visual-inertial odometry". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1680–1687.

[114] Shaojie Shen, Nathan Michael, and Vijay Kumar. "Tightly-coupled monocular visual-inertial fusion for autonomous flight of rotorcraft MAVs". In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 5303–5310.

[115] Wenzhe Shi et al. "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 1874–1883.

[116] Joan Sola. "Towards visual localization, mapping and moving objects tracking by a mobile robot: a geometric and probabilistic approach". PhD thesis. 2007.

[117] Ke Sun et al. "Robust stereo visual inertial odometry for fast autonomous flight". In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 965–972.

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

[119] Vladyslav Usenko et al. "Direct visual-inertial odometry with stereo cameras". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 1885–1892.

[120] Vladyslav Usenko et al. "Visual-inertial mapping with non-linear factor recovery". In: *IEEE Robotics and Automation Letters* (2019).

[121] Lukas Von Stumberg, Vladyslav Usenko, and Daniel Cremers. "Direct sparse visual-inertial odometry using dynamic marginalization". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2510–2517.

[122] Chieh-Chih Wang et al. "Simultaneous localization, mapping and moving object tracking". In: *The International Journal of Robotics Research* 26.9 (2007), pp. 889–916.

[123] Rui Wang, Martin Schwörer, and Daniel Cremers. "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras". In: *International Conference on Computer Vision (ICCV), Venice, Italy*. 2017.

[124] Hongchuan Wei et al. "Visibility-based Motion Planning for Active Target Tracking and Localization". In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Chicago, IL, Sept. 2014, pp. 76–82. DOI: 10.1109/IROS.2014.6942543. URL: http://dx.doi.org/10.1109/IROS.2014.6942543.

[125] Kejian J Wu et al. "A square root inverse filter for efficient vision-aided inertial navigation on mobile devices". In: *2015 Robotics: Science and Systems Conference, RSS 2015*. MIT Press Journals. 2015.

[126] Kejian J Wu et al. "VINS on wheels". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 5155–5162.

[127] Chun Yang, Erik Blasch, and Phil Douville. "Design of Schmidt-Kalman filter for target tracking with navigation errors". In: *2010 IEEE Aerospace Conference*. IEEE. 2010, pp. 1–12.

[128] Yulin Yang and Guoquan Huang. "Aided inertial navigation with geometric features: Observability analysis". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 2334–2340.

[129] Yulin Yang, James Maley, and Guoquan Huang. "Null-space-based marginalization: Analysis and algorithm". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 6749–6755.

[130] Ming Zhang et al. "A Lightweight and Accurate Localization Algorithm Using Multiple Inertial Measurement Units". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 1508–1515.

[131] Zichao Zhang and Davide Scaramuzza. "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 7244–7251.

[132] Zichao Zhang and Davide Scaramuzza. "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 7244–7251.

Appendix A

CLOSED-FORM PREINTEGRATION DERIVATIONS

Here we derive the preintegration means and Jacobians associated with each model.

A.1 Model 1: Piecewise Constant Measurements

IMU preintegration seeks to directly reduce the computational complexity of incorporating inertial measurements by removing the need to re-integrate the propagation function and noise covariance. This is achieved by processing IMU measurements in a *local* frame of reference, yielding measurements that are, in contrast to Equation (3.2), independent of the state [88].

Specifically, by denoting $\Delta T = t_{k+1} - t_k$, we have the following relationship between a series of IMU measurements, the start state, and the resulting end state [31]:

$$ {}^{G}\mathbf{p}_{k+1} = {}^{G}\mathbf{p}_k + {}^{G}\mathbf{v}_k \Delta T - \frac{1}{2}{}^{G}\mathbf{g}\Delta T^2 + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a\right) du ds \tag{A.1} $$

$$ {}^{G}\mathbf{v}_{k+1} = {}^{G}\mathbf{v}_k - {}^{G}\mathbf{g}\Delta T + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a\right) du \tag{A.2} $$

$$ {}^{k+1}_{G}\mathbf{R} = {}^{k+1}_{k}\mathbf{R} \, {}^{k}_{G}\mathbf{R} \tag{A.3} $$

$$ \mathbf{b}_{\omega_{k+1}} = \mathbf{b}_{\omega_k} + \int_{t_k}^{t_{k+1}} \mathbf{n}_{\omega b} \, du \tag{A.4} $$

$$ \mathbf{b}_{a_{k+1}} = \mathbf{b}_{a_k} + \int_{t_k}^{t_{k+1}} \mathbf{n}_{ab} \, du \tag{A.5} $$

From the above, we define the following preintegrated IMU measurements:

$$ {}^{k}\boldsymbol{\alpha}_{k+1} = \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a\right) du ds \tag{A.6} $$

$$ {}^{k}\boldsymbol{\beta}_{k+1} = \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R} \left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a\right) du \tag{A.7} $$

where along with these preintegrated inertial measurements the preintegrated relative-orientation measurement ${}^{k+1}_{k}\bar{q}$ (or ${}^{k+1}_{k}\mathbf{R}$) can be obtained from the integration of the gyro measurements.

To remove the dependencies of the above preintegrated measurements on the true biases, we linearize about the current bias estimates at time step t_k, $\mathbf{b}^{\star}_{a_k}$ and $\mathbf{b}^{\star}_{\omega_k}$. Defining $\Delta\mathbf{b} = \mathbf{b} - \mathbf{b}^{\star}$, we have:

$$
{}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_{k} - {}^{G}\mathbf{v}_{k}\Delta T + \frac{1}{2}{}^{G}\mathbf{g}\Delta T^{2}\right) \simeq
$$

$$
{}^{k}\boldsymbol{\alpha}_{k+1}\left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k}\right) + \frac{\partial \boldsymbol{\alpha}}{\partial \mathbf{b}_{\omega}}\Big|_{\mathbf{b}^{\star}_{\omega_k}} \Delta\mathbf{b}_{\omega} + \frac{\partial \boldsymbol{\alpha}}{\partial \mathbf{b}_{a}}\Big|_{\mathbf{b}^{\star}_{a_k}} \Delta\mathbf{b}_{a} \tag{A.8}
$$

$$
{}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_{k} + {}^{G}\mathbf{g}\Delta T\right) \simeq {}^{k}\boldsymbol{\beta}_{k+1}\left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k}\right) + \frac{\partial \boldsymbol{\beta}}{\partial \mathbf{b}_{\omega}}\Big|_{\mathbf{b}^{\star}_{\omega_k}} \Delta\mathbf{b}_{\omega} + \frac{\partial \boldsymbol{\beta}}{\partial \mathbf{b}_{a}}\Big|_{\mathbf{b}^{\star}_{a_k}} \Delta\mathbf{b}_{a} \tag{A.9}
$$

$$
{}^{k+1}_{G}\mathbf{R}\ {}^{k}_{G}\mathbf{R}^{\top} \simeq \mathbf{R}\left(\frac{\partial \mathbf{R}}{\partial \mathbf{b}_{\omega}}\Big|_{\mathbf{b}^{\star}_{\omega_k}} \Delta\mathbf{b}_{\omega}\right) {}^{k+1}_{k}\mathbf{R}\left(\mathbf{b}^{\star}_{\omega_k}\right) \tag{A.10}
$$

Note that Equations (A.8) and (A.9) are simple Taylor series expansions for our ${}^{k}\boldsymbol{\alpha}_{k+1}$ and ${}^{k}\boldsymbol{\beta}_{k+1}$ measurements, while Equation (A.10) models an additional rotation induced due to a change of the linearization point (estimate) of the gyro bias [31, 42].

The preintegrated measurement's mean values, ${}^{k}\breve{\boldsymbol{\alpha}}_{k+1}$, ${}^{k}\breve{\boldsymbol{\beta}}_{k+1}$, and ${}^{k+1}_{k}\breve{\bar{q}}$, must be computed for use in graph optimization. It is important to note that current preintegration methods [88, 42, 85] are all based on discrete integration of the measurement dynamics through Euler or midpoint integration. In particular, the discrete approximation used by Forster et al. [42] in fact corresponds to a piecewise constant while propagating the robot from one time to the next would create an edge between the corresponding historical statesglobal acceleration model. By contrast, we here offer *closed-form* solutions for the measurement means under the assumptions of piecewise constant *measurements* and of piecewise constant *local acceleration* (in Section A.2).

A.1.1 Measurement Mean

We now derive the closed-from solutions for ${}^{k+1}_{k}\breve{q}$, ${}^{k}\breve{\boldsymbol{\alpha}}_{k+1}$, ${}^{k}\breve{\boldsymbol{\beta}}_{k+1}$. In particular, the quaternion ${}^{\tau+1}_{\tau}\breve{q}$ can be found using the zeroth order quaternion integrator [118]. This can be compounded successively to get the final measurement mean ${}^{k+1}_{k}\breve{q}$. In a similar fashion, we can find, in closed form, ${}^{k}\breve{\boldsymbol{\alpha}}_{\tau+1}$ and ${}^{k}\breve{\boldsymbol{\beta}}_{\tau+1}$. We have derived the following continuous time linear system:

$$\begin{bmatrix} {}^{k}\dot{\breve{\boldsymbol{\alpha}}}_{u} \\ {}^{k}\dot{\breve{\boldsymbol{\beta}}}_{u} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{u} \\ {}^{k}\breve{\boldsymbol{\beta}}_{u} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ {}^{k}_{u}\breve{\mathbf{R}} \end{bmatrix} (\mathbf{a}_m - \mathbf{b}^{\star}_{a_k}) \tag{A.11}$$

The solution of this linear dynamical system is given by:

$$\begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau+1} \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau+1} \end{bmatrix} = \boldsymbol{\Phi}(t_{\tau+1}, t_\tau) \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau} \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau} \end{bmatrix} + \int_{t_\tau}^{t_{\tau+1}} \boldsymbol{\Phi}(t_{\tau+1}, u) \begin{bmatrix} \mathbf{0} \\ {}^{k}_{u}\breve{\mathbf{R}} \end{bmatrix} (\mathbf{a}_m - \mathbf{b}^{\star}_{a_k}) \, du \tag{A.12}$$

where the state-transition matrix $\boldsymbol{\Phi}(t_{\tau+1}, t_\tau)$ is given by the matrix exponential:

$$\boldsymbol{\Phi}(t_{\tau+1}, t_\tau) = \exp\left(\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \Delta t \right) = \mathbf{I} + \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \Delta t + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \Delta t^2 = \begin{bmatrix} \mathbf{I} & \mathbf{I}\Delta t \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{A.13}$$

where $\Delta t = t_{\tau+1} - t_\tau$. Substituting the state-transition into Equation (A.12) yields:

$$\begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau+1} \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau+1} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{I}\Delta t \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau} \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau} \end{bmatrix} + \int_{t_\tau}^{t_{\tau+1}} \begin{bmatrix} \mathbf{I} & \mathbf{I}(t_{\tau+1} - u) \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ {}^{k}_{u}\breve{\mathbf{R}} \end{bmatrix} \hat{\mathbf{a}} \, du \tag{A.14}$$

$$= \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau} + {}^{k}\breve{\boldsymbol{\beta}}_{\tau}\Delta t \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau} \end{bmatrix} + \int_{t_\tau}^{t_{\tau+1}} \begin{bmatrix} (t_{\tau+1} - u) {}^{k}_{u}\breve{\mathbf{R}}\hat{\mathbf{a}} \\ {}^{k}_{u}\breve{\mathbf{R}}\hat{\mathbf{a}} \end{bmatrix} du \tag{A.15}$$

$$= \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau} + {}^{k}\breve{\boldsymbol{\beta}}_{\tau}\Delta t \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau} \end{bmatrix} + \begin{bmatrix} {}^{k}_{\tau+1}\breve{\mathbf{R}} \int_{t_\tau}^{t_{\tau+1}} (t_{\tau+1} - u) {}^{\tau+1}_{u}\breve{\mathbf{R}}\hat{\mathbf{a}} \, du \\ {}^{k}_{\tau+1}\breve{\mathbf{R}} \int_{t_\tau}^{t_{\tau+1}} {}^{\tau+1}_{u}\breve{\mathbf{R}}\hat{\mathbf{a}} \, du \end{bmatrix} \tag{A.16}$$

where $\hat{\mathbf{a}} = \mathbf{a}_m - \mathbf{b}^{\star}_{a_k}$. Using $\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}_m - \mathbf{b}^{\star}_{\omega_k}$ and $\delta t = (t_{\tau+1} - u)$, and the Rodrigues' formula (A.17) we have:

$${}^{\tau+1}_{u}\mathbf{R} = \exp(-\lfloor \hat{\boldsymbol{\omega}} \times \rfloor \delta t) = \mathbf{I} - \frac{\sin(|\hat{\boldsymbol{\omega}}|\delta t)}{|\hat{\boldsymbol{\omega}}|} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + \frac{1 - \cos(|\hat{\boldsymbol{\omega}}|\delta t)}{|\hat{\boldsymbol{\omega}}|^2} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \tag{A.17}$$

$$\begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau+1} \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau+1} \end{bmatrix} = \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau} + {}^{k}\breve{\boldsymbol{\beta}}_{\tau}\Delta t \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau} \end{bmatrix} + \begin{bmatrix} {}^{k}_{\tau+1}\breve{\mathbf{R}} \int_0^{\Delta t} (\delta t)(\mathbf{I} - \frac{\sin(|\hat{\boldsymbol{\omega}}|(\delta t))}{|\hat{\boldsymbol{\omega}}|}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{1-\cos(|\hat{\boldsymbol{\omega}}|\delta t)}{|\hat{\boldsymbol{\omega}}|^2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2)(\hat{\mathbf{a}})d\delta t \\ {}^{k}_{\tau+1}\breve{\mathbf{R}} \int_0^{\Delta t} (\mathbf{I} - \frac{\sin(|\hat{\boldsymbol{\omega}}|(\delta t))}{|\hat{\boldsymbol{\omega}}|}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{1-\cos(|\hat{\boldsymbol{\omega}}|\delta t)}{|\hat{\boldsymbol{\omega}}|^2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2)(\hat{\mathbf{a}})d\delta t \end{bmatrix} \tag{A.18}$$

$$= \begin{bmatrix} {}^{k}\breve{\boldsymbol{\alpha}}_{\tau} + {}^{k}\breve{\boldsymbol{\beta}}_{\tau}\Delta t \\ {}^{k}\breve{\boldsymbol{\beta}}_{\tau} \end{bmatrix}$$
$$+ \begin{bmatrix} {}^{k}_{\tau+1}\breve{\mathbf{R}}(\frac{(\Delta t^2)}{2}\mathbf{I} + \frac{|\hat{\boldsymbol{\omega}}|\Delta t\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|w|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{(|\hat{\boldsymbol{\omega}}|\Delta t)^2 - 2\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - 2(|\hat{\boldsymbol{\omega}}|\Delta t)\sin(|\hat{\boldsymbol{\omega}}|\Delta t) + 2}{2|\hat{\boldsymbol{\omega}}|^4}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2)(\hat{\mathbf{a}}) \\ {}^{k}_{\tau+1}\breve{\mathbf{R}}(\Delta t\mathbf{I} - \frac{1-\cos(|\hat{\boldsymbol{\omega}}|(\Delta t))}{|\hat{\boldsymbol{\omega}}|^2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|\hat{\boldsymbol{\omega}}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2)(\hat{\mathbf{a}}) \end{bmatrix} \tag{A.19}$$

A.1.2 Acceleration Bias Jacobians

We have derived the closed-form measurement update for ${}^{k}\breve{\boldsymbol{\alpha}}_{\tau}$ and ${}^{k}\breve{\boldsymbol{\beta}}_{\tau}$ (see Equation (A.19)). We need only compute the gradients of our closed-form update expressions with respect to changes in bias. In particular, since each update term is linear in the estimated acceleration, $\hat{\mathbf{a}} = \mathbf{a}_m - \mathbf{b}^{\star}_{a_k}$, we can find the bias Jacobians of ${}^{k}\boldsymbol{\alpha}_{k+1}$ and ${}^{k}\boldsymbol{\beta}_{k+1}$ with respect to $\mathbf{b}_a$ as follows:

$$\begin{bmatrix} \frac{\partial \boldsymbol{\alpha}}{\partial \mathbf{b}_a} \\ \frac{\partial \boldsymbol{\beta}}{\partial \mathbf{b}_a} \end{bmatrix} =: \begin{bmatrix} \mathbf{H}_{\alpha}(\tau+1) \\ \mathbf{H}_{\beta}(\tau+1) \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{\alpha}(\tau) + \mathbf{H}_{\beta}(\tau)\Delta t \\ \mathbf{H}_{\beta}(\tau) \end{bmatrix} \tag{A.20}$$
$$- \begin{bmatrix} {}^{k}_{\tau+1}\breve{\mathbf{R}} \left(\frac{\Delta t^2}{2}\mathbf{I}_{3\times 3} + \frac{|\hat{\boldsymbol{\omega}}|\Delta t\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|\hat{\boldsymbol{\omega}}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{(|\hat{\boldsymbol{\omega}}|\Delta t)^2 - 2\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - 2(|\hat{\boldsymbol{\omega}}|\Delta t)\sin(|\hat{\boldsymbol{\omega}}|\Delta t) + 2}{2|\hat{\boldsymbol{\omega}}|^4}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2 \right) \\ {}^{k}_{\tau+1}\breve{\mathbf{R}} \left(\Delta t\mathbf{I}_{3\times 3} - \frac{1-\cos(|\hat{\boldsymbol{\omega}}|(\Delta t))}{|\hat{\boldsymbol{\omega}}|^2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|\boldsymbol{\omega}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2 \right) \end{bmatrix}$$

and for small values of $\hat{\boldsymbol{\omega}}$ we have:

$$\lim_{|\hat{\boldsymbol{\omega}}|\to 0} \begin{bmatrix} \mathbf{H}_{\alpha}(\tau+1) \\ \mathbf{H}_{\beta}(\tau+1) \end{bmatrix} = \begin{bmatrix} \mathbf{H}_{\alpha}(\tau) + \mathbf{H}_{\beta}(\tau)\Delta t \\ \mathbf{H}_{\beta}(\tau) \end{bmatrix} - \begin{bmatrix} {}^{k}_{\tau+1}\breve{\mathbf{R}}(\frac{\Delta t^2}{2}\mathbf{I} - \frac{\Delta t^3}{3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{\Delta t^4}{8}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2) \\ {}^{k}_{\tau+1}\breve{\mathbf{R}}(\Delta t\mathbf{I} - \frac{\Delta t^2}{2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{\Delta t^3}{6}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2) \end{bmatrix} \tag{A.21}$$

A.1.3 Gyro Bias Jacobians

We find the derivatives of ${}^{k}\boldsymbol{\alpha}_{\tau+1}$ and ${}^{k}\boldsymbol{\beta}_{\tau+1}$ with respect to each entry of the gyro bias by taking the derivative with respect to each gyro bias entry. We seek to find

the following entries:

$$\mathbf{J}_\alpha = \begin{bmatrix} \frac{\partial^k \boldsymbol{\alpha}_{\tau+1}}{\partial \mathbf{b}_{\omega_1}} & \frac{\partial^k \boldsymbol{\alpha}_{\tau+1}}{\partial \mathbf{b}_{\omega_2}} & \frac{\partial^k \boldsymbol{\alpha}_{\tau+1}}{\partial \mathbf{b}_{\omega_3}} \end{bmatrix} \tag{A.22}$$

$$\mathbf{J}_\beta = \begin{bmatrix} \frac{\partial^k \boldsymbol{\beta}_{\tau+1}}{\partial \mathbf{b}_{\omega_1}} & \frac{\partial^k \boldsymbol{\beta}_{\tau+1}}{\partial \mathbf{b}_{\omega_2}} & \frac{\partial^k \boldsymbol{\beta}_{\tau+1}}{\partial \mathbf{b}_{\omega_3}} \end{bmatrix} \tag{A.23}$$

For each of the above entries, from Equation (A.19), we have the following:

$$\begin{aligned} \frac{\partial^k \boldsymbol{\alpha}_{\tau+1}}{\partial \mathbf{b}_{\omega_i}} = & \frac{\partial^k \boldsymbol{\alpha}_{\tau}}{\partial \mathbf{b}_{\omega_i}} + \frac{\partial^k \boldsymbol{\beta}_{\tau} \Delta t}{\partial \mathbf{b}_{\omega_i}} \\ & + \frac{\partial}{\partial \mathbf{b}_{w_i}} \Bigg({}^{k}_{\tau+1}\mathbf{R} \Big(\frac{(\Delta t^2)}{2} \mathbf{I}_{3\times 3} + \frac{|\hat{\boldsymbol{\omega}}| \Delta t \cos(|\hat{\boldsymbol{\omega}}| \Delta t) - \sin(|\hat{\boldsymbol{\omega}}| \Delta t)}{|\hat{\boldsymbol{\omega}}|^3} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor \\ & + \frac{(|\hat{\boldsymbol{\omega}}| \Delta t)^2 - 2\cos(|\hat{\boldsymbol{\omega}}| \Delta t) - 2(|\hat{\boldsymbol{\omega}}| \Delta t)\sin(|\hat{\boldsymbol{\omega}}| \Delta t) + 2}{2|\hat{\boldsymbol{\omega}}|^4} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \Big) \Bigg) (\hat{\mathbf{a}}) \end{aligned} \tag{A.24}$$

The first two terms are from the previous integration time step, as we build these Jacobians incrementally. Defining $\hat{\mathbf{r}}_i$ as the unit vector in the i'th direction, and $\hat{\boldsymbol{\omega}}_i$ the corresponding entry in $\hat{\boldsymbol{\omega}}$ the third term can be found as the following:

$$\begin{aligned} \frac{\partial^k_{\tau+1}\mathbf{R}(\frac{(\Delta t)^2}{2}\mathbf{I} + f_1 \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + f_2 \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2)}{\partial \mathbf{b}_{\omega_i}} = & \frac{\partial^k_{\tau+1}\mathbf{R}}{\partial \mathbf{b}_{\omega_i}} \left(\frac{(\Delta t)^2}{2}\mathbf{I} + f_1 \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + f_2 \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 \right) \\ & + {}^{k}_{\tau+1}\breve{\mathbf{R}} \left(\frac{\partial f_1}{\partial \mathbf{b}_{\omega_i}} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor - f_1 \lfloor \hat{\mathbf{r}}_i \times \rfloor + \frac{\partial f_2}{\partial \mathbf{b}_{\omega_i}} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 - f_2 (\lfloor \hat{\mathbf{r}}_i \times \rfloor \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + \lfloor \hat{\boldsymbol{\omega}} \times \rfloor \lfloor \hat{\mathbf{r}}_i \times \rfloor) \right) \end{aligned} \tag{A.25}$$

where f_1 and f_2 are the corresponding coefficients in Equation (A.24), and their derivatives are computed as:

$$\frac{\partial f_1}{\partial \mathbf{b}_{\omega_i}} = \frac{\hat{\boldsymbol{\omega}}_i (|\hat{\boldsymbol{\omega}}|^2 \Delta t^2 \sin(|\hat{\boldsymbol{\omega}}| \Delta t) - 3\sin(|\hat{\boldsymbol{\omega}}| \Delta t) + 3|\hat{\boldsymbol{\omega}}| \Delta t \cos(|\hat{\boldsymbol{\omega}}| \Delta t))}{|\hat{\boldsymbol{\omega}}|^5} \tag{A.26}$$

$$\frac{\partial f_2}{\partial \mathbf{b}_{\omega_i}} = \frac{\hat{\boldsymbol{\omega}}_i ((|\hat{\boldsymbol{\omega}}| \Delta t)^2 - 4\cos(|\hat{\boldsymbol{\omega}}| \Delta t) - 4(|\hat{\boldsymbol{\omega}}| \Delta t)\sin(|\hat{\boldsymbol{\omega}}| \Delta t) + (|\hat{\boldsymbol{\omega}}| \Delta t)^2 \cos(|\hat{\boldsymbol{\omega}}| \Delta t) + 4)}{|\hat{\boldsymbol{\omega}}|^6} \tag{A.27}$$

For small $\hat{\boldsymbol{\omega}}$,

$$\lim_{|\hat{\boldsymbol{\omega}}| \to 0} \frac{\partial f_1}{\partial \mathbf{b}_{\omega_i}} = -\hat{\boldsymbol{\omega}}_i \frac{\Delta t^5}{15} \tag{A.28}$$

$$\lim_{|\hat{\boldsymbol{\omega}}| \to 0} \frac{\partial f_2}{\partial \mathbf{b}_{\omega_i}} = \hat{\boldsymbol{\omega}}_i \frac{\Delta t^6}{72} \tag{A.29}$$

Similarly, we have:

$$\frac{\partial^k \boldsymbol{\beta}_{\tau+1}}{\partial \mathbf{b}_{\omega_i}} = \frac{\partial^k \boldsymbol{\beta}_{\tau}}{\partial \mathbf{b}_{\omega_i}} + \frac{\partial^k_{\tau+1}\mathbf{R}}{\partial \mathbf{b}_{\omega_i}} \left(\Delta t \mathbf{I} + f_3 \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + f_4 \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2\right) \hat{\mathbf{a}} \\ + {}^k_{\tau+1}\breve{\mathbf{R}} \left(\frac{\partial f_3}{\partial \mathbf{b}_{\omega_i}} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor - f_3 \lfloor \hat{\mathbf{r}}_i \times \rfloor + \frac{\partial f_4}{\partial \mathbf{b}_{\omega_i}} \lfloor \hat{\boldsymbol{\omega}} \times \rfloor^2 - f_4 (\lfloor \hat{\mathbf{r}}_i \times \rfloor \lfloor \hat{\boldsymbol{\omega}} \times \rfloor + \lfloor \hat{\boldsymbol{\omega}} \times \rfloor \lfloor \hat{\mathbf{r}}_i \times \rfloor) \right) \hat{\mathbf{a}} \tag{A.30}$$

where

$$\frac{\partial f_3}{\partial \mathbf{b}_{\omega_i}} = \frac{\hat{\boldsymbol{\omega}}_i (2(\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - 1) + (|\hat{\boldsymbol{\omega}}|\Delta t)\sin(|\hat{\boldsymbol{\omega}}|\Delta t))}{|\hat{\boldsymbol{\omega}}|^4} \tag{A.31}$$

$$\frac{\partial f_4}{\partial \mathbf{b}_{\omega_i}} = \frac{\hat{\boldsymbol{\omega}}_i (2(|\hat{\boldsymbol{\omega}}|\Delta t) + (|\hat{\boldsymbol{\omega}}|\Delta t)\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - 3\sin(|\hat{\boldsymbol{\omega}}|\Delta t))}{|\hat{\boldsymbol{\omega}}|^5} \tag{A.32}$$

For small $\hat{\boldsymbol{\omega}}$ we have:

$$\lim_{|\hat{\boldsymbol{\omega}}| \to 0} \frac{\partial f_3}{\partial \mathbf{b}_{\omega_i}} = -\hat{\boldsymbol{\omega}}_i \frac{\Delta t^4}{12} \tag{A.33}$$

$$\lim_{|\hat{\boldsymbol{\omega}}| \to 0} \frac{\partial f_4}{\partial \mathbf{b}_{\omega_i}} = \hat{\boldsymbol{\omega}}_i \frac{\Delta t^5}{60} \tag{A.34}$$

We now show how to derive the derivative of the rotation matrix with respect to a change in bias, i.e., $\Delta \mathbf{b}_\omega = \mathbf{b}_\omega - \mathbf{b}^\star_{\omega_k}$. To do so, we consider a set of measurements over the interval and for the first measurement interval, $[t_k, t_{\tau_1}]$, with measurement $\boldsymbol{\omega}_{m_1}$, we integrate over its sub-interval to get the following:

$${}^{\tau_1}_{k}\mathbf{R} = ({}^{k}_{\tau_1}\mathbf{R})^\top = (\exp(\lfloor \boldsymbol{\omega}_{m_1} - \mathbf{b}^\star_{\omega_k} - \Delta \mathbf{b}_\omega \times \rfloor \Delta t))^\top \tag{A.35}$$

$$\simeq (\exp(\lfloor \boldsymbol{\omega}_{m_1} - \mathbf{b}^\star_{\omega_k} \times \rfloor \Delta t) \exp(\lfloor -\mathbf{J}_{r_1} \Delta \mathbf{b}_\omega \times \rfloor \Delta t))^\top \tag{A.36}$$

$$= \exp(\lfloor \mathbf{J}_{r_1} \Delta \mathbf{b}_\omega \times \rfloor \Delta t) \exp(-\lfloor \boldsymbol{\omega}_{m_1} - \mathbf{b}^\star_{\omega_k} \times \rfloor \Delta t) \tag{A.37}$$

$$= \exp(\lfloor \mathbf{J}_{r_1} \Delta \mathbf{b}_\omega \times \rfloor \Delta t) {}^{\tau_1}_{k}\breve{\mathbf{R}} \tag{A.38}$$

$$\simeq (\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_{r_1} \Delta \mathbf{b}_\omega \times \rfloor \Delta t) {}^{\tau_1}_{k}\breve{\mathbf{R}} \tag{A.39}$$

where $\mathbf{J}_{r_i}$ is the right Jacobian of $SO(3)$ evaluated at the i'th measurement (i.e., $\boldsymbol{\omega}_{m_i}$), see Equation (F.24). This decomposition can be done for every measurement interval:

$${}^{\tau_{i+1}}_{\tau_i}\mathbf{R} \simeq (\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_{r_{i+1}} \Delta \mathbf{b}_\omega \times \rfloor \Delta t) {}^{\tau_{i+1}}_{\tau_i}\breve{\mathbf{R}} \tag{A.40}$$

We can now look at what happens when we compound measurements. In particular, at the second-step $[t_{\tau_1}, t_{\tau_2}]$, we have the following:

$$
\begin{aligned}
{}^{\tau_2}_{k}\mathbf{R} &= {}^{\tau_2}_{\tau_1}\mathbf{R}\,{}^{\tau_1}_{k}\mathbf{R} && \text{(A.41)}\\
&\simeq (\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_{r_2}\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_2}_{\tau_1}\breve{\mathbf{R}}(\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_{r_1}\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_1}_{k}\breve{\mathbf{R}} && \text{(A.42)}\\
&= (\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_{r_2}\Delta\mathbf{b}_\omega\times\rfloor\Delta t)(\mathbf{I}_{3\times 3} + \lfloor {}^{\tau_2}_{\tau_1}\breve{\mathbf{R}}\mathbf{J}_{r_1}\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_2}_{k}\breve{\mathbf{R}} && \text{(A.43)}\\
&\approx (\mathbf{I}_{3\times 3} + \lfloor (\mathbf{J}_{r_2} + {}^{\tau_2}_{\tau_1}\breve{\mathbf{R}}\mathbf{J}_{r_1})\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_2}_{k}\breve{\mathbf{R}} && \text{(A.44)}
\end{aligned}
$$

Here we have used the property that $\mathbf{R}\lfloor\mathbf{w}\times\rfloor\mathbf{R}^\top = \lfloor\mathbf{R}\mathbf{w}\times\rfloor$ for a rotation matrix. Repeating this process for the interval $[t_{\tau_2}, t_{\tau_3}]$ yields:

$$
\begin{aligned}
{}^{\tau_3}_{k}\mathbf{R} &= {}^{\tau_3}_{\tau_2}\mathbf{R}\,{}^{\tau_2}_{k}\mathbf{R} && \text{(A.45)}\\
&\simeq (\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_{r_3}\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_3}_{\tau_2}\breve{\mathbf{R}}(\mathbf{I}_{3\times 3} + \lfloor (\mathbf{J}_{r_2} + {}^{\tau_2}_{\tau_1}\breve{\mathbf{R}}\mathbf{J}_{r_1})\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_2}_{k}\breve{\mathbf{R}} && \text{(A.46)}\\
&= (\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_{r_3}\Delta\mathbf{b}_\omega\times\rfloor)(\mathbf{I}_{3\times 3} + \lfloor ({}^{\tau_3}_{\tau_2}\breve{\mathbf{R}}\mathbf{J}_{r_2} + {}^{\tau_3}_{\tau_1}\breve{\mathbf{R}}\mathbf{J}_{r_1})\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_3}_{k}\breve{\mathbf{R}} && \text{(A.47)}\\
&\simeq (\mathbf{I}_{3\times 3} + \lfloor (\mathbf{J}_{r_3} + {}^{\tau_3}_{\tau_2}\breve{\mathbf{R}}\mathbf{J}_{r_2} + {}^{\tau_3}_{\tau_1}\breve{\mathbf{R}}\mathbf{J}_{r_1})\Delta\mathbf{b}_\omega\times\rfloor\Delta t)\,{}^{\tau_3}_{k}\breve{\mathbf{R}} && \text{(A.48)}
\end{aligned}
$$

We thus see the pattern developing and can write the updated rotation at any time step t_u as:

$$
\begin{aligned}
{}^{u}_{k}\mathbf{R} &= \exp(\lfloor \mathbf{J}_q(u)(\mathbf{b}_\omega - \mathbf{b}^\star_{\omega_k})\times\rfloor)\,{}^{u}_{k}\breve{\mathbf{R}} && \text{(A.49)}\\
\text{with}\quad \mathbf{J}_q(u) &= \sum_{\tau=\tau_1}^{u} {}^{u}_{\tau}\breve{\mathbf{R}}\mathbf{J}_{r_\tau}\Delta t && \text{(A.50)}
\end{aligned}
$$

Each of these values can be calculated *incrementally* by noting that:

$$
\begin{aligned}
\mathbf{J}_q(u+1) &= {}^{u+1}_{u}\breve{\mathbf{R}}\sum_{\tau=\tau_1}^{u} {}^{u}_{\tau}\breve{\mathbf{R}}\mathbf{J}_{r_\tau}\Delta t + \mathbf{J}_{r_{u+1}}\Delta t && \text{(A.51)}\\
&= {}^{u+1}_{u}\breve{\mathbf{R}}\mathbf{J}_q(u) + \mathbf{J}_{r_{u+1}}\Delta t && \text{(A.52)}
\end{aligned}
$$

The derivative of every rotation with respect to the i'th entry of the gyro bias, which appears in both Equation (A.25) and (A.30) can be approximated using:

$$
{}^{u}_{k}\mathbf{R} \simeq (\mathbf{I}_{3\times 3} + \lfloor \mathbf{J}_q(u)(\mathbf{b}_\omega - \mathbf{b}^\star_{\omega_k})\times\rfloor)\,{}^{u}_{k}\breve{\mathbf{R}} \qquad \text{(A.53)}
$$

$$\frac{\partial_k^u \mathbf{R}}{\partial \mathbf{b}_{\omega_i}} \approx \lfloor \mathbf{J}_q(u)\hat{\mathbf{r}}_i \times \rfloor {}_k^u\breve{\mathbf{R}} \tag{A.54}$$

$$\frac{\partial_u^k \mathbf{R}}{\partial \mathbf{b}_{\omega_i}} \approx -{}_u^k\breve{\mathbf{R}} \lfloor \mathbf{J}_q(u)\hat{\mathbf{r}}_i \times \rfloor \tag{A.55}$$

The total rotation after a bias update can be expressed as:

$${}_k^{k+1}\mathbf{R} = \exp(\lfloor \mathbf{J}_q(k+1)(\mathbf{b}_\omega - \mathbf{b}^\star_{\omega_k}) \times \rfloor) {}_k^{k+1}\breve{\mathbf{R}} \tag{A.56}$$

$$\tag{A.57}$$

A.1.4 Measurement Jacobian

Let us partition the preintegrated residual for the first model as:

$$\mathbf{r}_{IMU} = \begin{bmatrix} \mathbf{r}_\theta^\top & \mathbf{r}_p^\top & \mathbf{r}_v^\top & \mathbf{r}_{b_\omega}^\top & \mathbf{r}_{b_a}^\top \end{bmatrix}^\top \tag{A.58}$$

The measurement Jacobian with respect to one element of the state vector can be found by perturbing the measurement function by the corresponding element. For example, the relative-rotation measurement residual is perturbed by a change in gyro bias around the current estimate (i.e., $\mathbf{b}_{\omega_k} - \mathbf{b}^\star_{\omega_k} = \hat{\mathbf{b}}_{\omega_k} + \tilde{\mathbf{b}}_{\omega_k} - \mathbf{b}^\star_{\omega_k}$):

$$\begin{aligned}
\mathbf{r}_\theta &= 2\mathbf{vec}\left({}_G^{k+1}\hat{\bar{q}} \otimes {}_G^k\hat{\bar{q}}^{-1} \otimes {}_k^{k+1}\breve{\bar{q}}^{-1} \otimes \begin{bmatrix} \frac{\mathbf{J}_q(\hat{\mathbf{b}}_{\omega_k} + \tilde{\mathbf{b}}_{\omega_k} - \mathbf{b}^\star_{\omega_k})}{2} \\ 1 \end{bmatrix} \right) \\
&=: 2\mathbf{vec}\left(\hat{\bar{q}}_r \otimes \begin{bmatrix} \frac{\mathbf{J}_q(\hat{\mathbf{b}}_{\omega_k} + \tilde{\mathbf{b}}_{\omega_k} - \mathbf{b}^\star_{\omega_k})}{2} \\ 1 \end{bmatrix} \right) \\
&= 2\mathbf{vec}\left(\mathcal{L}(\hat{\bar{q}}_r) \begin{bmatrix} \frac{\mathbf{J}_q(\hat{\mathbf{b}}_{\omega_k} + \tilde{\mathbf{b}}_{\omega_k} - \mathbf{b}^\star_{\omega_k})}{2} \\ 1 \end{bmatrix} \right) \\
&= 2\mathbf{vec}\left(\begin{bmatrix} \hat{q}_{r,4}\mathbf{I}_{3\times 3} - \lfloor \hat{\mathbf{q}}_r \times \rfloor & \hat{\mathbf{q}}_r \\ -\hat{\mathbf{q}}_r^\top & \hat{q}_{r,4} \end{bmatrix} \begin{bmatrix} \frac{\mathbf{J}_q(\hat{\mathbf{b}}_{\omega_k} + \tilde{\mathbf{b}}_{\omega_k} - \mathbf{b}^\star_{\omega_k})}{2} \\ 1 \end{bmatrix} \right) \\
&= (\hat{q}_{r,4}\mathbf{I}_{3\times 3} - \lfloor \hat{\mathbf{q}}_r \times \rfloor)\mathbf{J}_q(\hat{\mathbf{b}}_{\omega_k} + \tilde{\mathbf{b}}_{\omega_k} - \mathbf{b}^\star_{\omega_k}) + \text{other terms}
\end{aligned}$$

So that our Jacobian with respect to a perturbance in bias is:

$$\frac{\partial \mathbf{r}_\theta}{\partial \tilde{\mathbf{b}}_{\omega_k}} = (\hat{q}_{r,4}\mathbf{I}_{3\times 3} - \lfloor \hat{\mathbf{q}}_r \times \rfloor)\mathbf{J}_q \tag{A.59}$$

Similarly, the Jacobian with respect to ${}^{k+1}_{G}\tilde{\boldsymbol{\theta}}$ can be found as follows:

$$
\begin{aligned}
\mathbf{r}_\theta &= 2\mathbf{vec}\left(\begin{bmatrix}\frac{{}^{k+1}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix} \otimes {}^{k+1}_{G}\hat{\bar{q}} \otimes {}^{k}_{G}\hat{\bar{q}}^{-1} \otimes {}^{k+1}_{k}\bar{q}^{-1} \otimes \hat{\bar{q}}_b\right) \\
&= 2\mathbf{vec}\left(\begin{bmatrix}\frac{{}^{k+1}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix} \otimes \hat{\bar{q}}_{rb}\right) \\
&= 2\mathbf{vec}\left(\mathcal{R}(\hat{q}_{rb})\begin{bmatrix}\frac{{}^{k+1}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix}\right) \\
&= 2\mathbf{vec}\left(\begin{bmatrix}\hat{q}_{rb,4}\mathbf{I}_{3\times3} + \lfloor\hat{\mathbf{q}}_{rb}\times\rfloor & \hat{\mathbf{q}}_{rb}\\ -\hat{\mathbf{q}}_{rb}^\top & \hat{q}_{rb,4}\end{bmatrix}\begin{bmatrix}\frac{{}^{k+1}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix}\right) \\
&= (\hat{q}_{rb,4}\mathbf{I}_{3\times3} + \lfloor\hat{\mathbf{q}}_{rb}\times\rfloor){}^{k+1}_{G}\tilde{\boldsymbol{\theta}} + \text{ other terms}
\end{aligned}
$$

Yielding the Jacobian:

$$
\frac{\partial \mathbf{r}_\theta}{\partial {}^{k+1}_{G}\tilde{\boldsymbol{\theta}}} = \hat{q}_{rb,4}\mathbf{I}_{3\times3} + \lfloor\hat{\mathbf{q}}_{rb}\times\rfloor \tag{A.60}
$$

The Jacobian with respect to ${}^{k}_{G}\tilde{\boldsymbol{\theta}}$ is given by:

$$
\begin{aligned}
\mathbf{r}_\theta &= 2\mathbf{vec}\left({}^{k+1}_{G}\hat{\bar{q}} \otimes {}^{k}_{G}\hat{\bar{q}}^{-1} \otimes \begin{bmatrix}-\frac{{}^{k}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix} \otimes {}^{k+1}_{k}\bar{q}^{-1} \otimes \hat{\bar{q}}_b\right) \\
&= 2\mathbf{vec}\left(\hat{q}_n \otimes \begin{bmatrix}-\frac{{}^{k}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix} \otimes \hat{q}_{mb}^{-1}\right) \\
&= 2\mathbf{vec}\left(\mathcal{L}(\hat{q}_n)\mathcal{R}(\bar{q}_{mb}^{-1})\begin{bmatrix}-\frac{{}^{k}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix}\right) \\
&= 2\mathbf{vec}\Big(\begin{bmatrix}\hat{q}_{n,4}\mathbf{I}_{3\times3} - \lfloor\hat{\mathbf{q}}_n\times\rfloor & \hat{\mathbf{q}}_n\\ -\hat{\mathbf{q}}_n^\top & \hat{q}_{n,4}\end{bmatrix} \\
&\quad \begin{bmatrix}\bar{q}_{mb,4}\mathbf{I}_{3\times3} - \lfloor\bar{\mathbf{q}}_{mb}\times\rfloor & -\mathbf{q}_{mb}\\ \mathbf{q}_{mb}^\top & \bar{q}_{mb,4}\end{bmatrix}\begin{bmatrix}-\frac{{}^{k}_{G}\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix}\Big) \\
&= -((\hat{q}_{n,4}\mathbf{I}_{3\times3} - \lfloor\hat{\mathbf{q}}_n\times\rfloor)(q_{mb,4}\mathbf{I}_{3\times3} - \lfloor\mathbf{q}_{mb}\times\rfloor) + \hat{\mathbf{q}}_n\mathbf{q}_{mb}^\top){}^{k}_{G}\tilde{\boldsymbol{\theta}} + \text{ other terms}
\end{aligned}
$$

Which gives the Jacobian:

$$\frac{\partial \mathbf{r}_\theta}{\partial {}^k_G\tilde{\boldsymbol{\theta}}} = -((\hat{q}_{n,4}\mathbf{I}_{3\times 3} - \lfloor \hat{\mathbf{q}}_n \times \rfloor)(\bar{q}_{mb,4}\mathbf{I}_{3\times 3} - \lfloor \mathbf{q}_{mb} \times \rfloor) + \hat{\mathbf{q}}_n \bar{\mathbf{q}}_{mb}^\top)$$

Note than in the preceding Jacobians, we have defined several intermediate quaternions, ($\hat{\bar{q}}_r, \hat{\bar{q}}_{rb}, \hat{\bar{q}}_n$, and $\hat{\bar{q}}_{mb}$) for ease of notation. Following the same methodology, we can find the Jacobians of the $\boldsymbol{\alpha}$ measurement with respect to the position, velocity and bias.

$$\frac{\partial \mathbf{r}_{b_\omega}}{\partial \tilde{\mathbf{b}}_{\omega_k}} = -\mathbf{I} \tag{A.61}$$

$$\frac{\partial \mathbf{r}_{b_\omega}}{\partial \tilde{\mathbf{b}}_{\omega_{k+1}}} = \mathbf{I} \tag{A.62}$$

$$\frac{\partial \mathbf{r}_v}{\partial {}^k_G\tilde{\boldsymbol{\theta}}} = \lfloor {}^k_G\hat{\mathbf{R}}({}^G\hat{\mathbf{v}}_{k+1} - {}^G\hat{\mathbf{v}}_k + {}^G\mathbf{g}\Delta t) \times \rfloor \tag{A.63}$$

$$\frac{\partial \mathbf{r}_v}{\partial \tilde{\mathbf{b}}_{\omega_k}} = -\mathbf{J}_\beta \tag{A.64}$$

$$\frac{\partial \mathbf{r}_v}{\partial {}^G\tilde{\mathbf{v}}_k} = -{}^k_G\hat{\mathbf{R}} \tag{A.65}$$

$$\frac{\partial \mathbf{r}_v}{\partial {}^G\tilde{\mathbf{v}}_{k+1}} = {}^k_G\hat{\mathbf{R}} \tag{A.66}$$

$$\frac{\partial \mathbf{r}_v}{\partial \tilde{\mathbf{b}}_a} = -\mathbf{H}_\beta \tag{A.67}$$

$$\frac{\partial \mathbf{r}_{b_a}}{\partial \tilde{\mathbf{b}}_{a_k}} = -\mathbf{I} \tag{A.68}$$

$$\frac{\partial \mathbf{r}_{b_a}}{\partial \tilde{\mathbf{b}}_{a_{k+1}}} = \mathbf{I} \tag{A.69}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^k_G\tilde{\boldsymbol{\theta}}} = \lfloor {}^k_G\hat{\mathbf{R}} \left({}^G\hat{\mathbf{p}}_{k+1} - {}^G\hat{\mathbf{p}}_k - {}^G\hat{\mathbf{v}}_k\Delta t + \frac{1}{2}{}^G\mathbf{g}\Delta t^2 \right) \times \rfloor \tag{A.70}$$

$$\frac{\partial \mathbf{r}_p}{\partial \tilde{\mathbf{b}}_{\omega_k}} = -\mathbf{J}_\alpha \tag{A.71}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{v}}_k} = -{}^k_G\hat{\mathbf{R}}\Delta t \tag{A.72}$$

$$\frac{\partial \mathbf{r}_p}{\partial \tilde{\mathbf{b}}_{a_k}} = -\mathbf{H}_\alpha \tag{A.73}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{p}}_k} = -{}^k_G\hat{\mathbf{R}} \tag{A.74}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{p}}_{k+1}} = {}^k_G\hat{\mathbf{R}} \tag{A.75}$$

A.2 Model 2: Piecewise Constant Local Acceleration

The previous preintegration (Model 1) assumes that noiseless IMU measurements can be approximated as remaining constant over a sampling interval, which, however, might not always be a good approximation. For example, in the case of an IMU rotating against the direction of gravity, the measurement will change over a sampling interval continuously due to the effect of gravity. In this section, we propose a new preintegration model that instead assumes piecewise constant *true* local acceleration during the sampling time interval, which may better approximate motion dynamics in practice.

To this end, we first rewrite the state dynamics as:

$$ {}^{G}\mathbf{p}_{k+1} = {}^{G}\mathbf{p}_k + {}^{G}\mathbf{v}_k \Delta T + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R}\mathbf{a} \; duds \tag{A.76} $$

$$ {}^{G}\mathbf{v}_{k+1} = {}^{G}\mathbf{v}_k + {}^{G}_{k}\mathbf{R} \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R}\mathbf{a} \; du \tag{A.77} $$

Note that we have moved the effect of gravity back inside the integrals. We then define the following vectors:

$$ \Delta\mathbf{p} = \int_{t_k}^{t_{k+1}} \int_{t_k}^{s} {}^{k}_{u}\mathbf{R}\mathbf{a} \; duds \tag{A.78} $$

$$ \Delta\mathbf{v} = \int_{t_k}^{t_{k+1}} {}^{k}_{u}\mathbf{R}\mathbf{a} \; du \tag{A.79} $$

which essentially are the true local position displacement and velocity change during $[t_k, t_{k+1}]$, and yields:

$$ \Delta\dot{\mathbf{p}} = \Delta\mathbf{v} \tag{A.80} $$

$$ \Delta\dot{\mathbf{v}} = {}^{k}_{u}\mathbf{R}\mathbf{a} \tag{A.81} $$

In particular, between two IMU measurement times inside the preintegration interval, $[t_\tau, t_{\tau+1}] \subset [t_k, t_{k+1}]$, we assume that the *local* acceleration will be constant:

$$ \forall t_u \in [t_\tau, t_{\tau+1}], \quad \mathbf{a}(t_u) = \mathbf{a}(t_\tau) \tag{A.82} $$

Using this sampling model we can rewrite Equation (A.81) as:

$$\Delta\dot{\mathbf{v}} = {}^{k}_{u}\mathbf{R}\left(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a - {}^{\tau}_{k}\mathbf{R}^{k}_{G}\mathbf{R}^{G}\mathbf{g}\right) \tag{A.83}$$

We now write the relationship of the states at the beginning and end of the interval as (see Equations (A.76) and (A.77)):

$${}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_k - {}^{G}\mathbf{v}_k\Delta T\right) = \Delta\mathbf{p} \tag{A.84}$$

$${}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_k\right) = \Delta\mathbf{v} \tag{A.85}$$

It is important to note that, since $\Delta\mathbf{p}$ and $\Delta\mathbf{v}$ are functions of both the biases *and* the initial orientation, we perform the following linearization with respect to these states:

$$\begin{aligned}{}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{k+1} - {}^{G}\mathbf{p}_k - {}^{G}\mathbf{v}_k\Delta T\right) \simeq{}& \Delta\mathbf{p}\left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k}, {}^{k}_{G}\bar{q}^{\star}\right) \\ &+ \frac{\partial\Delta\mathbf{p}}{\partial\mathbf{b}_\omega}\Big|_{\mathbf{b}^{\star}_{\omega_k}}\Delta\mathbf{b}_\omega + \frac{\partial\Delta\mathbf{p}}{\partial\mathbf{b}_a}\Big|_{\mathbf{b}^{\star}_{a_k}}\Delta\mathbf{b}_a + \frac{\partial\Delta\mathbf{p}}{\partial\Delta\boldsymbol{\theta}_k}\Big|_{{}^{k}_{G}\bar{q}^{\star}}\Delta\boldsymbol{\theta}_k\end{aligned} \tag{A.86}$$

$${}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{v}_{k+1} - {}^{G}\mathbf{v}_k\right) \simeq \Delta\mathbf{v}\left(\mathbf{b}^{\star}_{\omega_k}, \mathbf{b}^{\star}_{a_k}, {}^{k}_{G}\bar{q}^{\star}\right) + \frac{\partial\Delta\mathbf{v}}{\partial\mathbf{b}_\omega}\Big|_{\mathbf{b}^{\star}_{\omega_k}}\Delta\mathbf{b}_\omega + \frac{\partial\Delta\mathbf{v}}{\partial\mathbf{b}_a}\Big|_{\mathbf{b}^{\star}_{a_k}}\Delta\mathbf{b}_a + \frac{\partial\Delta\mathbf{v}}{\partial\Delta\boldsymbol{\theta}_k}\Big|_{{}^{k}_{G}\bar{q}^{\star}}\Delta\boldsymbol{\theta}_k \tag{A.87}$$

where $\Delta\boldsymbol{\theta}_k = 2\mathbf{vec}\left({}^{k}_{G}\bar{q} \otimes {}^{k}_{G}\bar{q}^{\star-1}\right)$ is the rotation angle change associated with the change of the linearization point of quaternion ${}^{k}_{G}\bar{q}$.

A.2.1 Measurement Mean

To compute the new preintegrated measurement mean values, we first determine the continuous-time dynamics of the expected preintegration vectors by taking expectations of Equations (A.80) and (A.83), given by:

$$\Delta\dot{\breve{\mathbf{p}}} = \Delta\breve{\mathbf{v}} \tag{A.88}$$

$$\Delta\dot{\breve{\mathbf{v}}} = {}^{k}_{u}\breve{\mathbf{R}}\left(\mathbf{a}_m - \mathbf{b}^{\star}_{a_k} - {}^{\tau}_{k}\breve{\mathbf{R}}^{k}_{G}\mathbf{R}^{\star G}\mathbf{g}\right) \tag{A.89}$$

As in the case of Model 1, we can formulate a linear system of the new preintegration measurement vectors and find the closed-from solutions. Specifically, we can integrate these differential equations and obtain the solution similar to Equation (A.19), while using the new definition: $\hat{\mathbf{a}} = \mathbf{a}_m - \mathbf{b}^{\star}_{a_k} - {}^{\tau}_{k}\breve{\mathbf{R}}^{k}_{G}\mathbf{R}^{\star G}\mathbf{g}$, which serves as the estimate for the piecewise constant local acceleration over the sampling interval.

A.2.2 Bias Jacobian Discussion

As we linearize this preintegration with respect to the IMU biases and the initial orientation, it is important to compute the Jacobians with respect to these quantities. In particular, we note that the solution to the preintegration equation for Model 2 can be expressed as:

$$\begin{aligned}\Delta\breve{\mathbf{p}}_{\tau+1} &= \Delta\breve{\mathbf{p}}_{\tau} + \Delta\breve{\mathbf{v}}_{\tau}\Delta t + \mathbf{A}_{\tau}\left(\mathbf{a}_m - \mathbf{b}_a^{\star} - {}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\right)\\ \Delta\breve{\mathbf{v}}_{\tau+1} &= \Delta\breve{\mathbf{v}}_{\tau} + \mathbf{B}_{\tau}\left(\mathbf{a}_m - \mathbf{b}_a^{\star} - {}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\right)\end{aligned} \tag{A.90}$$

where $\mathbf{A}_\tau$ and $\mathbf{B}_\tau$ are defined as the following (see Equation (A.19) for a parallel with model 1):

$$\begin{aligned}\mathbf{A}_{\tau} = {}_{\tau+1}^{k}\breve{\mathbf{R}}\Big(&\frac{\Delta t^2}{2}\mathbf{I}_{3\times3} + \frac{|\hat{\boldsymbol{\omega}}|\Delta t\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|\hat{\boldsymbol{\omega}}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor\\ &+ \frac{(|\hat{\boldsymbol{\omega}}|\Delta t)^2 - 2\cos(|\hat{\boldsymbol{\omega}}|\Delta t) - 2(|\hat{\boldsymbol{\omega}}|\Delta t)\sin(|\hat{\boldsymbol{\omega}}|\Delta t) + 2}{2|\hat{\boldsymbol{\omega}}|^4}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2\Big)\end{aligned} \tag{A.91}$$

$$\mathbf{B}_{\tau} = {}_{\tau+1}^{k}\breve{\mathbf{R}}\Big(\Delta t\mathbf{I}_{3\times3} - \frac{1-\cos(|\hat{\boldsymbol{\omega}}|(\Delta t))}{|\hat{\boldsymbol{\omega}}|^2}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor + \frac{(|\hat{\boldsymbol{\omega}}|\Delta t) - \sin(|\hat{\boldsymbol{\omega}}|\Delta t)}{|\hat{\boldsymbol{\omega}}|^3}\lfloor\hat{\boldsymbol{\omega}}\times\rfloor^2\Big) \tag{A.92}$$

Letting $\mathbf{O}_\alpha$ and $\mathbf{O}_\beta$ denote the Jacobians of the position and velocity preintegrated measurements with respect to the initial orientation, we have:

$$\begin{aligned}\begin{bmatrix}\mathbf{J}_\alpha(\tau+1)\\ \mathbf{J}_\beta(\tau+1)\end{bmatrix} &= \begin{bmatrix}\mathbf{J}_\alpha(\tau) + \mathbf{J}_\beta(\tau)\Delta t\\ \mathbf{J}_\beta(\tau)\end{bmatrix}\\ &+ \begin{bmatrix}\frac{\partial\mathbf{A}_\tau\hat{\mathbf{a}}}{\partial\mathbf{b}_\omega} + \mathbf{A}_\tau\lfloor{}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\times\rfloor\mathbf{J}_q(\tau)\\ \frac{\partial\mathbf{B}_\tau\hat{\mathbf{a}}}{\partial\mathbf{b}_\omega} + \mathbf{B}_\tau\lfloor{}_k^{\tau}\breve{\mathbf{R}}_G^k\mathbf{R}^{\star G}\mathbf{g}\times\rfloor\mathbf{J}_q(\tau)\end{bmatrix}\end{aligned} \tag{A.93}$$

$$\begin{bmatrix}\mathbf{H}_\alpha(\tau+1)\\ \mathbf{H}_\beta(\tau+1)\end{bmatrix} = \begin{bmatrix}\mathbf{H}_\alpha(\tau) + \mathbf{H}_\beta(\tau)\Delta t - \mathbf{A}_\tau\\ \mathbf{H}_\beta(\tau) - \mathbf{B}_\tau\end{bmatrix} \tag{A.94}$$

$$\begin{bmatrix}\mathbf{O}_\alpha(\tau+1)\\ \mathbf{O}_\beta(\tau+1)\end{bmatrix} = \begin{bmatrix}\mathbf{O}_\alpha(\tau) + \mathbf{O}_\beta(\tau)\Delta t\\ \mathbf{O}_\beta(\tau)\end{bmatrix} \tag{A.95}$$

$$- \begin{bmatrix} \mathbf{A}_{\tau_k^\tau} \breve{\mathbf{R}} \lfloor {}^k_G \mathbf{R}^{\star G} \mathbf{g} \times \rfloor \\ \mathbf{B}_{\tau_k^\tau} \breve{\mathbf{R}} \lfloor {}^k_G \mathbf{R}^{\star G} \mathbf{g} \times \rfloor \end{bmatrix}$$

We note that Equation (A.95) reveals that only changes in the initial orientation *perpendicular* to local gravity (${}^k\mathbf{g}$) will cause a change in the preintegrated measurement. As these directions of orientation are observable and thus are expected to have small errors, this highlights the fact that our linearization scheme about the initial orientation is appropriate.

A.2.3 Measurement Jacobian

Let us partition the preintegrated residual for the second model as:

$$\mathbf{r}_{IMU} = \begin{bmatrix} \mathbf{r}_\theta^\top & \mathbf{r}_p^\top & \mathbf{r}_v^\top & \mathbf{r}_{b_\omega}^\top & \mathbf{r}_{b_a}^\top \end{bmatrix}^\top \tag{A.96}$$

Instead of directly computing the derivatives, the measurement Jacobian with respect to one element of the state vector can be found by perturbing the measurement function by the corresponding element. The orientation and bias change measurement Jacobians remain unchanged under the new model. However, since the definition of $\mathbf{r}_v$ and $\mathbf{r}_p$ have changed, their Jacobians must be changed appropriately. The Jacobians are as follows:

$$\frac{\partial \mathbf{r}_{b_\omega}}{\partial \tilde{\mathbf{b}}_{\omega_k}} = -\mathbf{I} \tag{A.97}$$

$$\frac{\partial \mathbf{r}_{b_\omega}}{\partial \tilde{\mathbf{b}}_{\omega_{k+1}}} = \mathbf{I} \tag{A.98}$$

$$\frac{\partial \mathbf{r}_v}{\partial {}^k_G \tilde{\boldsymbol{\theta}}} = \lfloor {}^k_G \hat{\mathbf{R}} \left({}^G\hat{\mathbf{v}}_{k+1} - {}^G\hat{\mathbf{v}}_k \right) \times \rfloor - \mathbf{O}_\beta \left(\tilde{q}_4 \mathbf{I} + \lfloor \tilde{\mathbf{q}} \times \rfloor \right) \tag{A.99}$$

$$\frac{\partial \mathbf{r}_v}{\partial \tilde{\mathbf{b}}_{\omega_k}} = -\mathbf{J}_\beta \tag{A.100}$$

$$\frac{\partial \mathbf{r}_v}{\partial {}^G\tilde{\mathbf{v}}_k} = -{}^k_G \hat{\mathbf{R}} \tag{A.101}$$

$$\frac{\partial \mathbf{r}_v}{\partial {}^G\tilde{\mathbf{v}}_{k+1}} = {}^k_G \hat{\mathbf{R}} \tag{A.102}$$

$$\frac{\partial \mathbf{r}_v}{\partial \tilde{\mathbf{b}}_{a_k}} = -\mathbf{H}_\beta \tag{A.103}$$

$$\frac{\partial \mathbf{r}_{b_a}}{\partial \tilde{\mathbf{b}}_{a_{k+1}}} = \mathbf{I} \tag{A.104}$$

$$\frac{\partial \mathbf{r}_{b_a}}{\partial \tilde{\mathbf{b}}_{a_k}} = -\mathbf{I} \tag{A.105}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^k_G\tilde{\boldsymbol{\theta}}} = \lfloor {}^k_G\hat{\mathbf{R}} \left({}^G\hat{\mathbf{p}}_{k+1} - {}^G\hat{\mathbf{p}}_k - {}^G\hat{\mathbf{v}}_k \Delta T \right) \times \rfloor - \mathbf{O}_\alpha \left(\tilde{q}_4 \mathbf{I} + \lfloor \tilde{\mathbf{q}} \times \rfloor \right) \tag{A.106}$$

$$\frac{\partial \mathbf{r}_p}{\partial \tilde{\mathbf{b}}_{\omega_k}} = -\mathbf{J}_\alpha \tag{A.107}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{v}}_k} = -{}^k_G\hat{\mathbf{R}} \Delta T \tag{A.108}$$

$$\frac{\partial \mathbf{r}_p}{\partial \tilde{\mathbf{b}}_{a_k}} = -\mathbf{H}_\alpha \tag{A.109}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{p}}_k} = -{}^k_G\hat{\mathbf{R}} \tag{A.110}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{p}}_{k+1}} = {}^k_G\hat{\mathbf{R}} \tag{A.111}$$

where $[\tilde{\mathbf{q}}^\top \ \tilde{q}_4]^\top = {}^k_G\hat{\bar{q}} \otimes {}^k_G\bar{q}^{\star -1}$

Appendix B

BATCH MARGINALIZATION

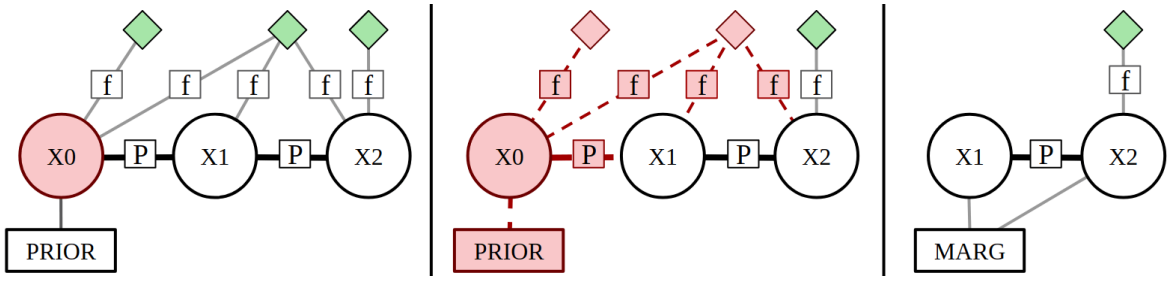


Figure B.1: During graph optimization of VINS, IMU states (shown in circles) and 3D features (diamonds) are included in the graph. Image projection measurements connect features and the IMU state corresponding to the time that the image was recorded. Subsequent IMU states are connected with preintegrated factors, while a prior factor connects to the oldest IMU state. During marginalization, we first select the states to be marginalized, e.g., the oldest IMU state in the window and its associated features (in red). With these measurements we perform marginalization to form a new marginal measurement for future optimization.

In a naive graph SLAM formulation, nodes are continuously added to the graph as time progresses without consideration to the computational burden. For example, as a robot moves through an unknown environment we would add robot state nodes at *every* measurement time. This becomes a problem due to the high computational complexity, $O(n^3)$ with $n = \dim(\tilde{\mathbf{x}})$, of batch optimization, in the worst case. In order to bound the computational complexity of the system, marginalization is often performed to remove a set of nodes, called marginalized states, from the graph, while retaining the information contained in their incident edges (see Figure B.1 for an example) [62, 33]. Partitioning the optimization variables into states remaining after marginalization, $\mathbf{x}_r$, and the to-be marginalized states, $\mathbf{x}_m$, we can write (2.52) as the solution of the following minimization [65]:

$$\{\hat{\mathbf{x}}_r, \hat{\mathbf{x}}_m\} = \underset{\mathbf{x}_r, \mathbf{x}_m}{\operatorname{argmin}} \Big(c_r(\mathbf{x}_r) + c_m(\mathbf{x}_m, \mathbf{x}_r) \Big) \tag{B.1}$$

The second subcost, $c_m(\mathbf{x}_m, \mathbf{x}_r)$, is associated with the measurements incident to the marginalized states, and is a function of both these states and the remaining ones. The first, $c_r(\mathbf{x}_r)$, refers to all other edges in the graph. The optimal estimate for the remaining nodes can be written as:

$$\hat{\mathbf{x}}_r = \underset{\mathbf{x}_r}{\operatorname{argmin}} \left(c_r(\mathbf{x}_r) + \min_{\mathbf{x}_m} c_m(\mathbf{x}_m, \mathbf{x}_r) \right) \tag{B.2}$$

That is, minimizing $c_m(\mathbf{x}_m, \mathbf{x}_r)$ with respect to $\mathbf{x}_m$ yields a cost that is a function *only* of the remaining states. This minimization is performed as in (2.58), where we write out the linear system for only the measurements involved in c_m:

$$\begin{bmatrix} \mathbf{\Lambda}_{rr} & \mathbf{\Lambda}_{rm} \\ \mathbf{\Lambda}_{mr} & \mathbf{\Lambda}_{mm} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{x}}_r \\ \tilde{\mathbf{x}}_m \end{bmatrix} = \begin{bmatrix} -\mathbf{g}_r \\ -\mathbf{g}_m \end{bmatrix} \tag{B.3}$$

The optimal subcost c_m, up to an irrelevant constant, is given by [97]:[1]

$$c_{marg}\left(\mathbf{x}_r\right) = \frac{1}{2} \left|\left| \mathbf{x}_r \boxminus \breve{\mathbf{x}}_r \right|\right|^2_{\mathbf{\Lambda}_{marg}} + \mathbf{g}_{marg}^\top \left(\mathbf{x}_r \boxminus \breve{\mathbf{x}}_r \right) \tag{B.4}$$

where $\breve{\mathbf{x}}_r$ is the linearization point used to build the system (in practice, the current state estimate at the time of marginalization), and $\mathbf{\Lambda}_{marg} = \mathbf{\Lambda}_{rr} - \mathbf{\Lambda}_{rm}\mathbf{\Lambda}_{mm}^{-1}\mathbf{\Lambda}_{mr}$ and $\mathbf{g}_{marg} = \mathbf{g}_r - \mathbf{\Lambda}_{rm}\mathbf{\Lambda}_{mm}^{-1}\mathbf{g}_m$ are the marginalized Hessian and gradient, respectively.

In future optimization, this marginalization creates both a new quadratic *and* linear cost in terms of the error between the remaining states and their linearization points. This then replaces the marginalized measurements in the original graph, and we can write this new cost (B.4) up to a constant in the form of (2.52):

$$c_{marg}\left(\mathbf{x}_r\right) = \frac{1}{2} \left|\left| \mathbf{A}_m \left(\mathbf{x}_r \boxminus \breve{\mathbf{x}}_r \right) + \mathbf{b}_m \right|\right|_2^2 \tag{B.5}$$

$$\text{with } \mathbf{A}_m^\top \mathbf{A}_m = \mathbf{\Lambda}_{marg} \tag{B.6}$$

$$\mathbf{A}_m^\top \mathbf{b}_m = \mathbf{g}_{marg} \tag{B.7}$$

[1] Throughout the thesis, we reserve the symbol $\hat{x}$ to denote the current estimate of state variable x in optimization, while $\breve{x}$ refers to the (inferred) measurement mean value.

This cost yields the following residual and Jacobian for use in optimization (see (2.55) and (2.56)):

$$\mathbf{r}_{marg}(\hat{\mathbf{x}}) = \mathbf{A}_m \left(\hat{\mathbf{x}}_r \boxminus \breve{\mathbf{x}}_r\right) + \mathbf{b}_m \tag{B.8}$$

$$\mathbf{J}_{marg} = \mathbf{A}_m \frac{\partial \left(\left(\hat{\mathbf{x}}_r \boxplus \tilde{\mathbf{x}}_r\right) \boxminus \breve{\mathbf{x}}_r\right)}{\partial \tilde{\mathbf{x}}_r}\Big|_{\tilde{\mathbf{x}}_r = \mathbf{0}} \tag{B.9}$$

where for the Jacobian of a vector (i.e., if $\mathbf{x}_r = \mathbf{v}$):

$$\frac{\partial \left(\left(\hat{\mathbf{v}} \boxplus \tilde{\mathbf{v}}\right) \boxminus \breve{\mathbf{v}}\right)}{\partial \tilde{\mathbf{v}}} = \frac{\partial \left(\hat{\mathbf{v}} + \tilde{\mathbf{v}} - \breve{\mathbf{v}}\right)}{\partial \tilde{\mathbf{v}}} = \mathbf{I} \tag{B.10}$$

and for a quaternion $\bar{q}$, with $\tilde{\bar{q}} = \hat{\bar{q}} \otimes \breve{\bar{q}}^{-1}$, we have:

$$\begin{aligned}
\frac{\partial \left(\left(\hat{\bar{q}} \boxplus \tilde{\boldsymbol{\theta}}\right) \boxminus \breve{\bar{q}}\right)}{\partial \tilde{\boldsymbol{\theta}}} &= \frac{\partial 2\mathbf{vec}\left(\begin{bmatrix} \frac{\tilde{\boldsymbol{\theta}}}{2} \\ 1 \end{bmatrix} \otimes \hat{\bar{q}} \otimes \breve{\bar{q}}^{-1}\right)}{\partial \tilde{\boldsymbol{\theta}}} \\
&= \frac{\partial 2\mathbf{vec}\left(\mathcal{R}\left(\tilde{\bar{q}}\right) \begin{bmatrix} \frac{\tilde{\boldsymbol{\theta}}}{2} \\ 1 \end{bmatrix}\right)}{\partial \tilde{\boldsymbol{\theta}}} \\
&= \tilde{q}_4 \mathbf{I} + \lfloor \tilde{\mathbf{q}} \rfloor
\end{aligned} \tag{B.11}$$

Appendix C

INVERSE-DEPTH MEASUREMENT JACOBIANS

We denote a and i the anchoring time step and the associated anchoring camera frame, respectively. Consider the case where we receive an image of the same feature at step k from camera j. This measurement can be divided into three categories: (i) when the measurement refers to both the anchoring time and camera that the inverse depth is being represented in; (ii) when the measurement refers to the same anchoring time, but a different camera; (iii) when the anchoring time and measurement time are distinct.

In case (i), we have (see Equation (3.64)):

$$\mathbf{h} = \begin{bmatrix} \alpha \\ \beta \\ 1 \end{bmatrix} \tag{C.1}$$

Then the measurement Jacobians are computed by (see Equations (3.63) and (3.64)):

$$\frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\alpha}} = \mathbf{H}_{proj}(0,0,2,1) \tag{C.2}$$

$$\frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\beta}} = \mathbf{H}_{proj}(0,1,2,1) \tag{C.3}$$

$$\frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\rho}} = \mathbf{0} \tag{C.4}$$

$$\mathbf{H}_{proj} = \begin{bmatrix} \frac{1}{h_3} & 0 & \frac{-h_1}{(h_3)^2} \\ 0 & \frac{1}{h_3} & \frac{-h_2}{(h_3)^2} \end{bmatrix} \tag{C.5}$$

where $\mathbf{H}_{proj}(i,j,k,l)$ refers to the block matrix of size (k,l) with starting index (i,j).

In case (ii) where k refers to the same imaging time but a different camera (such as a stereo partner), we have (see Equation (3.64)):

$$\mathbf{h} = {}^{C_j}_{C_i}\mathbf{R} \begin{bmatrix} \alpha \\ \beta \\ 1 \end{bmatrix} + \rho^{C_j}\mathbf{p}_{C_i} \tag{C.6}$$

Because in this case the transformation parameters are rigid and known, we need only the derivatives with respect to the unknown feature parameterization:

$$\frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\alpha}} = \left[\mathbf{H}_{proj}{}^{C_j}_{C_i}\hat{\mathbf{R}}\right](0,0,2,1) \tag{C.7}$$

$$\frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\beta}} = \left[\mathbf{H}_{proj}{}^{C_j}_{C_i}\hat{\mathbf{R}}\right](0,1,2,1) \tag{C.8}$$

$$\frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\rho}} = {}^{C_j}\hat{\mathbf{p}}_{C_i} \tag{C.9}$$

In case (iii) where instead the measurement refers to a different time, we can write out the rigid transformation between the anchor and new current camera frame as follows:

$$\begin{aligned}
{}^{C_{k,j}}\mathbf{p}_f &= {}^{C_{k,j}}_{C_{a,i}}\mathbf{R}\,{}^{C_{a,i}}\mathbf{p}_f + {}^{C_{k,j}}\mathbf{p}_{C_{a,i}} \\
&= \frac{1}{\rho_I}{}^{C_j}\mathbf{R}^k_G\mathbf{R}^G_a\mathbf{R}^I_{C_i}\mathbf{R} \begin{bmatrix} \hat{\alpha} \\ \hat{\beta} \\ 1 \end{bmatrix} + \\
&\quad {}^{C_{k,j}}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{C_{a,i}} - {}^{G}\mathbf{p}_{C_{k,j}}\right) \\
&= \frac{1}{\rho_I}{}^{C_j}\mathbf{R}^k_G\mathbf{R}^G_a\mathbf{R}^I_{C_i}\mathbf{R} \begin{bmatrix} \hat{\alpha} \\ \hat{\beta} \\ 1 \end{bmatrix} + \\
&\quad {}^{C_j}_{I}\mathbf{R}^k_G\mathbf{R}\left({}^{G}\mathbf{p}_a + {}^{G}_{a}\mathbf{R}^I\mathbf{p}_{C_i} - {}^{G}\mathbf{p}_k - {}^{G}_{k}\mathbf{R}^I\mathbf{p}_{C_j}\right) \\
&= {}^{C_j}_{I}\mathbf{R}^k_G\mathbf{R}^G_a\mathbf{R}^I_{C_i}\mathbf{R}\left(\frac{1}{\rho}\begin{bmatrix} \alpha \\ \beta \\ 1 \end{bmatrix} - {}^{C_i}\mathbf{p}_I\right) +
\end{aligned}$$

$$ {}^{C_j}_{I}\mathbf{R}^{k}_{G}\mathbf{R}({}^{G}\mathbf{p}_a - {}^{G}\mathbf{p}_k) + {}^{C_j}\mathbf{p}_I \tag{C.10} $$

With this, we have:

$$ \mathbf{h} = {}^{C_j}_{I}\mathbf{R}^{k}_{G}\mathbf{R}^{G}_{a}\mathbf{R}^{I}_{C_i}\mathbf{R}\left(\begin{bmatrix}\alpha\\ \beta\\ 1\end{bmatrix} - \rho^{C_i}\mathbf{p}_I\right) + $$

$$ \rho^{C_j}_{I}\mathbf{R}^{k}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_a - {}^{G}\mathbf{p}_k\right) + \rho^{C_j}\mathbf{p}_I \tag{C.11} $$

We can then take the derivative with respect to each variable:

$$ \frac{\partial \mathbf{r}_{fjk}}{\partial {}^{a}_{G}\tilde{\boldsymbol{\theta}}} = -\mathbf{H}_{proj}\,{}^{C_j}_{I}\hat{\mathbf{R}}^{k}_{G}\hat{\mathbf{R}}^{G}_{a}\hat{\mathbf{R}}\left|{}^{I}_{C_i}\hat{\mathbf{R}}\left(\begin{bmatrix}\hat{\alpha}\\ \beta\\ 1\end{bmatrix} - \hat{\rho}^{C_i}\hat{\mathbf{p}}_I\right)\right| \tag{C.12} $$

$$ \frac{\partial \mathbf{r}_{fjk}}{\partial {}^{G}\tilde{\mathbf{p}}_a} = \mathbf{H}_{proj}\hat{\rho}^{C_j}_{I}\hat{\mathbf{R}}^{k}_{G}\hat{\mathbf{R}} \tag{C.13} $$

$$ \frac{\partial \mathbf{r}_{fjk}}{\partial {}^{k}_{G}\tilde{\boldsymbol{\theta}}} = \mathbf{H}_{proj}\,{}^{C_j}_{I}\hat{\mathbf{R}}\left|{}^{k}_{G}\hat{\mathbf{R}}^{G}_{a}\hat{\mathbf{R}}^{I}_{C_i}\hat{\mathbf{R}}\left(\begin{bmatrix}\hat{\alpha}\\ \beta\\ 1\end{bmatrix} - \hat{\rho}^{C_i}\hat{\mathbf{p}}_I\right) + \right. $$

$$ \left. \hat{\rho}^{k}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_a - {}^{G}\hat{\mathbf{p}}_k\right)\right| \tag{C.14} $$

$$ \frac{\partial \mathbf{r}_{fjk}}{\partial {}^{G}\tilde{\mathbf{p}}_k} = -\mathbf{H}_{proj}\hat{\rho}^{C_j}_{I}\hat{\mathbf{R}}^{k}_{G}\hat{\mathbf{R}} \tag{C.15} $$

$$ \frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\alpha}} = \left[\mathbf{H}_{proj}\,{}^{C_j}_{I}\hat{\mathbf{R}}^{k}_{G}\hat{\mathbf{R}}^{G}_{a}\hat{\mathbf{R}}^{I}_{C_i}\hat{\mathbf{R}}\right](0,0,2,1) \tag{C.16} $$

$$ \frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\beta}} = \left[\mathbf{H}_{proj}\,{}^{C_j}_{I}\hat{\mathbf{R}}^{k}_{G}\hat{\mathbf{R}}^{G}_{a}\hat{\mathbf{R}}^{I}_{C_i}\hat{\mathbf{R}}\right](0,1,2,1) \tag{C.17} $$

$$ \frac{\partial \mathbf{r}_{fjk}}{\partial \tilde{\rho}} = \mathbf{H}_{proj}(-{}^{C_j}_{I}\hat{\mathbf{R}}^{k}_{G}\hat{\mathbf{R}}^{G}_{a}\hat{\mathbf{R}}^{I}_{C_i}\hat{\mathbf{R}}^{C_i}\hat{\mathbf{p}}_I + $$

$$ {}^{C_j}_{I}\hat{\mathbf{R}}^{k}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_a - {}^{G}\hat{\mathbf{p}}_k\right) + {}^{C_j}\hat{\mathbf{p}}_I) \tag{C.18} $$

Appendix D

RELATIVE-POSE MEASUREMENT JACOBIAN

Recall that in Equation (3.78), j denotes the query image and k is the keyframe. We partition the relative-pose residual $\mathbf{r}_d$ into the relative-orientation residual $\mathbf{r}_\theta$ and the relative-position residual $\mathbf{r}_p$. The Jacobians with respect to the states can be found by perturbation in the same way as before.

$$\begin{aligned}
\mathbf{r}_\theta &= 2\mathbf{vec}\left(\begin{bmatrix}\frac{{}^j_G\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix} \otimes {}^j_G\hat{\bar{q}} \otimes {}^k_G\hat{\bar{q}}^{-1} \otimes {}^j_k\breve{\bar{q}}^{-1}\right) \\
&= 2\mathbf{vec}\left(\mathcal{R}\left({}^j_G\hat{\bar{q}} \otimes {}^k_G\hat{\bar{q}}^{-1} \otimes {}^j_k\breve{\bar{q}}^{-1}\right)\begin{bmatrix}\frac{{}^j_G\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix}\right) \\
&= (\bar{q}_{r,4}\mathbf{I} + \lfloor \mathbf{q}_r \rfloor)\,{}^j_G\tilde{\boldsymbol{\theta}} + \cdots \\
\Rightarrow \quad \frac{\partial \mathbf{r}_\theta}{\partial {}^j_G\tilde{\boldsymbol{\theta}}} &= (\bar{q}_{r,4}\mathbf{I} + \lfloor \mathbf{q}_r \rfloor)
\end{aligned} \tag{D.1}$$

Similarly, we perturb the quaternion estimate of the keyframe to compute the corresponding Jacobian as:

$$\begin{aligned}
\mathbf{r}_\theta &= 2\mathbf{vec}\left({}^j_G\hat{\bar{q}} \otimes {}^k_G\hat{\bar{q}}^{-1} \otimes \begin{bmatrix}\frac{-{}^k_G\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix} \otimes {}^j_k\breve{\bar{q}}^{-1}\right) \\
&= 2\mathbf{vec}\left(\mathcal{L}\left({}^k_j\hat{\bar{q}}\right)\mathcal{R}\left({}^k_j\breve{\bar{q}}\right)^\top \begin{bmatrix}\frac{-{}^k_G\tilde{\boldsymbol{\theta}}}{2}\\ 1\end{bmatrix}\right) = \\
&-\left(\left({}^j_k\hat{\bar{q}}_4\mathbf{I} - \lfloor {}^j_k\hat{\mathbf{q}} \rfloor\right)\left({}^j_k\breve{\bar{q}}_4\mathbf{I} - \lfloor {}^j_k\breve{\mathbf{q}} \rfloor\right) + {}^j_k\hat{\mathbf{q}}\,{}^j_k\breve{\mathbf{q}}^\top\right){}^k_G\tilde{\boldsymbol{\theta}} + \cdots \\
\Rightarrow \frac{\partial \mathbf{r}_\theta}{\partial {}^k_G\tilde{\boldsymbol{\theta}}} &= -\left(\left({}^j_k\hat{\bar{q}}_4\mathbf{I} - \lfloor {}^j_k\hat{\mathbf{q}} \rfloor\right)\left({}^j_k\breve{\bar{q}}_4\mathbf{I} - \lfloor {}^j_k\breve{\mathbf{q}} \rfloor\right) + {}^j_k\hat{\mathbf{q}}\,{}^j_k\breve{\mathbf{q}}^\top\right)
\end{aligned}$$

Again by following a similar procedure, we can find the Jacobians of the relative-position residual with respect to the state as follows:

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{p}}_j} = {}^k_G\hat{\mathbf{R}} \tag{D.2}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^G\tilde{\mathbf{p}}_k} = -{}^k_G\hat{\mathbf{R}} \tag{D.3}$$

$$\frac{\partial \mathbf{r}_p}{\partial {}^k_G\tilde{\boldsymbol{\theta}}} = \lfloor {}^k_G\hat{\mathbf{R}} \left({}^G\hat{\mathbf{p}}_j - {}^G\hat{\mathbf{p}}_k \right) \rfloor \tag{D.4}$$

Appendix E

NEW VARIABLE INITIALIZATION WITHIN THE EKF

During the operation of a filter, it may become necessary to add new variables to the estimation problem. Such an example would be adding newly seen features as the sensor suite explores a new area rather than nullspace projecting them, to allow for tracks longer than the given window size. In this work we utilize the delayed intialization technique presented in [76]. In particular, we collect a set of initializing measurements related to the new variable, $\mathbf{v}$ in a similar matter to how the standard MSCKF collects visual measurements:

$$\mathbf{r} = \mathbf{H}_x\tilde{\mathbf{x}} + \mathbf{H}_v\tilde{\mathbf{v}} + \mathbf{n} \tag{E.1}$$

We note that in general the measurement Jacobian corresponding to the new variable will be tall. We can therefore project this system onto the range and nullspace of this measurement Jacobians. In particular, we use Givens rotations to zero out the bottom (n-k) rows (where k is the dimension of the new variable error state). This yields:

$$\begin{bmatrix}\mathbf{r}_I\\ \mathbf{r}_U\end{bmatrix} = \begin{bmatrix}\mathbf{H}_{Ix}\\ \mathbf{H}_{Ux}\end{bmatrix}\tilde{\mathbf{x}} + \begin{bmatrix}\mathbf{H}_{Iv}\\ \mathbf{0}\end{bmatrix}\tilde{\mathbf{v}} + \begin{bmatrix}\mathbf{n}_I\\ \mathbf{n}_U\end{bmatrix} \tag{E.2}$$

The lower linear system is simply the MSCKF nullspace projection which contains all the information available in the measurements about the current state and can be used to update the system without adding the new variable to the state. We note that as pointed out in [129], the projection we have performed is only optimal if the noise is isotropic, and thus in the case it is not we can simply first perform a whitening transformation to the system before applying the Givens rotations. Next, consider the top system. We note that after projection $\mathbf{H}_{Iv}$ will be invertible.

Thus, we can directly solve for the error of the new variable:

$$\tilde{\mathbf{v}} = \mathbf{H}_{Iv}^{-1} \left(\mathbf{r}_I - \mathbf{H}_{Ix} \tilde{\mathbf{x}} - \mathbf{n}_I \right) \tag{E.3}$$

We can take the expectation of the above system to update our feature estimate:

$$\hat{\tilde{\mathbf{v}}} = \mathbf{H}_{Iv}^{-1} \mathbf{r}_I \tag{E.4}$$

We can then use the definition of the covariance to compute the autocovariance of the feature as well as the cross correlation with the current state:

$$\mathbf{P}_{ff} = \mathbb{E} \left[\left(\tilde{\mathbf{v}} - \hat{\tilde{\mathbf{v}}} \right) \left(\tilde{\mathbf{v}} - \hat{\tilde{\mathbf{v}}} \right)^\top \right] \tag{E.5}$$

$$= \mathbf{H}_{Iv}^{-1} \left(\mathbf{H}_{Ix} \mathbf{P}_{xx} \mathbf{H}_{Ix}^\top + \mathbf{R}_I \right) \mathbf{H}_{Iv}^{-\top} \tag{E.6}$$

$$\mathbf{P}_{xf} = \mathbb{E} \left[(\tilde{\mathbf{x}}) \left(\tilde{\mathbf{v}} - \hat{\tilde{\mathbf{v}}} \right)^\top \right] \tag{E.7}$$

$$= -\mathbf{P}_{xx} \mathbf{H}_{Ix}^\top \mathbf{H}_{Iv}^{-\top} \tag{E.8}$$

We can then use the new feature estimates and covariance entries to add the system to the estimation problem. After this, the updating nullspace system can be used to update the state.

Appendix F

INTERPOLATION MEASUREMENT JACOBIANS

F.1 Linear Interpolation

Consider a 3D feature captured in the image of the base camera b with timestamp ${}^{C_b}t$. The normalized feature measurement for this is given by:

$$\mathbf{z}_k = \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \end{bmatrix} + \mathbf{n}_k \tag{F.1}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = {}^{C_b(t)}\mathbf{p}_f = {}^{C_b}_{I}\mathbf{R}\,{}^{I({}^{b}t_k)}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_{I({}^{b}t_k)}\right) + {}^{C_b}\mathbf{p}_I \tag{F.2}$$

Here $I(t)$ refers to the state of the IMU at true imaging time ${}^{b}t := t$, which gives simple Jacobians because the clone corresponding to $I(t)$ is contained in our state vector.

The chain rule gives:

$$\frac{\partial \mathbf{z}_k}{\partial \tilde{\mathbf{x}}} = \frac{\partial \mathbf{z}_k}{\partial {}^{C_b(t)}\tilde{\mathbf{p}}_f}\frac{\partial {}^{C_b(t)}\tilde{\mathbf{p}}_f}{\partial \tilde{\mathbf{x}}} \tag{F.3}$$

$$\frac{\partial \mathbf{z}_k}{\partial {}^{C_b(t)}\tilde{\mathbf{p}}_f} = \begin{bmatrix} \frac{1}{z} & 0 & -\frac{x}{z^2} \\ 0 & \frac{1}{z} & -\frac{y}{z^2} \end{bmatrix} \tag{F.4}$$

$$\frac{\partial {}^{C_b(t)}\tilde{\mathbf{p}}_f}{\partial {}^{I(t)}_{G}\tilde{\boldsymbol{\theta}}} = {}^{C_b}_{I}\hat{\mathbf{R}}\lfloor {}^{I(t)}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_f - {}^{G}\hat{\mathbf{p}}_{I(t)}\right)\rfloor \tag{F.5}$$

$$\frac{\partial {}^{C_b(t)}\tilde{\mathbf{p}}_f}{\partial {}^{C}_{G}\tilde{\boldsymbol{\theta}}} = \lfloor {}^{C_b}_{I}\hat{\mathbf{R}}\,{}^{I(t)}_{G}\hat{\mathbf{R}}\left({}^{G}\hat{\mathbf{p}}_f - {}^{G}\hat{\mathbf{p}}_{I(t)}\right)\rfloor \tag{F.6}$$

$$\frac{\partial {}^{C_b(t)}\tilde{\mathbf{p}}_f}{\partial {}^{G}\tilde{\mathbf{p}}_{I(t)}} = -{}^{C_b}_{I}\hat{\mathbf{R}}\,{}^{I(t)}_{G}\hat{\mathbf{R}} \tag{F.7}$$

$$\frac{\partial^{C_b(t)}\tilde{\mathbf{p}}_f}{\partial^G\tilde{\mathbf{p}}_f} = {}^{C_b}_{I}\hat{\mathbf{R}}^{I(t)}_{G}\hat{\mathbf{R}} \tag{F.8}$$

Now let us suppose that we we capture an image from another camera, cam i, and suppose that there exists an offset in the timestamps between cam b and cam i, such that if ${}^{C_b}t = {}^{C_i}t + {}^{C_i}t_{C_b}$, where ${}^{C_b}t$ is the true time of the image expressed in the base camera's clock, ${}^{C_i}t$ is the reported timestamp in the measuring camera's clock, and ${}^{C_i}t_{C_b}$ is the unknown time offset. For the measurement of a feature, we have:

$${}^{C_i(t)}\mathbf{p}_f = {}^{C_i}_{I}\mathbf{R}^{I(t)}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_f - {}^{G}\mathbf{p}_{I(t)}\right) + {}^{C_i}\mathbf{p}_I \tag{F.9}$$

We do not keep an estimate for the clone associated with this time, and instead rely on interpolation. Letting ${}^{C_b}t_1 := t_1$ and ${}^{C_b}t_2 := t_2$ denote the bounding IMU clones between which ${}^{C_i}t_k + {}^{C_i}\hat{t}_{C_b} := \hat{t}$ falls, we have:

$${}^{I(t)}_{G}\mathbf{R} = \text{Exp}\left(\hat{\lambda}\text{Log}\left({}^{I(t_2)}_{G}\mathbf{R}^{G}_{I(t_1)}\mathbf{R}\right)\right){}^{I(t_1)}_{G}\mathbf{R} \tag{F.10}$$

$${}^{G}\mathbf{p}_{I(t)} = (1-\lambda){}^{G}\mathbf{p}_{I(t_1)} + \lambda{}^{G}\mathbf{p}_{I(t_2)} \tag{F.11}$$

$$\lambda = \frac{t - t_1}{t_2 - t_1} \tag{F.12}$$

For this measurement, we can use the chain rule to compute the derivatives with respect to the bounding states and the unknown time offset:

$$\frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}} = \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}\frac{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}{\partial^{I(t_2)}_{G}\tilde{\boldsymbol{\theta}}} \tag{F.13}$$

$$\frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^{I(t_2)}_{G}\tilde{\boldsymbol{\theta}}} = \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}\frac{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}{\partial^{I(t_2)}_{G}\tilde{\boldsymbol{\theta}}} \tag{F.14}$$

$$\frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^G\tilde{\mathbf{p}}_{I(t_1)}} = \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^G\tilde{\mathbf{p}}_{I(t)}}\frac{\partial^G\mathbf{p}_{I(t)}}{\partial^G\tilde{\mathbf{p}}_{I(t_1)}} \tag{F.15}$$

$$\frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^G\tilde{\mathbf{p}}_{I(t_2)}} = \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^G\tilde{\mathbf{p}}_{I(t)}}\frac{\partial^G\tilde{\mathbf{p}}_{I(t)}}{\partial^G\tilde{\mathbf{p}}_{I(t_2)}} \tag{F.16}$$

$$\frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^{C_i}\tilde{t}_{C_b}} = \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^G\tilde{\mathbf{p}}_{I(t)}}\frac{\partial^G\tilde{\mathbf{p}}_{I(t)}}{\partial^{C_i}\tilde{t}_{C_b}} + \frac{\partial^{C_i(t)}\tilde{\mathbf{p}}_f}{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}\frac{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}{\partial^{C_i}\tilde{t}_{C_b}} \tag{F.17}$$

The first terms in the chain rule are identical to those derived for the cam b case, but with the calibration between the IMU to cam i. The derivatives with respect to the position are computed as:

$$\frac{\partial {}^{G}\tilde{\mathbf{p}}_{I(t)}}{\partial {}^{G}\tilde{\mathbf{p}}_{I(t_1)}} = (1-\hat{\lambda})\mathbf{I} \tag{F.18}$$

$$\frac{\partial {}^{G}\tilde{\mathbf{p}}_{I(t)}}{\partial {}^{G}\tilde{\mathbf{p}}_{I(t_2)}} = \hat{\lambda}\mathbf{I} \tag{F.19}$$

$$\frac{\partial {}^{G}\tilde{\mathbf{p}}_{I(t)}}{\partial {}^{C_i}\tilde{t}_{C_b}} = \frac{1}{t_1 - t_1}\left({}^{G}\hat{\mathbf{p}}_{I(t_2)} - {}^{G}\hat{\mathbf{p}}_{I(t_1)}\right) \tag{F.20}$$

For the orientation derivatives, we use the following approximations for small angles $\boldsymbol{\psi}$

$$\mathrm{Exp}\left(\boldsymbol{\theta}+\boldsymbol{\psi}\right) \approx \mathrm{Exp}\left(\mathbf{J}_l\left(\boldsymbol{\theta}\right)\boldsymbol{\psi}\right)\mathrm{Exp}\left(\boldsymbol{\theta}\right) \tag{F.21}$$

$$\approx \mathrm{Exp}\left(\boldsymbol{\theta}\right)\mathrm{Exp}\left(\mathbf{J}_r\left(\boldsymbol{\theta}\right)\boldsymbol{\psi}\right) \tag{F.22}$$

where the definition of Jacobians of $SO(3)$ $\mathbf{J}_l(\cdot)$ and $\mathbf{J}_r(\cdot)$ are the following:

$$\mathbf{J}_l(\phi) = \mathbf{I} + \frac{1-cos(\parallel \phi \parallel)}{\parallel \phi \parallel^2}\lfloor\phi\times\rfloor + \frac{\parallel \phi \parallel - sin(\parallel \phi \parallel)}{\parallel \phi \parallel^3}\lfloor\phi\times\rfloor^2 \tag{F.23}$$

$$\mathbf{J}_r(\phi) = \mathbf{I} - \frac{1-cos(\parallel \phi \parallel)}{\parallel \phi \parallel^2}\lfloor\phi\times\rfloor + \frac{\parallel \phi \parallel - sin(\parallel \phi \parallel)}{\parallel \phi \parallel^3}\lfloor\phi\times\rfloor^2 \tag{F.24}$$

We also define for convenience ${}^{2}_{1}\boldsymbol{\phi} = \mathrm{Log}\left({}^{I(t_2)}_{I(t_1)}\mathbf{R}\right)$. In addition we will use the fact that $\mathbf{R}\mathrm{Exp}\left(\boldsymbol{\theta}\right) = \mathrm{Exp}\left(\mathbf{R}\boldsymbol{\theta}\right)\mathbf{R}$. We first expand the rotation interpolation equation with respect to a perturbation in the clone $I(t_1)$:

$$\mathrm{Exp}\left(-{}^{I(t)}_{G}\tilde{\boldsymbol{\theta}}\right){}^{I(t)}_{G}\hat{\mathbf{R}} \tag{F.25}$$

$$= \mathrm{Exp}\left(\hat{\lambda}\mathrm{Log}\left({}^{I(t_2)}_{G}\hat{\mathbf{R}}\,{}^{G}_{I(t_1)}\hat{\mathbf{R}}\mathrm{Exp}\left({}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right)\right)\right)\mathrm{Exp}\left(-{}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right){}^{I(t_1)}_{G}\hat{\mathbf{R}} \tag{F.26}$$

$$\approx \mathrm{Exp}\left(\hat{\lambda}\left({}^{2}_{1}\hat{\boldsymbol{\phi}} + \mathbf{J}_r^{-1}\left({}^{2}_{1}\hat{\boldsymbol{\phi}}\right){}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right)\right)\mathrm{Exp}\left(-{}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right){}^{I(t_1)}_{G}\hat{\mathbf{R}} \tag{F.27}$$

$$\approx \mathrm{Exp}\left(\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right)\mathbf{J}_r^{-1}\left({}^{2}_{1}\hat{\boldsymbol{\phi}}\right){}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right)\mathrm{Exp}\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right)\mathrm{Exp}\left(-{}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right){}^{I(t_1)}_{G}\hat{\mathbf{R}} \tag{F.28}$$

$$= \mathrm{Exp}\left(\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right)\mathbf{J}_r^{-1}\left({}^{2}_{1}\hat{\boldsymbol{\phi}}\right){}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right)\mathrm{Exp}\left(-\mathrm{Exp}\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right){}^{I(t_1)}_{G}\tilde{\boldsymbol{\theta}}\right)\mathrm{Exp}\left(\hat{\lambda}^2_1\hat{\boldsymbol{\phi}}\right){}^{I(t_1)}_{G}\hat{\mathbf{R}} \tag{F.29}$$

$$= \text{Exp}\left(\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\mathbf{J}_r^{-1}\left({}_1^2\hat{\boldsymbol{\phi}}\right){}_G^{I(t_1)}\tilde{\boldsymbol{\theta}}\right)\text{Exp}\left(-\text{Exp}\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right){}_G^{I(t_1)}\tilde{\boldsymbol{\theta}}\right){}_G^{I(t)}\hat{\mathbf{R}} \tag{F.30}$$

$$\approx \text{Exp}\left(\left(\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\mathbf{J}_r^{-1}\left({}_1^2\hat{\boldsymbol{\phi}}\right) - \text{Exp}\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\right){}_G^{I(t_1)}\tilde{\boldsymbol{\theta}}\right){}_G^{I(t)}\hat{\mathbf{R}} \tag{F.31}$$

$$\Rightarrow \frac{\partial_G^{I(t)}\tilde{\boldsymbol{\theta}}}{\partial_G^{I(t_1)}\tilde{\boldsymbol{\theta}}} = -\left(\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\mathbf{J}_r^{-1}\left({}_1^2\hat{\boldsymbol{\phi}}\right) - \text{Exp}\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\right) \tag{F.32}$$

Next we perturb $I(t_2)$

$$\text{Exp}\left(-{}_G^{I(t)}\tilde{\boldsymbol{\theta}}\right){}_G^{I(t)}\hat{\mathbf{R}} \tag{F.33}$$

$$= \text{Exp}\left(\hat{\lambda}\text{Log}\left(\text{Exp}\left(-{}_G^{I(t_2)}\tilde{\boldsymbol{\theta}}\right)\right){}_G^{I(t_2)}\hat{\mathbf{R}}{}_{I(t_1)}^{G}\hat{\mathbf{R}}\right){}_G^{I(t_1)}\hat{\mathbf{R}} \tag{F.34}$$

$$\approx \text{Exp}\left(\hat{\lambda}\left({}_1^2\hat{\boldsymbol{\phi}} - \mathbf{J}_l^{-1}\left({}_1^2\hat{\boldsymbol{\phi}}\right){}_G^{I(t_2)}\tilde{\boldsymbol{\theta}}\right)\right){}_G^{I(t_1)}\hat{\mathbf{R}} \tag{F.35}$$

$$\approx \text{Exp}\left(-\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\mathbf{J}_l^{-1}\left({}_1^2\hat{\boldsymbol{\phi}}\right){}_G^{I(t_2)}\tilde{\boldsymbol{\theta}}\right)\text{Exp}\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right){}_G^{I(t_1)}\hat{\mathbf{R}} \tag{F.36}$$

$$= \text{Exp}\left(-\hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\mathbf{J}_l^{-1}\left({}_1^2\hat{\boldsymbol{\phi}}\right){}_G^{I(t_2)}\tilde{\boldsymbol{\theta}}\right){}_G^{I(t)}\hat{\mathbf{R}} \tag{F.37}$$

$$\Rightarrow \frac{\partial_G^{I(t)}\tilde{\boldsymbol{\theta}}}{\partial_G^{I(t_2)}\tilde{\boldsymbol{\theta}}} = \hat{\lambda}\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right)\mathbf{J}_l^{-1}\left({}_1^2\hat{\boldsymbol{\phi}}\right) \tag{F.38}$$

Lastly we perturb λ by $\tilde{\lambda} = \frac{{}^{C_i}\tilde{t}_{C_b}}{t_2 - t_1}$:

$$\text{Exp}\left(-{}_G^{I(t)}\tilde{\boldsymbol{\theta}}\right){}_G^{I(t)}\hat{\mathbf{R}} \tag{F.39}$$

$$= \text{Exp}\left(\left(\hat{\lambda} + \tilde{\lambda}_k\right)\text{Log}\left({}_G^{I(t_2)}\hat{\mathbf{R}}{}_{I(t_1)}^{G}\hat{\mathbf{R}}\right)\right){}_G^{I(t_1)}\hat{\mathbf{R}} \tag{F.40}$$

$$\approx \text{Exp}\left(\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}})\right){}_1^2\hat{\boldsymbol{\phi}}\tilde{\lambda}_k\right){}_G^{I(t)}\hat{\mathbf{R}} \tag{F.41}$$

$$\Rightarrow \frac{\partial_G^{I(t)}\tilde{\boldsymbol{\theta}}}{\partial\tilde{\lambda}_k} = -\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right){}_1^2\hat{\boldsymbol{\phi}} \tag{F.42}$$

$$\Rightarrow \frac{\partial_G^{I(t)}\tilde{\boldsymbol{\theta}}}{\partial^{C_i}\tilde{t}_{C_b}} = -\frac{1}{t_2 - t_1}\mathbf{J}_l\left(\hat{\lambda}_1^2\hat{\boldsymbol{\phi}}\right){}_1^2\hat{\boldsymbol{\phi}} = -\frac{1}{t_2 - t_1}{}_1^2\hat{\boldsymbol{\phi}} \tag{F.43}$$

F.2 Higher-Order Interpolation Equations

To simplify derivations, we define the following matrices:

$$\mathbf{m}(t) = \begin{bmatrix}\Delta t\mathbf{I}_3 & \Delta t^2\mathbf{I}_3 & \cdots & \Delta t^n\mathbf{I}_3\end{bmatrix} \tag{F.44}$$

$$\mathbf{A}_\theta = \mathbf{m}(t)\mathbf{a}_\theta \tag{F.45}$$

$$\mathbf{A}_p = \mathbf{m}(t)\mathbf{a}_p \tag{F.46}$$

Then we can write the orientation interpolation as:

$$ {}^{I(t)}_{G}\mathbf{R} = \mathbf{Exp}\left(\mathbf{A}_\theta\right) {}^{I(t_0)}_{G}\mathbf{R} \tag{F.47}$$

We now wish to compute the Jacobian of this pose with respect to the poses the polynomial was fit to. We next perturb each side:

$$\begin{aligned}
\mathrm{Exp}\left(-{}^{I(t)}_{G}\tilde{\boldsymbol{\theta}}\right) {}^{I(t)}_{G}\hat{\mathbf{R}} &= \mathrm{Exp}\left(\hat{\mathbf{A}}_\theta + \tilde{\mathbf{A}}_\theta\right) \mathrm{Exp}\left(-{}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}\right) {}^{I(t_0)}_{G}\hat{\mathbf{R}} && \text{(F.48)}\\
&\approx \mathrm{Exp}\left(\mathbf{J}_l\left(\hat{\mathbf{A}}_\theta\right)\tilde{\mathbf{A}}_\theta\right) \mathrm{Exp}\left(\hat{\mathbf{A}}_\theta\right) \mathrm{Exp}\left(-{}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}\right) {}^{I(t_0)}_{G}\hat{\mathbf{R}} && \text{(F.49)}\\
&= \mathrm{Exp}\left(\mathbf{J}_l\left(\hat{\mathbf{A}}_\theta\right)\tilde{\mathbf{A}}_\theta\right) \mathrm{Exp}\left(-\mathrm{Exp}\left(\hat{\mathbf{A}}_\theta\right) {}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}\right) \mathrm{Exp}\left(\hat{\mathbf{A}}_\theta\right) {}^{I(t_0)}_{G}\hat{\mathbf{R}} && \text{(F.50)}\\
&\approx \mathrm{Exp}\left(\mathbf{J}_l\left(\hat{\mathbf{A}}_\theta\right)\tilde{\mathbf{A}}_\theta - \mathrm{Exp}\left(\hat{\mathbf{A}}_\theta\right) {}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}\right) {}^{I(t)}_{G}\hat{\mathbf{R}} && \text{(F.51)}
\end{aligned}$$

Thus we can immediately pull out Jacobians as:

$$\begin{aligned}
\frac{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}{\partial^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}} &= -\mathbf{J}_l\left(\hat{\mathbf{A}}_\theta\right)\frac{\partial\tilde{\mathbf{A}}_\theta}{\partial^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}} + \mathrm{Exp}\left(\hat{\mathbf{A}}_\theta\right) && \text{(F.52)}\\
\frac{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}{\partial^{I(t_k)}_{G}\tilde{\boldsymbol{\theta}}} &= -\mathbf{J}_l\left(\hat{\mathbf{A}}_\theta\right)\frac{\partial\tilde{\mathbf{A}}_\theta}{\partial^{I(t_k)}_{G}\tilde{\boldsymbol{\theta}}} && \text{(F.53)}\\
& && \text{(F.54)}
\end{aligned}$$

In order to compute this, we also perturb $\mathbf{A}_\theta$:

$$\begin{aligned}
\hat{\mathbf{A}}_\theta + \tilde{\mathbf{A}}_\theta &= \mathbf{m}(t)\left(\hat{\mathbf{a}}_\theta + \tilde{\mathbf{a}}_\theta\right) && \text{(F.55)}\\
&= \mathbf{m}(t)\left(\hat{\mathbf{a}}_\theta + \mathbf{V}^{-1}\Delta\tilde{\boldsymbol{\phi}}\right) && \text{(F.56)}\\
\Rightarrow \frac{\partial\tilde{\mathbf{A}}_\theta}{\partial^{I(t_i)}_{G}\tilde{\boldsymbol{\theta}}} &= \mathbf{m}(t)\mathbf{V}^{-1}\frac{\partial\Delta\tilde{\boldsymbol{\phi}}}{\partial^{I(t_i)}_{G}\tilde{\boldsymbol{\theta}}} && \text{(F.57)}
\end{aligned}$$

Next, we perturb $\Delta\boldsymbol{\phi}$:

$$\Delta\hat{\boldsymbol{\phi}} + \Delta\tilde{\boldsymbol{\phi}} = \begin{bmatrix} \Delta\hat{\boldsymbol{\phi}}_1 + \Delta\tilde{\boldsymbol{\phi}}_1 \\ \Delta\hat{\boldsymbol{\phi}}_2 + \Delta\tilde{\boldsymbol{\phi}}_2 \\ \vdots \\ \Delta\hat{\boldsymbol{\phi}}_n + \Delta\tilde{\boldsymbol{\phi}}_n \end{bmatrix} \tag{F.58}$$

$$\Delta\hat{\boldsymbol{\phi}}_k + \Delta\tilde{\boldsymbol{\phi}}_k = \text{Log}\left(\text{Exp}\left(-{}^{I(t_k)}_{G}\tilde{\boldsymbol{\theta}}\right){}^{I(t_k)}_{G}\hat{\mathbf{R}}^{G}_{I(t_0)}\hat{\mathbf{R}}\right) \tag{F.59}$$

$$\approx \text{Log}\left(\text{Exp}\left(\Delta\hat{\boldsymbol{\phi}}_k - \mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_k\right){}^{I(t_k)}_{G}\tilde{\boldsymbol{\theta}}\right)\right) \tag{F.60}$$

$$= \Delta\hat{\boldsymbol{\phi}}_k - \mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_k\right){}^{I(t_k)}_{G}\tilde{\boldsymbol{\theta}} \tag{F.61}$$

$$\Delta\hat{\boldsymbol{\phi}}_k + \Delta\tilde{\boldsymbol{\phi}}_k = \text{Log}\left({}^{I(t_k)}_{G}\hat{\mathbf{R}}^{G}_{I(t_0)}\hat{\mathbf{R}}\text{Exp}\left({}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}\right)\right) \tag{F.62}$$

$$= \text{Log}\left(\text{Exp}\left({}^{I(t_k)}_{I(t_0)}\hat{\mathbf{R}}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}\right){}^{I(t_k)}_{I(t_0)}\hat{\mathbf{R}}\right) \tag{F.63}$$

$$\approx \text{Log}\left(\text{Exp}\left(\Delta\hat{\boldsymbol{\phi}}_k + \mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_k\right){}^{I(t_k)}_{I(t_0)}\hat{\mathbf{R}}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}\right)\right) \tag{F.64}$$

$$\Delta\hat{\boldsymbol{\phi}}_k + \mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_k\right){}^{I(t_k)}_{I(t_0)}\hat{\mathbf{R}}^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}} \tag{F.65}$$

Thus we have:

$$\frac{\partial\Delta\tilde{\boldsymbol{\phi}}}{\partial^{I(t_0)}_{G}\tilde{\boldsymbol{\theta}}} = \begin{bmatrix} -\mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_1\right) \\ -\mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_2\right) \\ \vdots \\ -\mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_n\right) \end{bmatrix} \tag{F.66}$$

$$\frac{\partial\tilde{\boldsymbol{\theta}}}{\partial^{I(t_k)}_{G}\tilde{\boldsymbol{\theta}}} = \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{J}_l^{-1}\left(\Delta\hat{\boldsymbol{\phi}}_k\right){}^{I(t_k)}_{I(t_0)}\hat{\mathbf{R}} \\ \vdots \\ \mathbf{0} \end{bmatrix} \tag{F.67}$$

In the case that we are estimating a time offset, that is $t = t_m + {}^{C_i}t_{C_b}$, then we will additionally have:

$$\frac{\partial^{I(t)}_{G}\tilde{\boldsymbol{\theta}}}{\partial^{C_i}\tilde{t}_{C_b}} = -\mathbf{J}_l\left(\hat{\mathbf{A}}_\theta\right)\frac{\partial\tilde{\mathbf{A}}_\theta}{\partial^{C_i}\tilde{t}_{C_b}} \tag{F.68}$$

$$\frac{\partial\tilde{\mathbf{A}}_\theta}{\partial^{C_i}\tilde{t}_{C_b}} = \frac{\partial\mathbf{m}(t)}{\partial^{C_i}\tilde{t}_{C_b}}\hat{\mathbf{a}}_\theta \tag{F.69}$$

$$\frac{\partial\mathbf{m}(t)}{\partial^{C_i}\tilde{t}_{C_b}} = \begin{bmatrix}\mathbf{I}_3 & 2\Delta\hat{t}\mathbf{I}_3 & \cdots & n\Delta\hat{t}^{n-1}\mathbf{I}_3\end{bmatrix} \tag{F.70}$$

Next we look at the position interpolation:

$$\frac{\partial^{G}\tilde{\mathbf{p}}_{I(t)}}{\partial^{G}\tilde{\mathbf{p}}_{I(t_0)}} = \mathbf{I}_3 + \mathbf{m}(t)\mathbf{V}^{-1}\frac{\partial\Delta\mathbf{p}}{\partial^{G}\tilde{\mathbf{p}}_{I(t_0)}} \tag{F.71}$$

$$\frac{\partial \Delta \mathbf{p}}{\partial {}^{G}\tilde{\mathbf{p}}_{I(t_0)}} = \begin{bmatrix} -\mathbf{I}_3 \\ -\mathbf{I}_3 \\ \vdots \\ -\mathbf{I}_3 \end{bmatrix} \tag{F.72}$$

$$\frac{\partial {}^{G}\tilde{\mathbf{p}}_{I(t)}}{\partial {}^{G}\tilde{\mathbf{p}}_{I(t_k)}} = \mathbf{m}(t)\mathbf{V}^{-1}\frac{\partial \Delta \mathbf{p}}{\partial {}^{G}\tilde{\mathbf{p}}_{I(t_k)}} \tag{F.73}$$

$$\frac{\partial \Delta \mathbf{p}}{\partial {}^{G}\tilde{\mathbf{p}}_{I(t_k)}} = \begin{bmatrix} \mathbf{0}_3 \\ \vdots \\ \mathbf{I}_3 \\ \vdots \\ \mathbf{0}_3 \end{bmatrix} \tag{F.74}$$

$$\frac{\partial {}^{G}\tilde{\mathbf{p}}_{I(t)}}{\partial {}^{C_i}\tilde{t}_{C_b}} = \frac{\partial \mathbf{m}(t)}{\partial {}^{C_i}\tilde{t}_{C_b}}\hat{\mathbf{a}}_p \tag{F.75}$$

Appendix G

OBSERVABILITY ANALYSIS FOR VILTT

G.1 Problem Formulation

In this section we present our observability analysis for the proposed VILTT system using different motion models.

G.1.1 Feature Measurement - Static Environmental

As the moving IMU sensor platform moves through the environment, static environmental feature tracks are collected and used to update the state. For simplicity, we assume the camera-to-IMU extrinsic calibration is identity, so that the static feature measurement can be written as:

$$\mathbf{z}_1 = \mathbf{h}({}^G\mathbf{p}_{fs}, \mathbf{x}_I) + \mathbf{n}_{f1} \tag{G.1}$$

$$= \begin{bmatrix} \frac{{}^I x_{fs}}{{}^I z_{fs}} \\ \frac{{}^I y_{fs}}{{}^I z_{fs}} \end{bmatrix} + \mathbf{n}_{f1} \tag{G.2}$$

where ${}^I\mathbf{p}_{fs} = \begin{bmatrix} {}^I x_{fs} & {}^I y_{fs} & {}^I z_{fs} \end{bmatrix}^\top$ can be written as:

$${}^I\mathbf{p}_{fs} = {}^I_G\mathbf{R} \left({}^G\mathbf{p}_{fs} - {}^G\mathbf{p}_I \right) \tag{G.3}$$

G.1.2 Feature Measurement - Target Non-Representative

The moving IMU sensor platform also tracks features that reside on the moving target but are not of the representative point/pose of the target. We represent these target features in the local target frame of reference and write them as the following:

$$\mathbf{z}_2 = \mathbf{h}(\mathbf{x}_I, \mathbf{x}_T, {}^T\mathbf{p}_{ft}) + \mathbf{n}_{f2} \tag{G.4}$$

$$= \begin{bmatrix} \frac{{}^I x_{ft}}{{}^I z_{ft}} \\ \frac{{}^I y_{ft}}{{}^I z_{ft}} \end{bmatrix} + \mathbf{n}_{f2} \tag{G.5}$$

where ${}^I\mathbf{p}_{ft} = \begin{bmatrix} {}^I x_{ft} & {}^I y_{ft} & {}^I z_{ft} \end{bmatrix}^\top$ can be written as:

$$\begin{aligned} {}^I\mathbf{p}_{ft} &= {}^I_T\mathbf{R}\left({}^T\mathbf{p}_{ft} - {}^T\mathbf{p}_I\right) && \text{(G.6)} \\ &= {}^I_G\mathbf{R}\,{}^T_G\mathbf{R}^\top\left({}^T\mathbf{p}_{ft} - {}^T_G\mathbf{R}\left({}^G\mathbf{p}_I - {}^G\mathbf{p}_T\right)\right) && \text{(G.7)} \\ &= {}^I_G\mathbf{R}\left({}^T_G\mathbf{R}^\top {}^T\mathbf{p}_{ft} - \left({}^G\mathbf{p}_I - {}^G\mathbf{p}_T\right)\right) && \text{(G.8)} \end{aligned}$$

G.1.3 Feature Measurement - Target Representative

A single feature is taken to be the "representative point" that the pose of the rigid-body is estimated to be at. Visual bearing measurements of this representative point are a direct measurements of the target's position, thus we can write the following:

$$\begin{aligned} \mathbf{z}_3 &= \mathbf{h}(\mathbf{x}_T, \mathbf{x}_I) + \mathbf{n}_3 && \text{(G.9)} \\ &= \begin{bmatrix} \frac{{}^I x_T}{{}^I z_T} \\ \frac{{}^I y_T}{{}^I z_T} \end{bmatrix} + \mathbf{n}_3 && \text{(G.10)} \end{aligned}$$

Note that ${}^I\mathbf{p}_T = \begin{bmatrix} {}^I x_T & {}^I y_T & {}^I z_T \end{bmatrix}^\top$ can be described as:

$${}^I\mathbf{p}_T = {}^I_G\mathbf{R}({}^G\mathbf{p}_T - {}^G\mathbf{p}_I) \tag{G.11}$$

G.2 Observability Analysis

Following the methodology of Hesch et al. [58], we perform the observability analysis for the linearized system. The observability matrix $\mathbf{M}(\mathbf{x})$ can constructed as:

$$\mathbf{M}(\mathbf{x}) = \begin{bmatrix} \mathbf{H}_{\mathbf{x}0}\mathbf{\Phi}(0,0) \\ \mathbf{H}_{\mathbf{x}1}\mathbf{\Phi}(1,0) \\ \mathbf{H}_{\mathbf{x}2}\mathbf{\Phi}(2,0) \\ \vdots \\ \mathbf{H}_{\mathbf{x}k}\mathbf{\Phi}(k,0) \end{bmatrix} \tag{G.12}$$

where $\mathbf{H}_{\mathbf{x}k}$ represents the system measurement Jacobians at time step k, $\mathbf{\Phi}(k,0)$ represents the state transition matrix from time 0 to time k. The right nullspace $\mathbf{N}$ of the observability matrix $\mathbf{M}(\mathbf{x})$ spans the unobservable directions of the system. For each of the following models, we first compute the measurement Jacobian matrix $\mathbf{H}_{\mathbf{x}}$, the state transition matrix $\mathbf{\Phi}^{(i)}$, and observability matrix $\mathbf{M}^{(i)}$. From this we find the nullspace and offer their geometric interpretation. The procedure can be found in Fig. G.1.

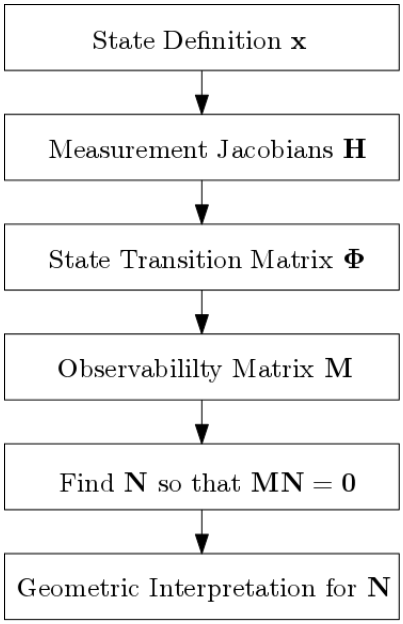


Figure G.1: Visual diagram of the steps needed to perform observability analysis. For each of the target models we perform these steps to find the unobservable directions of the VILTT system.

G.3 Observability Analysis - Motion Model 1

G.3.1 Measurement Jacobians

According to previous sections, we have 3 measurement models: the measurement to static features in the environment $\mathbf{z}_1$, the measurement to the features in target body $\mathbf{z}_2$, and the measurement to the representative point of the rigid body $\mathbf{z}_3$.

Therefore, the total measurement Jacobians can be written as:

$$\mathbf{H}_{\mathbf{x}} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \\ \mathbf{H}_3 \end{bmatrix} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}} \\ \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} \\ \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}} \end{bmatrix} \tag{G.13}$$

where $\mathbf{H}_i$, $i = 1, 2, 3$, represent the measurement Jacobians of the 3 above measurement models. We first compute the Jacobians $\mathbf{H}_1$ as:

$$\mathbf{H}_1 = \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^G\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_T^{(1)}} & \frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^T\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.14}$$

where we have:

$$\mathbf{H}_{C1} = \frac{1}{{}^I\hat{z}_{fs}^2} \begin{bmatrix} {}^I\hat{z}_{fs} & 0 & -{}^I\hat{x}_{fs} \\ 0 & {}^I\hat{z}_{fs} & -{}^I\hat{y}_{fs} \end{bmatrix} \tag{G.15}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C1} {}^I_G\hat{\mathbf{R}} \begin{bmatrix} \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_I \rfloor {}^G_I\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & -\mathbf{I}_3 \end{bmatrix} \tag{G.16}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^G\tilde{\mathbf{p}}_{fs}} = \mathbf{H}_{C1} {}^I_G\hat{\mathbf{R}} \mathbf{I}_3 \tag{G.17}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_T^{(1)}} = \mathbf{0}_{3\times 12} \tag{G.18}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^T\tilde{\mathbf{p}}_{fs}} = \mathbf{0}_3 \tag{G.19}$$

The Jacobians $\mathbf{H}_2$ can be written as:

$$\mathbf{H}_2 = \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^G\tilde{\mathbf{p}}_{ft}} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}_T^{(1)}} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^T\tilde{\mathbf{p}}_{fs}} \end{bmatrix} \tag{G.20}$$

where we have:

$$\mathbf{H}_{C2} = \frac{1}{{}^I\hat{z}_{ft}^2} \begin{bmatrix} {}^I\hat{z}_{ft} & 0 & -{}^I\hat{x}_{ft} \\ 0 & {}^I\hat{z}_{ft} & -{}^I\hat{y}_{ft} \end{bmatrix} \tag{G.21}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C2} {}^I_G\hat{\mathbf{R}} \begin{bmatrix} \lfloor \left({}^T_G\hat{\mathbf{R}}^{\top} {}^T\hat{\mathbf{p}}_{ft} - \left({}^G\hat{\mathbf{p}}_I - {}^G\hat{\mathbf{p}}_T \right) \right) \rfloor {}^G_I\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & -\mathbf{I}_3 \end{bmatrix} \tag{G.22}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^G\tilde{\mathbf{p}}_{ft}} = \mathbf{0}_3 \tag{G.23}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}_T^{(1)}} = \mathbf{H}_{C2} {}^I_G\hat{\mathbf{R}} \begin{bmatrix} -{}^T_G\hat{\mathbf{R}}^{\top} \lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix} \tag{G.24}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^T\tilde{\mathbf{p}}_{ft}} = \mathbf{H}_{C2} {}^I_G\hat{\mathbf{R}} {}^T_G\hat{\mathbf{R}}^\top \tag{G.25}$$

The measurement Jacobians $\mathbf{H}_3$ can be written as:

$$\mathbf{H}_3 = \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^G\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}_T^{(1)}} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^T\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.26}$$

where we have:

$$\mathbf{H}_{C3} = \frac{1}{{}^I\hat{z}_T^2} \begin{bmatrix} {}^I\hat{z}_T & 0 & -{}^I\hat{x}_T \\ 0 & {}^I\hat{z}_T & -{}^I\hat{y}_T \end{bmatrix} \tag{G.27}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C3} {}^I_G\hat{\mathbf{R}} \begin{bmatrix} \lfloor ({}^G\hat{\mathbf{p}}_T - {}^G\hat{\mathbf{p}}_I) \rfloor {}^G_I\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & \mathbf{0}_3 \end{bmatrix} \tag{G.28}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^G\tilde{\mathbf{p}}_{fs}} = \mathbf{0}_3 \tag{G.29}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}_T^{(1)}} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix} \tag{G.30}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^T\tilde{\mathbf{p}}_{ft}} = \mathbf{0}_3 \tag{G.31}$$

G.3.2 State Transition Matrix

The total system state transition matrix can be written as:

$$\mathbf{\Phi}^{(1)} = \begin{bmatrix} \mathbf{\Phi}_I & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_{fs} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_T^{(1)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_{ft} \end{bmatrix} \tag{G.32}$$

where $\mathbf{\Phi}_I$, $\mathbf{\Phi}_{fs}$, $\mathbf{\Phi}_T^{(1)}$ and $\mathbf{\Phi}_{ft}$ represent the state transition matrix for $\mathbf{x}_I$, ${}^G\mathbf{p}_{fs}$, $\mathbf{x}_T^{(1)}$ and ${}^T\mathbf{p}_{ft}$, respectively. Note that $\mathbf{\Phi}_{fs} = \mathbf{I}_3$, $\mathbf{\Phi}_{ft} = \mathbf{I}_3$ and $\mathbf{\Phi}_I$ can be written from [58] as:

$$\mathbf{\Phi}_I(k,0) = \begin{bmatrix} \mathbf{\Phi}_{11} & \mathbf{\Phi}_{12} & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{\Phi}_{31} & \mathbf{\Phi}_{32} & \mathbf{I}_3 & \mathbf{\Phi}_{34} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{\Phi}_{51} & \mathbf{\Phi}_{52} & \mathbf{\Phi}_{53} & \mathbf{\Phi}_{54} & \mathbf{I}_3 \end{bmatrix} \tag{G.33}$$

where the related items in the matrix can be written as:

$$\begin{aligned}
\mathbf{\Phi}_{I11} &= {}^{I_k}_{I_0}\mathbf{R} && \text{(G.34)}\\
\mathbf{\Phi}_{I31} &= -\lfloor \left({}^{G}\mathbf{v}_{I_k} - {}^{G}\mathbf{v}_{I_0}\right) + {}^{G}\mathbf{g}\Delta t_k \times \rfloor {}^{G}_{I_0}\mathbf{R} && \text{(G.35)}\\
\mathbf{\Phi}_{I51} &= \lfloor {}^{G}\mathbf{p}_{I_0} + {}^{G}\mathbf{v}_{I_0}\Delta t_k - \frac{1}{2}{}^{G}\mathbf{g}\Delta t_k^2 - {}^{G}\mathbf{p}_{I_k} \times \rfloor {}^{G}_{I_0}\mathbf{R} && \text{(G.36)}\\
\mathbf{\Phi}_{I12} &= -\int_{t_0}^{t_k} {}^{I_\tau}_{I_k}\mathbf{R}^\top \mathbf{d}\tau && \text{(G.37)}\\
\mathbf{\Phi}_{I32} &= \int_{t_0}^{t_k} {}^{I_s}_{G}\mathbf{R}^\top \lfloor {}^{I_s}\mathbf{a} \times \rfloor \int_{t_0}^{s} {}^{I_\tau}_{I_s}\mathbf{R}^\top \mathbf{d}\tau \mathbf{d}s && \text{(G.38)}\\
\mathbf{\Phi}_{I52} &= \int_{t_0}^{t_k} \int_{t_0}^{\theta} {}^{I_s}_{G}\mathbf{R}^\top \lfloor {}^{I_s}\mathbf{a} \times \rfloor \int_{t_0}^{t_s} {}^{I_\tau}_{I_s}\mathbf{R}^\top \mathbf{d}\tau \mathbf{d}s \mathbf{d}\theta && \text{(G.39)}\\
\mathbf{\Phi}_{I53} &= \mathbf{I}_3 \Delta t_k && \text{(G.40)}\\
\mathbf{\Phi}_{I34} &= -\int_{t_0}^{t_k} {}^{I_\tau}_{G}\mathbf{R}^\top \mathbf{d}\tau && \text{(G.41)}\\
\mathbf{\Phi}_{I54} &= -\int_{t_0}^{t_k} \int_{t_0}^{t_s} {}^{I_\tau}_{G}\mathbf{R}^\top \mathbf{d}\tau \mathbf{d}s && \text{(G.42)}
\end{aligned}$$

Now we need to solve for the remaining target state transition matrix $\mathbf{\Phi}_T^{(1)}$. The linearized system for the target motion model can be written as:

$$\dot{\tilde{x}}_T^{(1)} = \begin{bmatrix} {}^{T}_{G}\dot{\tilde{\boldsymbol{\theta}}} \\ {}^{T}\dot{\tilde{\boldsymbol{\omega}}}_T \\ {}^{G}\dot{\tilde{\mathbf{p}}}_T \\ {}^{G}\dot{\tilde{\mathbf{v}}}_T \end{bmatrix} \simeq \begin{bmatrix} -\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0} \end{bmatrix} \begin{bmatrix} {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T}\tilde{\boldsymbol{\omega}}_T \\ {}^{G}\tilde{\mathbf{p}}_T \\ {}^{G}\tilde{\mathbf{v}}_T \end{bmatrix} + \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \mathbf{n}_{t\omega} \\ \mathbf{n}_{tv} \end{bmatrix} \qquad \text{(G.43)}$$

Thus, the state transition $\mathbf{\Phi}_T^{(1)}$ evolution can be written as:

$$\dot{\mathbf{\Phi}}_T^{(1)} = \begin{bmatrix} -\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}_{T11} & \mathbf{\Phi}_{T12} & \mathbf{\Phi}_{T13} & \mathbf{\Phi}_{T14} \\ \mathbf{\Phi}_{T21} & \mathbf{\Phi}_{T22} & \mathbf{\Phi}_{T23} & \mathbf{\Phi}_{T24} \\ \mathbf{\Phi}_{T31} & \mathbf{\Phi}_{T32} & \mathbf{\Phi}_{T33} & \mathbf{\Phi}_{T34} \\ \mathbf{\Phi}_{T41} & \mathbf{\Phi}_{T42} & \mathbf{\Phi}_{T43} & \mathbf{\Phi}_{T44} \end{bmatrix} \qquad \text{(G.44)}$$

Then, the target state transition matrix can be solved as:

$$\mathbf{\Phi}_T^{(1)} = \begin{bmatrix} \mathbf{\Phi}_{T11} & \mathbf{\Phi}_{T12} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{\Phi}_{T34} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{G.45}$$

where we have:

$$\dot{\mathbf{\Phi}}_{T11} = -\lfloor {}^T\hat{\omega}_T \times \rfloor \dot{\mathbf{\Phi}}_{T11} + \mathbf{I} \tag{G.46}$$

$$\text{Exp}\left(\int_{t_o}^{t_k} {}^T\hat{\omega}_T d\tau\right) \dot{\mathbf{\Phi}}_{T11} + \text{Exp}\left(\int_{t_o}^{t_k} {}^T\hat{\omega}_T d\tau\right) \lfloor {}^T\hat{\omega}_T \times \rfloor \mathbf{\Phi}_{T11} = \text{Exp}\left(\int_{t_o}^{t_k} {}^T\hat{\omega}_T d\tau\right) \tag{G.47}$$

$$\frac{d}{dt}\left(\text{Exp}\left(\int_{t_o}^{t_t au} {}^T\hat{\omega}_T d\tau\right) \mathbf{\Phi}_{T11}\right) = {}^{T_0}_{T_\tau}\mathbf{R} \tag{G.48}$$

$${}^{T_0}_{T_k}\mathbf{R}\mathbf{\Phi}_{T11} - \mathbf{0} = \int_{t_0}^{t_k} {}^{T_0}_{T_\tau}\mathbf{R} d\tau \tag{G.49}$$

Proceeding in this fashion yields:

$$\mathbf{\Phi}_{T11} = {}^{T_k}_{T_0}\mathbf{R} \tag{G.50}$$

$$\mathbf{\Phi}_{T12} = \int_{t_0}^{t} {}^{T_k}_{T_\tau}\mathbf{R} d\tau \tag{G.51}$$

$$\mathbf{\Phi}_{T34} = \mathbf{I}_3 \Delta t_k \tag{G.52}$$

Therefore, the target state transition matrix can be written as:

$$\mathbf{\Phi}_T^{(1)} = \begin{bmatrix} {}^{T_k}_{T_0}\mathbf{R} & \int_{t_0}^{t} {}^{T_k}_{T_\tau}\mathbf{R} d\tau & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{I}_3 \Delta t_k \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{G.53}$$

G.3.3 Observability Matrix

With abuse of notation, the k'th block of the observability matrix can be written as:

$$\mathbf{M}_k^{(1)} = \mathbf{H}_{\mathbf{x}k}\mathbf{\Phi}^{(1)}(k,0) = \mathbf{H}_{\mathbf{x}k}\begin{bmatrix}\mathbf{\Phi}_I & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{\Phi}_{fs} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_T^{(1)} & \mathbf{0}\\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_{ft}\end{bmatrix} \tag{G.54}$$

$$= \begin{bmatrix}\mathbf{H}_{C1}{}_G^I\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{H}_{C2}{}_G^I\hat{\mathbf{R}} & \mathbf{0}_3\\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{H}_{C3}{}_G^I\hat{\mathbf{R}}\end{bmatrix}\times$$

$$\begin{bmatrix}\mathbf{M}_{11} & \mathbf{M}_{12} & \mathbf{M}_{13} & \mathbf{M}_{14} & \mathbf{M}_{15} & \mathbf{M}_{16} & \mathbf{M}_{17} & \mathbf{M}_{18} & \mathbf{M}_{19} & \mathbf{M}_{1,10} & \mathbf{M}_{1,11}\\ \mathbf{M}_{21} & \mathbf{M}_{22} & \mathbf{M}_{23} & \mathbf{M}_{24} & \mathbf{M}_{25} & \mathbf{M}_{26} & \mathbf{M}_{27} & \mathbf{M}_{28} & \mathbf{M}_{29} & \mathbf{M}_{2,10} & \mathbf{M}_{2,11}\end{bmatrix} \tag{G.55}$$

The first row of this matrix can be denoted as:

$$\mathbf{M}_{11} = \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_{I_k}\rfloor_{I_k}^G\hat{\mathbf{R}}\mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} = \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_{I_0} - {}^G\hat{\mathbf{v}}_{I_0}\Delta t_k + \frac{1}{2}{}^G\mathbf{g}\Delta t_k^2\rfloor_{I_0}^G\hat{\mathbf{R}} \tag{G.56}$$

$$\mathbf{M}_{12} = \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_{I_k}\rfloor_{I_k}^G\hat{\mathbf{R}}\mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.57}$$

$$\mathbf{M}_{13} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3\Delta t_k \tag{G.58}$$

$$\mathbf{M}_{14} = -\mathbf{\Phi}_{I54} \tag{G.59}$$

$$\mathbf{M}_{15} = -\mathbf{I}_3 \tag{G.60}$$

$$\mathbf{M}_{16} = \mathbf{I}_3 \tag{G.61}$$

$$\mathbf{M}_{17} = \mathbf{M}_{18} = \mathbf{M}_{19} = \mathbf{M}_{1,10} = \mathbf{M}_{1,11} = \mathbf{0}_3 \tag{G.62}$$

The second row of this matrix can be described as:

$$\mathbf{M}_{21} = \lfloor {}_{T_k}^G\hat{\mathbf{R}}^T\hat{\mathbf{p}}_{ft} - {}^G\hat{\mathbf{p}}_{I_k} + {}^G\hat{\mathbf{p}}_{T_k}\rfloor_{I_k}^G\hat{\mathbf{R}}\mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} \tag{G.63}$$

$$= \lfloor {}_{T_k}^G\hat{\mathbf{R}}^T\hat{\mathbf{p}}_{\mathbf{f}} + {}^G\hat{\mathbf{p}}_{T_k} - {}^G\hat{\mathbf{p}}_{I_0} - {}^G\hat{\mathbf{v}}_{I_0}\Delta t_k + \frac{1}{2}{}^G\mathbf{g}\Delta t_k^2\rfloor_{I_0}^G\hat{\mathbf{R}} \tag{G.64}$$

$$\mathbf{M}_{22} = \lfloor {}_{T_k}^G\hat{\mathbf{R}}^T\hat{\mathbf{p}}_{ft} - {}^G\hat{\mathbf{p}}_{I_k} + {}^G\hat{\mathbf{p}}_{T_k}\rfloor_I^G\hat{\mathbf{R}}\mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.65}$$

$$\mathbf{M}_{23} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3 \Delta t_k \tag{G.66}$$

$$\mathbf{M}_{24} = -\mathbf{\Phi}_{I54} \tag{G.67}$$

$$\mathbf{M}_{25} = -\mathbf{I}_3 \tag{G.68}$$

$$\mathbf{M}_{26} = \mathbf{0}_3 \tag{G.69}$$

$$\mathbf{M}_{27} = -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{\Phi}_{T11} = \lfloor -{}^{G}_{T_k}\mathbf{R}^T \mathbf{p}_{ft} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \tag{G.70}$$

$$\mathbf{M}_{28} = -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{\Phi}_{T12} \tag{G.71}$$

$$\mathbf{M}_{29} = \mathbf{I}_3 \tag{G.72}$$

$$\mathbf{M}_{2,10} = \mathbf{\Phi}_{T34} = \mathbf{I}_3 \Delta t_k \tag{G.73}$$

$$\mathbf{M}_{2,11} = {}^{G}_{T_k}\hat{\mathbf{R}} \tag{G.74}$$

The third row of this matrix can be described as:

$$\mathbf{M}_{31} = \lfloor -{}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor {}^{G}_{I_k}\hat{\mathbf{R}} \mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} = \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{I_0} - {}^{G}\hat{\mathbf{v}}_{I_0} \Delta t_k + \frac{1}{2} {}^{G}\mathbf{g} \Delta t_k^2 \rfloor {}^{G}_{I_0}\mathbf{R} \tag{G.75}$$

$$\mathbf{M}_{32} = \lfloor -{}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor {}^{G}_{I}\mathbf{R} \mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.76}$$

$$\mathbf{M}_{33} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3 \Delta t_k \tag{G.77}$$

$$\mathbf{M}_{34} = -\mathbf{\Phi}_{I54} \tag{G.78}$$

$$\mathbf{M}_{35} = -\mathbf{I}_3 \tag{G.79}$$

$$\mathbf{M}_{36} = \mathbf{0}_3 \tag{G.80}$$

$$\mathbf{M}_{37} = \mathbf{0}_3 \tag{G.81}$$

$$\mathbf{M}_{38} = \mathbf{0} \tag{G.82}$$

$$\mathbf{M}_{39} = \mathbf{I}_3 \tag{G.83}$$

$$\mathbf{M}_{3,10} = \mathbf{\Phi}_{T34} = \mathbf{I}_3 \Delta t_k \tag{G.84}$$

$$\mathbf{M}_{3,11} = \mathbf{0}_3 \tag{G.85}$$

Therefore, we can rewrite the k's block of the observability matrix as:

$$\mathbf{M}_k^{(1)} = \mathbf{H}_{\mathbf{x}k} \mathbf{\Phi}^{(1)}(k, 0) \tag{G.86}$$

$$= \begin{bmatrix} \mathbf{H}_{C1}{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{H}_{C2}{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{H}_{C3}{}^{I}_{G}\hat{\mathbf{R}} \end{bmatrix} \times$$

$$\begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} & -\mathbf{I}_3\Delta t_k & \mathbf{M}_{14} & -\mathbf{I}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{M}_{21} & \mathbf{M}_{22} & -\mathbf{I}_3\Delta t_k & \mathbf{M}_{24} & -\mathbf{I}_3 & \mathbf{0}_3 & \lfloor -{}^{G}_{T_k}\hat{\mathbf{R}}^T\hat{\mathbf{p}}_{\mathbf{f}}\rfloor {}^{G}_{T_0}\hat{\mathbf{R}} & \mathbf{M}_{28} & \mathbf{I}_3 & \mathbf{I}_3\Delta t_k & {}^{G}_{T_k}\hat{\mathbf{R}} \\ \mathbf{M}_{31} & \mathbf{M}_{32} & -\mathbf{I}_3\Delta t_k & \mathbf{M}_{24} & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{I}_3\Delta t_k & \mathbf{0}_3 \end{bmatrix} \tag{G.87}$$

Note that given the assumption that constant global velocity ${}^{G}\hat{\mathbf{v}}_T$, we consider the ideal noise free case in observability analysis, the ${}^{G}\hat{\mathbf{p}}_{T_k}$ can be written as:

$${}^{G}\hat{\mathbf{p}}_{T_k} = {}^{G}\hat{\mathbf{p}}_{T_0} + {}^{G}\hat{\mathbf{v}}_T\Delta t_k \tag{G.88}$$

Based on the constant ${}^{T}\hat{\boldsymbol{\omega}}_T$ assumption, we can have that:

$$\boldsymbol{\Phi}_{T12}\lfloor {}^{T}\hat{\boldsymbol{\omega}}\rfloor = \int_{t_0}^{t} {}^{T_k}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T\rfloor = \mathbf{I}_3 - {}^{T_k}_{T_0}\hat{\mathbf{R}} \tag{G.89}$$

$${}^{G}_{T_k}\hat{\mathbf{R}}^T\hat{\boldsymbol{\omega}}_T = {}^{G}_{T_0}\hat{\mathbf{R}}^{T_0}_{T_k}\hat{\mathbf{R}}^T\boldsymbol{\omega} = {}^{G}_{T_0}\hat{\mathbf{R}}\left(\mathbf{I}_3 + \frac{\sin|\boldsymbol{\theta}(\Delta t_k)|}{|{}^{T}\hat{\boldsymbol{\omega}}_T|}\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T\rfloor + \frac{1-\cos|\boldsymbol{\theta}(\Delta t_k)|}{|{}^{T}\hat{\boldsymbol{\omega}}_T|^2}\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T\rfloor^2\right){}^{T}\hat{\boldsymbol{\omega}}_T = {}^{G}_{T_0}\hat{\mathbf{R}}^T\hat{\boldsymbol{\omega}}_T \tag{G.90}$$

The unobservable directions $\mathbf{N}^{(1)}$ span the right null space of the observability matrix $\mathbf{M}^{(1)}$, that is: $\mathbf{M}^{(1)}\mathbf{N}^{(1)} = \mathbf{0}$. Based on the derivation of the observability matrix, we

can have the unobservable directions as:

$$\mathbf{N}^{(1)} = \begin{bmatrix} \mathbf{N}_1^{(1)} & \mathbf{N}_{2:4}^{(1)} & \mathbf{N}_{{}_G^T\mathbf{R}}^{(1)} \end{bmatrix} = \begin{bmatrix} {}_G^{I_0}\hat{\mathbf{R}}\,{}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{v}}_{I_0} \rfloor {}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{p}}_{I_0} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{p}}_{fs} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_3 \\ -{}_G^{T_0}\hat{\mathbf{R}}\,{}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \lfloor {}^T\hat{\boldsymbol{\omega}}_T \rfloor \\ -\lfloor {}^G\hat{\mathbf{p}}_{T_0} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{v}}_{T_0} \rfloor {}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor \end{bmatrix} \tag{G.91}$$

Note that the $\mathbf{N}_{1:4}^{(1)}$ relate to the global yaw and the global IMU position. $\mathbf{N}_{{}_G^T\mathbf{R}}^{(1)}$ relate to target body orientation.

But, if without the direct target representative point measurement (due to occlusion), we will have one additional unobservable direction related the representative point position as:

$$\mathbf{N}_{{}_G\mathbf{p}_T}^{(1)} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \left({}_{T_0}^{G}\hat{\mathbf{R}}\,{}^T\hat{\boldsymbol{\omega}}_T\right)^\top & \mathbf{0}_{1\times 3} & -\left({}^T\hat{\boldsymbol{\omega}}_T\right)^\top \end{bmatrix}^\top \tag{G.92}$$

G.3.4 Geometric Interpretation - Verification for $\mathbf{N}_{{}_G^T\mathbf{R}}$

If we disturb the target orientation by a small angle vector $\delta\boldsymbol{\phi}$, then we will have the disturbed target state parameters as:

$${}_G^{T'}\mathbf{R} = {}_T^{T'}\mathbf{R}\,{}_G^T\mathbf{R} \simeq (\mathbf{I}_3 - \lfloor \delta\boldsymbol{\phi} \rfloor)\,{}_G^T\mathbf{R} \tag{G.93}$$

$${}^{T'}\omega = {}_T^{T'}\mathbf{R}\,{}^T\omega \simeq (\mathbf{I}_3 - \lfloor \delta\boldsymbol{\phi} \rfloor)\,{}^T\boldsymbol{\omega} = {}^T\boldsymbol{\omega} + \lfloor {}^T\boldsymbol{\omega} \rfloor \delta\boldsymbol{\phi} \tag{G.94}$$

$${}^G\mathbf{p}_{T'} = {}^G\mathbf{p}_T \tag{G.95}$$

$${}^G\mathbf{v}_{T'} = {}^G\mathbf{v}_T \tag{G.96}$$

$${}^{T'}\mathbf{p}_{ft} = {}_T^{T'}\mathbf{R}\,{}^T\mathbf{p}_{ft} \simeq (\mathbf{I}_3 - \lfloor \delta\boldsymbol{\phi} \rfloor)\,{}^T\mathbf{p}_{ft} = {}^T\mathbf{p}_{ft} + \lfloor {}^T\mathbf{p}_{ft} \rfloor \delta\boldsymbol{\phi} \tag{G.97}$$

Then, we can have:

$$
\begin{aligned}
{}^{I}\mathbf{p}_{ft'} &= {}^{I}_{G}\mathbf{R}\left({}^{G}_{T'}\mathbf{R}{}^{T'}\mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T'}\right) = {}^{I}_{G}\mathbf{R}\left({}^{G}_{T}\mathbf{R}^{T}_{T'}\mathbf{R}^{T'}_{T}\mathbf{R}^{T}\mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T}\right) && \text{(G.98)}\\
&= {}^{I}\mathbf{p}_{ft} && \text{(G.99)}\\
{}^{I}\mathbf{p}_{T'} &= {}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T'} - {}^{G}\mathbf{p}_{I}\right) = {}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T} - {}^{G}\mathbf{p}_{I}\right) && \text{(G.100)}\\
&= {}^{I}\mathbf{p}_{T} && \text{(G.101)}
\end{aligned}
$$

Based on (G.1)(G.4)(G.9), even with the orientation disturbance $\delta\boldsymbol{\phi}$, the system will still have the same target feature and target representative point measurements. Therefore, even with measurements, the system still cannot distinguish the ambiguity caused by this disturbance.

Since we assume $\delta\boldsymbol{\phi}$ is a small perturbation, we can linearize the disturbed system state vector $\mathbf{x}'$ at current state estimate $\hat{\mathbf{x}}$. Thus, the related error states can be written as:

$$
\tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}'_{I} \\ {}^{G}\tilde{\mathbf{p}}'_{fs} \\ {}^{T'}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T'}\tilde{\boldsymbol{\omega}} \\ {}^{G}\tilde{\mathbf{p}}_{T'} \\ {}^{G}\tilde{\mathbf{v}}_{T'} \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_{I} \\ {}^{G}\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} + \delta\boldsymbol{\phi} \\ {}^{T}\tilde{\boldsymbol{\omega}} + \lfloor {}^{T}\hat{\boldsymbol{\omega}} \rfloor \delta\boldsymbol{\phi} \\ {}^{G}\tilde{\mathbf{p}}_{T} \\ {}^{G}\tilde{\mathbf{v}}_{T} \\ {}^{T}\tilde{\mathbf{p}}_{ft} + \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \delta\boldsymbol{\phi} \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times 3} \\ \mathbf{0}_{3} \\ \mathbf{I}_{3} \\ \lfloor {}^{T}\hat{\boldsymbol{\omega}} \rfloor \\ \mathbf{0}_{3} \\ \mathbf{0}_{3} \\ \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \end{bmatrix} \delta\boldsymbol{\phi} = \tilde{\mathbf{x}} + \mathbf{N}^{(1)}_{{}^{T}_{G}\mathbf{R}}\delta\boldsymbol{\phi} \qquad \text{(G.102)}
$$

It can be seen that the disturbance exactly follow the unobservable directions related to the target orientation.

G.3.5 Geometric Interpretation - Verification for $\mathbf{N}^{(1)}_{{}^{G}\mathbf{p}_{T}}$

Next, we verify $\mathbf{N}^{(1)}_{{}^{G}\mathbf{p}_{T}}$. Without the representative point measurement $\mathbf{z}_3$, if we disturb the target representative point position with δp along the direction of the rotation axis ${}^{G}_{T_0}\mathbf{R}^{T}\boldsymbol{\omega}$, then we can have the disturbed state vector as:

$$
{}^{T'}_{G}\mathbf{R} = {}^{T}_{G}\mathbf{R} \qquad \text{(G.103)}
$$

$$
\begin{aligned}
{}^{T'}\boldsymbol{\omega} &= {}^{T}\boldsymbol{\omega} && \text{(G.104)}\\
{}^{G}\mathbf{p}_{T'} &= {}^{G}\mathbf{p}_{T} + {}^{G}_{T_0}\mathbf{R}\,{}^{T}\boldsymbol{\omega}\delta p && \text{(G.105)}\\
\Rightarrow {}^{T}\mathbf{p}_{T'} &= {}^{T}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T'} - {}^{G}\mathbf{p}_{T}\right) = {}^{T}_{T_0}\mathbf{R}\,{}^{T}\boldsymbol{\omega}\delta p = {}^{T}\boldsymbol{\omega}\delta p && \text{(G.106)}\\
{}^{G}\mathbf{v}_{T'} &= {}^{G}\mathbf{v}_{T} && \text{(G.107)}\\
{}^{T'}\mathbf{p}_{ft} &= {}^{T'}_{T}\mathbf{R}\left({}^{T}\mathbf{p}_{ft} - {}^{T}\mathbf{p}_{T'}\right) = {}^{T}\mathbf{p}_{ft} - {}^{T}\boldsymbol{\omega}\delta p && \text{(G.108)}
\end{aligned}
$$

If we only have target feature measurements, then we have:

$$
\begin{aligned}
{}^{I}\mathbf{p}_{ft'} &= {}^{I}_{G}\mathbf{R}\left({}^{G}_{T'}\mathbf{R}\,{}^{T'}\mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T'}\right) && \text{(G.109)}\\
&= {}^{I}_{G}\mathbf{R}\left({}^{G}_{T}\mathbf{R}\,{}^{T}_{T'}\mathbf{R}\,{}^{T'}_{T}\mathbf{R}\left({}^{T}\mathbf{p}_{ft} - {}^{T}\boldsymbol{\omega}\delta p\right) - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T} + {}^{G}_{T_0}\mathbf{R}\,{}^{T}\boldsymbol{\omega}\delta p\right) && \text{(G.110)}\\
&= {}^{I}\mathbf{p}_{ft} && \text{(G.111)}
\end{aligned}
$$

Based on (G.1)(G.4), the system will still have the same target feature measurements. That means the ambiguity caused by the target position disturbance cannot be distinguished by the measurements. Since δp is assumed to be a small perturbation, if we linearized the disturbed system state $\mathbf{x}'$ at current state estimate $\hat{\mathbf{x}}$, then the related error states can be written as:

$$
\tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}'_I \\ {}^{G}\tilde{\mathbf{p}}'_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}}' \\ {}^{T'}\tilde{\boldsymbol{\omega}} \\ {}^{G}\tilde{\mathbf{p}}_{T'} \\ {}^{G}\tilde{\mathbf{v}}_{T'} \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_I \\ {}^{G}\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T}\tilde{\boldsymbol{\omega}}_T \\ {}^{G}\tilde{\mathbf{p}}_T + {}^{G}_{T_0}\hat{\mathbf{R}}\,{}^{T}\boldsymbol{\omega}_T\delta p \\ {}^{G}\tilde{\mathbf{v}}_T \\ {}^{T}\tilde{\mathbf{p}}_{ft} - {}^{T}\hat{\boldsymbol{\omega}}_T\delta p \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times 3} \\ \mathbf{0}_3 \\ \mathbf{0}_3 \\ \mathbf{0}_3 \\ {}^{G}_{T_0}\hat{\mathbf{R}}\,{}^{T}\boldsymbol{\omega}_T \\ \mathbf{0}_3 \\ -{}^{T}\hat{\boldsymbol{\omega}}_T \end{bmatrix}\delta p = \tilde{\mathbf{x}} + \mathbf{N}^{(1)}_{{}^{G}\mathbf{p}_T}\delta p \quad \text{(G.112)}
$$

Similarly, it can be seen that the disturbance exactly follow the unobservable directions related to the target representative point position.

G.4 Observability Analysis - Motion Model 2

G.4.1 Measurement Jacobians

With abuse of notation, the total system measurements for motion model 2 can still be written as:

$$\mathbf{H}_{\mathbf{x}} = \begin{bmatrix} \mathbf{H}_1 \\ \mathbf{H}_2 \\ \mathbf{H}_3 \end{bmatrix} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}} \\ \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} \\ \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}} \end{bmatrix} \tag{G.113}$$

Based on the motion model 2 for target rigid body, the measurement Jacobians for the static environment feature $\mathbf{H}_1$, the target feature $\mathbf{H}_2$ and the target representative point measurements $\mathbf{H}_3$ can be computed respectively. The Jacobians of the measurement to the environmental feature $\mathbf{H}_1$ can be written as:

$$\mathbf{H}_1 = \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^G\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_T^{(2)}} & \frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^T\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.114}$$

where we have:

$$\mathbf{H}_{C1} = \frac{1}{{}^I\hat{z}_{fs}^2} \begin{bmatrix} {}^I\hat{z}_{fs} & 0 & -{}^I\hat{x}_{fs} \\ 0 & {}^I\hat{z}_{fs} & -{}^I\hat{y}_{fs} \end{bmatrix} \tag{G.115}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C1} {}^I_G\hat{\mathbf{R}} \begin{bmatrix} \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_I \rfloor {}^G_I\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & -\mathbf{I}_3 \end{bmatrix} \tag{G.116}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^G\tilde{\mathbf{p}}_{fs}} = \mathbf{H}_{C1} {}^I_G\hat{\mathbf{R}} \mathbf{I}_3 \tag{G.117}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_T^{(2)}} = \mathbf{0}_{3\times 12} \tag{G.118}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^T\tilde{\mathbf{p}}_{ft}} = \mathbf{0}_3 \tag{G.119}$$

The target feature measurement Jacobians $\mathbf{H}_2$ can be written as:

$$\mathbf{H}_2 = \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^G\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}_T^{(2)}} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^T\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.120}$$

where we have:

$$\mathbf{H}_{C2} = \frac{1}{{}^I\hat{z}_{ft}^2} \begin{bmatrix} {}^I\hat{z}_{ft} & 0 & -{}^I\hat{x}_{ft} \\ 0 & {}^I\hat{z}_{ft} & -{}^I\hat{y}_{ft} \end{bmatrix} \tag{G.121}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C2} {}^{I}_{G}\hat{\mathbf{R}} \begin{bmatrix} \lfloor \left({}^{T}_{G}\hat{\mathbf{R}}^{\top} {}^{T}\hat{\mathbf{p}}_{ft} - \left({}^{G}\hat{\mathbf{p}}_I - {}^{G}\hat{\mathbf{p}}_T \right) \right) \rfloor {}^{G}_{I}\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & -\mathbf{I}_3 \end{bmatrix} \tag{G.122}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^{G}\tilde{\mathbf{p}}_{ft}} = \mathbf{0}_3 \tag{G.123}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}_T^{(2)}} = \mathbf{H}_{C2} {}^{I}_{G}\hat{\mathbf{R}} \begin{bmatrix} -{}^{T}_{G}\hat{\mathbf{R}}^{\top} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix} \tag{G.124}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} = \mathbf{H}_{C2} {}^{I}_{G}\hat{\mathbf{R}} {}^{T}_{G}\hat{\mathbf{R}}^{\top} \tag{G.125}$$

The target representative point measurement Jacobians $\mathbf{H}_3$ can be written as:

$$\mathbf{H}_3 = \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}_T^{(2)}} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.126}$$

where we have:

$$\mathbf{H}_{C3} = \frac{1}{{}^{I}\hat{z}_T^2} \begin{bmatrix} {}^{I}\hat{z}_T & 0 & -{}^{I}\hat{x}_T \\ 0 & {}^{I}\hat{z}_T & -{}^{I}\hat{y}_T \end{bmatrix} \tag{G.127}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C3} {}^{I}_{G}\hat{\mathbf{R}} \begin{bmatrix} \lfloor \left({}^{G}\hat{\mathbf{p}}_T - {}^{G}\hat{\mathbf{p}}_I \right) \rfloor {}^{G}_{I}\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & \mathbf{0}_3 \end{bmatrix} \tag{G.128}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} = \mathbf{0}_3 \tag{G.129}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}_T^{(2)}} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \end{bmatrix} \tag{G.130}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} = \mathbf{0}_3 \tag{G.131}$$

G.4.2 State Transition Matrix

The total system state transition matrix can be written as:

$$\mathbf{\Phi}^{(2)} = \begin{bmatrix} \mathbf{\Phi}_I & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_{fs} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_T^{(2)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_{ft} \end{bmatrix} \tag{G.132}$$

Note that $\mathbf{\Phi}_I$, $\mathbf{\Phi}_{fs}$ and $\mathbf{\Phi}_{ft}$ are the same as previous section. We still need to compute the target state transition matrix $\mathbf{\Phi}^{(2)}$ based on motion model 2. The linearized target motion system based on motion model 2 can be written as:

$$\dot{\tilde{\mathbf{x}}}_T^{(2)} = \begin{bmatrix} {}^T_G\dot{\tilde{\boldsymbol{\theta}}} \\ {}^T\dot{\tilde{\boldsymbol{\omega}}}_T \\ {}^G\dot{\tilde{\mathbf{p}}}_T \\ {}^T\dot{\tilde{\mathbf{v}}}_T \end{bmatrix} \simeq \begin{bmatrix} -\lfloor {}^T\hat{\boldsymbol{\omega}}_T \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -{}^G_T\hat{\mathbf{R}}\lfloor {}^T\hat{\mathbf{v}}_T \rfloor & \mathbf{0}_3 & \mathbf{0}_3 & {}^G_T\hat{\mathbf{R}} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0} \end{bmatrix} \begin{bmatrix} {}^T_G\tilde{\boldsymbol{\theta}} \\ {}^T\tilde{\boldsymbol{\omega}}_T \\ {}^G\tilde{\mathbf{p}}_T \\ {}^T\tilde{\mathbf{v}}_T \end{bmatrix} + \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \begin{bmatrix} \mathbf{n}_{t\omega} \\ \mathbf{n}_{tv} \end{bmatrix} \tag{G.133}$$

From the linearized system, we can get the state transition $\mathbf{\Phi}_T^{(2)}$ evolution as:

$$\dot{\mathbf{\Phi}}_T^{(2)} = \begin{bmatrix} -\lfloor {}^T\hat{\boldsymbol{\omega}}_T \rfloor & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -{}^G_T\hat{\mathbf{R}}\lfloor {}^T\hat{\mathbf{v}}_T \rfloor & \mathbf{0}_3 & \mathbf{0}_3 & {}^G_T\hat{\mathbf{R}} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}_{T11} & \mathbf{\Phi}_{T12} & \mathbf{\Phi}_{T13} & \mathbf{\Phi}_{T14} \\ \mathbf{\Phi}_{T21} & \mathbf{\Phi}_{T22} & \mathbf{\Phi}_{T23} & \mathbf{\Phi}_{T24} \\ \mathbf{\Phi}_{T31} & \mathbf{\Phi}_{T32} & \mathbf{\Phi}_{T33} & \mathbf{\Phi}_{T34} \\ \mathbf{\Phi}_{T41} & \mathbf{\Phi}_{T42} & \mathbf{\Phi}_{T43} & \mathbf{\Phi}_{T44} \end{bmatrix} \tag{G.134}$$

Hence, the target state transition matrix from t_0 to t_k can be solved as:

$$\mathbf{\Phi}_T^{(2)} = \begin{bmatrix} \mathbf{\Phi}_{T11} & \mathbf{\Phi}_{T12} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{\Phi}_{T31} & \mathbf{\Phi}_{T32} & \mathbf{I}_3 & \mathbf{\Phi}_{T34} \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{G.135}$$

where we have:

$$\mathbf{\Phi}_{T11} = {}^{T_k}_{T_0}\hat{\mathbf{R}} \tag{G.136}$$

$$\mathbf{\Phi}_{T12} = \int_{t_0}^{t} {}^{T_k}_{T_\tau}\hat{\mathbf{R}} d\tau \tag{G.137}$$

$$\dot{\mathbf{\Phi}}_{T31} = -\lfloor {}^G\hat{\mathbf{v}}_T(t) \rfloor {}^G_{T_0}\hat{\mathbf{R}} \tag{G.138}$$

$$\dot{\mathbf{\Phi}}_{T32} = -\lfloor {}^G\hat{\mathbf{v}}_T(t) \rfloor \int_{t_0}^{t} {}^G_{T_\tau}\hat{\mathbf{R}} d\tau \tag{G.139}$$

$$\mathbf{\Phi}_{T31} = -\lfloor {}^G\hat{\mathbf{p}}_{T_k} - {}^G\hat{\mathbf{p}}_{T_0} \rfloor {}^G_{T_0}\hat{\mathbf{R}} \tag{G.140}$$

$$\mathbf{\Phi}_{T32} = -\int_{t_0}^{t} \lfloor {}^G\hat{\mathbf{v}}_T(s) \rfloor \int_{t_0}^{s} {}^G_{T_\tau}\hat{\mathbf{R}} d\tau ds \tag{G.141}$$

$$\mathbf{\Phi}_{T34} = \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}} d\tau \tag{G.142}$$

Therefore, the total state transition matrix can be written as:

$$\mathbf{\Phi}_T^{(2)} = \begin{bmatrix} {}^{T_k}_{T_0}\hat{\mathbf{R}} & \int_{t_0}^{t} {}^{T_k}_{T_\tau}\hat{\mathbf{R}} d\tau & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -\lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} & -\int_{t_0}^{t} \lfloor {}^{G}\hat{\mathbf{v}}_T(s) \rfloor \int_{t_0}^{s} {}^{G}_{T_\tau}\hat{\mathbf{R}} d\tau ds & \mathbf{I}_3 & \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}} d\tau \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} \tag{G.143}$$

G.4.3 Observability Matrix

With abuse of notation, the k'th block of observability matrix can be written as:

$$\mathbf{M}_k^{(2)} = \mathbf{H}_{\mathbf{x}k}\mathbf{\Phi}^{(2)}(k,0) \tag{G.144}$$

$$= \mathbf{H}_{\mathbf{x}k} \begin{bmatrix} \mathbf{\Phi}_I & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi}_{fs} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_T^{(2)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{\Phi}_{ft} \end{bmatrix} \tag{G.145}$$

$$= \begin{bmatrix} \mathbf{H}_{C1}{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{H}_{C2}{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{H}_{C3}{}^{I}_{G}\hat{\mathbf{R}} \end{bmatrix} \times$$
$$\begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} & \mathbf{M}_{13} & \mathbf{M}_{14} & \mathbf{M}_{15} & \mathbf{M}_{16} & \mathbf{M}_{17} & \mathbf{M}_{18} & \mathbf{M}_{19} & \mathbf{M}_{1,10} & \mathbf{M}_{1,11} \\ \mathbf{M}_{21} & \mathbf{M}_{22} & \mathbf{M}_{23} & \mathbf{M}_{24} & \mathbf{M}_{25} & \mathbf{M}_{26} & \mathbf{M}_{27} & \mathbf{M}_{28} & \mathbf{M}_{29} & \mathbf{M}_{2,10} & \mathbf{M}_{2,11} \\ \mathbf{M}_{31} & \mathbf{M}_{32} & \mathbf{M}_{33} & \mathbf{M}_{34} & \mathbf{M}_{35} & \mathbf{M}_{36} & \mathbf{M}_{37} & \mathbf{M}_{38} & \mathbf{M}_{39} & \mathbf{M}_{3,10} & \mathbf{M}_{3,11} \end{bmatrix} \tag{G.146}$$

Then, we can solve this matrix. The first row of this matrix can be denoted as:

$$\mathbf{M}_{11} = \lfloor {}^{G}\hat{\mathbf{p}}_{fs} - {}^{G}\hat{\mathbf{p}}_{I_k} \rfloor {}^{G}_{I_k}\hat{\mathbf{R}}\mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} = \lfloor {}^{G}\hat{\mathbf{p}}_{fs} - {}^{G}\hat{\mathbf{p}}_{I_0} - {}^{G}\hat{\mathbf{v}}_{I_0}\Delta t_k + \frac{1}{2}{}^{G}\mathbf{g}\Delta t_k^2 \rfloor {}^{G}_{I_0}\hat{\mathbf{R}} \tag{G.147}$$

$$\mathbf{M}_{12} = \lfloor {}^{G}\hat{\mathbf{p}}_{fs} - {}^{G}\hat{\mathbf{p}}_{I_k} \rfloor {}^{G}_{I_k}\hat{\mathbf{R}}\mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.148}$$

$$\mathbf{M}_{13} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3\Delta t_k \tag{G.149}$$

$$\mathbf{M}_{14} = -\mathbf{\Phi}_{I54} \tag{G.150}$$

$$\mathbf{M}_{15} = -\mathbf{I}_3 \tag{G.151}$$

$$\mathbf{M}_{16} = \mathbf{I}_3 \tag{G.152}$$

$$\mathbf{M}_{17} = \mathbf{M}_{18} = \mathbf{M}_{19} = \mathbf{M}_{1,10} = \mathbf{M}_{1,11} = \mathbf{0}_3 \tag{G.153}$$

The second row of this matrix can be described as:

$$\mathbf{M}_{21} = \lfloor {}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor^{G}_{I_k}\hat{\mathbf{R}}\mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} \tag{G.154}$$

$$= \lfloor {}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} + {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{I_0} - {}^{G}\mathbf{v}_{I_0}\Delta t_k + \frac{1}{2}{}^{G}\mathbf{g}\Delta t_k^2 \rfloor^{G}_{I_0}\mathbf{R} \tag{G.155}$$

$$\mathbf{M}_{22} = \lfloor {}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor^{G}_{I_k}\hat{\mathbf{R}}\mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.156}$$

$$\mathbf{M}_{23} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3\Delta t_k \tag{G.157}$$

$$\mathbf{M}_{24} = -\mathbf{\Phi}_{I54} \tag{G.158}$$

$$\mathbf{M}_{25} = -\mathbf{I}_3 \tag{G.159}$$

$$\mathbf{M}_{26} = \mathbf{0}_3 \tag{G.160}$$

$$\mathbf{M}_{27} = -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{\Phi}_{T11} + \mathbf{\Phi}_{T31} = \lfloor -{}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{T_k} + {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor^{G}_{T_0}\hat{\mathbf{R}} \tag{G.161}$$

$$\mathbf{M}_{28} = -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{\Phi}_{T12} + \mathbf{\Phi}_{T32} \tag{G.162}$$

$$\mathbf{M}_{29} = \mathbf{I}_3 \tag{G.163}$$

$$\mathbf{M}_{2,10} = \mathbf{\Phi}_{T34} = \int_{t_0}^{t} {}^{G}_{T_\tau}\mathbf{R}\mathbf{d}\tau \tag{G.164}$$

$$\mathbf{M}_{2,11} = {}^{G}_{T_k}\hat{\mathbf{R}} \tag{G.165}$$

The third row of this matrix can be described as:

$$\mathbf{M}_{31} = \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{I_k} \rfloor^{G}_{I_k}\hat{\mathbf{R}}\mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} = \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{I_0} - {}^{G}\hat{\mathbf{v}}_{I_0} + \frac{1}{2}{}^{G}\mathbf{g}\Delta t_k^2 \rfloor^{G}_{I_0}\hat{\mathbf{R}} \tag{G.166}$$

$$\mathbf{M}_{32} = \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{I_k} \rfloor^{G}_{I_k}\hat{\mathbf{R}}\mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.167}$$

$$\mathbf{M}_{33} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3\Delta t_k \tag{G.168}$$

$$\mathbf{M}_{34} = -\mathbf{\Phi}_{I54} \tag{G.169}$$

$$\mathbf{M}_{35} = -\mathbf{I}_3 \tag{G.170}$$

$$\mathbf{M}_{36} = \mathbf{0}_3 \tag{G.171}$$

$$\mathbf{M}_{37} = \mathbf{\Phi}_{T31} = -\lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \tag{G.172}$$

$$\mathbf{M}_{38} = \mathbf{\Phi}_{T32} \tag{G.173}$$

$$\mathbf{M}_{39} = \mathbf{I}_3 \tag{G.174}$$

$$\mathbf{M}_{3,10} = \mathbf{\Phi}_{T34} \tag{G.175}$$

$$\mathbf{M}_{3,11} = \mathbf{0}_3 \tag{G.176}$$

Therefore, we can rewrite the equation as:

$$\mathbf{M}_k^{(2)} = \mathbf{H}_{\mathbf{x}k}\mathbf{\Phi}^{(2)}(k,0) \tag{G.177}$$

$$= \begin{bmatrix} \mathbf{H}_{C1}{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{H}_{C2}{}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{H}_{C3}{}^{I}_{G}\hat{\mathbf{R}} \end{bmatrix} \times$$

$$\begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} & -\mathbf{I}_3\Delta t_k & \mathbf{M}_{14} & -\mathbf{I}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{M}_{21} & \mathbf{M}_{22} & -\mathbf{I}_3\Delta t_k & \mathbf{M}_{24} & -\mathbf{I}_3 & \mathbf{0}_3 & \lfloor -{}^{G}_{T_k}\hat{\mathbf{R}}^T\hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{T_k} + {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} & \mathbf{M}_{28} & \mathbf{I}_3 & \mathbf{M}_{2,10} & {}^{G}_{T_k}\hat{\mathbf{R}} \\ \mathbf{M}_{31} & \mathbf{0}_3 & -\mathbf{I}_3\Delta t_k & \mathbf{M}_{34} & -\mathbf{I}_3 & \mathbf{0}_3 & \lfloor -{}^{G}\hat{\mathbf{p}}_{T_k} + {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} & \mathbf{M}_{38} & -\mathbf{I}_3 & \mathbf{M}_{3,10} & \mathbf{0}_3 \end{bmatrix} \tag{G.178}$$

Based on the constant local velocity assumption, the ${}^{T}\boldsymbol{\omega}_T$ and ${}^{T}\mathbf{v}_T$ are constant. Therefore, we have:

$$\mathbf{\Phi}_{T12}\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor = \int_{t_0}^{t} {}^{T_k}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau \lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor = \int_{t_0}^{t} {}^{T_k}_{T_\tau}\hat{\mathbf{R}} \lfloor {}^{T_\tau}\hat{\boldsymbol{\omega}}_T \rfloor \mathbf{d}\tau = {}^{T_k}_{T_k}\hat{\mathbf{R}} - {}^{T_k}_{T_0}\hat{\mathbf{R}} = \mathbf{I}_3 - {}^{T_k}_{T_0}\hat{\mathbf{R}} \tag{G.179}$$

$$\mathbf{\Phi}_{T32}\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor = -\int_{t_0}^{t} \lfloor {}^{G}\hat{\mathbf{v}}_T(s) \rfloor \int_{t_0}^{s} {}^{G}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau\mathbf{d}s \lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor \tag{G.180}$$

$$= -\int_{t_0}^{t} \lfloor {}^{G}\hat{\mathbf{v}}_T(s) \rfloor \left({}^{G}_{T_s}\hat{\mathbf{R}} - {}^{G}_{T_0}\hat{\mathbf{R}} \right) \mathbf{d}s \tag{G.181}$$

$$= -\int_{t_0}^{t} \lfloor {}^{G}\hat{\mathbf{v}}_T(s) \rfloor {}^{G}_{T_s}\hat{\mathbf{R}}\mathbf{d}s - \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \tag{G.182}$$

$$= -\int_{t_0}^{t} {}^{G}_{T_s}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{v}}_T \rfloor \mathbf{d}s - \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \tag{G.183}$$

$$\mathbf{\Phi}_{T34}\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor = \int_{t_0}^{t} {}^{G}_{T_\tau}\mathbf{R}\mathbf{d}\tau \lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor = \int_{t_0}^{t} {}^{G}_{T_\tau}\mathbf{R} \lfloor {}^{T_\tau}\boldsymbol{\omega} \rfloor \mathbf{d}\tau = {}^{G}_{T_k}\hat{\mathbf{R}} - {}^{G}_{T_0}\hat{\mathbf{R}} \tag{G.184}$$

The unobservable directions $\mathbf{N}^{(2)}$ span the right null space of the observability matrix $\mathbf{M}^{(2)}$, that is: $\mathbf{M}^{(2)}\mathbf{N}^{(2)} = \mathbf{0}$. Based on the derivation of the observability matrix, we can have the unobservable directions as:

$$\mathbf{N}^{(2)} = \begin{bmatrix} \mathbf{N}_1^{(2)} & \mathbf{N}_{2:4}^{(2)} & \mathbf{N}_{{}_G^T\mathbf{R}}^{(2)} \end{bmatrix} = \begin{bmatrix} {}_G^{I_0}\hat{\mathbf{R}}\,{}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{v}}_{I_0} \rfloor {}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{p}}_{I_0} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{p}}_{fs} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_3 \\ -{}_G^{T_0}\hat{\mathbf{R}}\,{}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \lfloor {}^T\hat{\boldsymbol{\omega}}_T \rfloor \\ -\lfloor {}^G\hat{\mathbf{p}}_{T_0} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \lfloor {}^T\hat{\mathbf{v}}_T \rfloor \\ \mathbf{0}_3 & \mathbf{0}_3 & \lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor \end{bmatrix} \tag{G.185}$$

If without the representative point ${}^G\mathbf{p}_T$ measurement, there will be one more unobservable direction relating to the representative point position of the target as:

$$\mathbf{N}_{{}^G\mathbf{p}_T}^{(2)} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \left({}_{T_0}^{G}\hat{\mathbf{R}}\right)^\top & \left(\lfloor {}^I\hat{\boldsymbol{\omega}} \rfloor\right)^\top & -\mathbf{I}_3 \end{bmatrix}^\top \tag{G.186}$$

G.4.4 Geometric Interpretation - Verification of $\mathbf{N}_{{}_G^T\mathbf{R}}^{(2)}$

If we disturb the target orientation by $\delta\boldsymbol{\phi}$, then we will have the disturbed state vector as:

$${}_G^{T'}\mathbf{R} = {}_T^{T'}\mathbf{R}\,{}_G^T\mathbf{R} \simeq (\mathbf{I}_3 - \lfloor \delta\boldsymbol{\phi} \rfloor)\,{}_G^T\mathbf{R} \tag{G.187}$$

$${}^{T'}\boldsymbol{\omega}_{T'} = {}_T^{T'}\mathbf{R}\,{}^T\boldsymbol{\omega}_T \simeq (\mathbf{I}_3 - \lfloor \delta\boldsymbol{\phi} \rfloor)\,{}^T\boldsymbol{\omega}_T = {}^T\boldsymbol{\omega}_T + \lfloor {}^T\boldsymbol{\omega}_T \rfloor \delta\boldsymbol{\phi} \tag{G.188}$$

$${}^G\mathbf{p}_{T'} = {}^G\mathbf{p}_T \tag{G.189}$$

$${}^{T'}\mathbf{v}_{T'} = {}_T^{T'}\mathbf{R}\,{}^T\mathbf{v}_T \simeq (\mathbf{I}_3 - \lfloor \delta\boldsymbol{\phi} \rfloor)\,{}^T\mathbf{v}_T = {}^T\mathbf{v}_T + \lfloor {}^T\mathbf{v}_T \rfloor \delta\boldsymbol{\phi} \tag{G.190}$$

$${}^{T'}\mathbf{p}_{ft} = {}_T^{T'}\mathbf{R}\,{}^T\mathbf{p}_{ft} \simeq (\mathbf{I}_3 - \lfloor \delta\boldsymbol{\phi} \rfloor)\,{}^T\mathbf{p}_{ft} = {}^T\mathbf{p}_{ft} + \lfloor {}^T\mathbf{p}_{ft} \rfloor \delta\boldsymbol{\phi} \tag{G.191}$$

Then, for the target feature measurements and target representative point measurement, we have:

$$
\begin{aligned}
{}^{I}\mathbf{p}_{ft'} &= {}^{I}_{G}\mathbf{R}\left({}^{G}_{T'}\mathbf{R}{}^{T'}\mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T'}\right) = {}^{I}_{G}\mathbf{R}\left({}^{G}_{T}\mathbf{R}^{T}_{T'}\mathbf{R}^{T'}_{T}\mathbf{R}^{T}\mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T}\right) && \text{(G.192)}\\
&= {}^{I}\mathbf{p}_{ft} && \text{(G.193)}\\
{}^{I}\mathbf{p}_{T'} &= {}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T'} - {}^{G}\mathbf{p}_{I}\right) = {}^{I}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T} - {}^{G}\mathbf{p}_{I}\right) && \text{(G.194)}\\
&= {}^{I}\mathbf{p}_{T} && \text{(G.195)}
\end{aligned}
$$

Hence, after the orientation disturbance, the system will still have the same target feature and target representative point measurements. Since we assume $\delta\boldsymbol{\phi}$ is a small perturbation, we can linearize the disturbed system state vector at $\hat{\mathbf{x}}$, then the related error states can be written as:

$$
\tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}'_{I} \\ {}^{G}\tilde{\mathbf{p}}'_{fs} \\ {}^{T'}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T'}\tilde{\boldsymbol{\omega}}_{T'} \\ {}^{G}\tilde{\mathbf{p}}_{T'} \\ {}^{T'}\tilde{\mathbf{v}}_{T'} \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_{I} \\ {}^{G}\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} + \delta\boldsymbol{\phi} \\ {}^{T}\tilde{\boldsymbol{\omega}}_{T} + \lfloor {}^{T}\hat{\boldsymbol{\omega}}_{T}\rfloor\delta\boldsymbol{\phi} \\ {}^{G}\tilde{\mathbf{p}}_{T} \\ {}^{T}\tilde{\mathbf{v}}_{T} + \lfloor {}^{T}\hat{\mathbf{v}}_{T}\rfloor\delta\boldsymbol{\phi} \\ {}^{T}\tilde{\mathbf{p}}_{ft} + \lfloor {}^{T}\hat{\mathbf{p}}_{ft}\rfloor\delta\boldsymbol{\phi} \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times 3} \\ \mathbf{0}_{3} \\ \mathbf{I}_{3} \\ \lfloor {}^{T}\hat{\boldsymbol{\omega}}_{T'}\rfloor \\ \mathbf{0}_{3} \\ \lfloor {}^{T}\hat{\mathbf{v}}_{T}\rfloor \\ \lfloor {}^{T}\hat{\mathbf{p}}_{ft}\rfloor \end{bmatrix} \delta\boldsymbol{\phi} = \tilde{\mathbf{x}} + \mathbf{N}^{(2)}_{{}^{T}_{G}\mathbf{R}}\delta\boldsymbol{\phi} \qquad \text{(G.196)}
$$

G.4.5 Geometric Interpretation - Verification of $\mathbf{N}^{(2)}_{{}^{G}\mathbf{p}_{T}}$

Similarly, we can verify unobservable directions related to the target representative point position $\mathbf{N}_{{}^{G}\mathbf{p}_{T}}$. We assume there is a small disturbance to representative position as $\delta\mathbf{p}$, then we can have the disturbed state parameters as:

$$
\begin{aligned}
{}^{T'}_{G}\mathbf{R} &= {}^{T}_{G}\mathbf{R} && \text{(G.197)}\\
{}^{T'}\boldsymbol{\omega}_{T'} &= {}^{T}\boldsymbol{\omega}_{T} && \text{(G.198)}\\
{}^{G}\mathbf{p}_{T'} &= {}^{G}\mathbf{p}_{T} + {}^{G}_{T}\mathbf{R}\delta\mathbf{p} \Rightarrow {}^{T}\mathbf{p}_{T'} = {}^{T}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T'} - {}^{G}\mathbf{p}_{T}\right) = \delta\mathbf{p} && \text{(G.199)}\\
{}^{T'}\mathbf{v}_{T'} &= {}^{T'}_{T}\mathbf{R}\left({}^{T}\mathbf{v}_{T} + \lfloor {}^{T}\boldsymbol{\omega}_{T}\rfloor\delta\mathbf{p}\right) && \text{(G.200)}\\
{}^{T'}\mathbf{p}_{ft} &= {}^{T'}_{T}\mathbf{R}\left({}^{T}\mathbf{p}_{ft} - {}^{T}\mathbf{p}_{T'}\right) = {}^{T}\mathbf{p}_{ft} - \delta\mathbf{p} && \text{(G.201)}
\end{aligned}
$$

Therefore, the related target feature in IMU frame can be written as:

$$ {}^{I}\mathbf{p}_{ft'} = {}^{I}_{G}\mathbf{R}\left({}^{G}_{T'}\mathbf{R}\,{}^{T'}\mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T'}\right) = {}^{I}_{G}\mathbf{R}\left({}^{G}_{T}\mathbf{R}\left({}^{T}\mathbf{p}_{ft} - \delta\mathbf{p}\right) - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T} + {}^{G}_{T}\mathbf{R}\delta\mathbf{p}\right) \tag{G.202} $$

$$ = {}^{I}\mathbf{p}_{ft} \tag{G.203} $$

That means, the system will have the the same target feature measurements even given the distance disturbance. Hence, with the disturbance to target position, we can write the disturbed error states as:

$$ \tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}'_{I} \\ {}^{G}\tilde{\mathbf{p}}'_{fs} \\ {}^{T'}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T'}\tilde{\boldsymbol{\omega}}_{T} \\ {}^{G}\tilde{\mathbf{p}}_{T'} \\ {}^{T'}\tilde{\mathbf{v}}_{T'} \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_{I} \\ {}^{G}\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ {}^{T}\tilde{\boldsymbol{\omega}}_{T'} \\ {}^{G}\tilde{\mathbf{p}}_{T} + {}^{G}_{T}\hat{\mathbf{R}}\delta\mathbf{p} \\ {}^{T}\tilde{\mathbf{v}}_{T} + \lfloor {}^{T}\hat{\boldsymbol{\omega}}_{T} \rfloor \delta\mathbf{p} \\ {}^{T}\tilde{\mathbf{p}}_{ft} - \delta\mathbf{p} \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times 3} \\ \mathbf{0}_{3} \\ \mathbf{0}_{3} \\ \mathbf{0}_{3} \\ {}^{G}_{T}\hat{\mathbf{R}} \\ \lfloor {}^{T}\hat{\boldsymbol{\omega}}_{T} \rfloor \\ -\mathbf{I}_{3} \end{bmatrix} \delta\mathbf{p} = \tilde{\mathbf{x}} + \mathbf{N}^{(2)}_{{}^{G}\mathbf{p}_{T}}\delta\mathbf{p} \tag{G.204} $$

G.5 Observability Analysis - Motion Model 3

G.5.1 Measurement Jacobians

Based on motion model 3, the measurement Jacobians can be written as:

$$ \mathbf{H}_{\mathbf{x}} = \begin{bmatrix} \mathbf{H}_{1} \\ \mathbf{H}_{2} \\ \mathbf{H}_{3} \end{bmatrix} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_{1}}{\partial \tilde{\mathbf{x}}} \\ \frac{\partial \tilde{\mathbf{z}}_{2}}{\partial \tilde{\mathbf{x}}} \\ \frac{\partial \tilde{\mathbf{z}}_{3}}{\partial \tilde{\mathbf{x}}} \end{bmatrix} \tag{G.205} $$

The Jacobians of the measurement to the environmental feature $\mathbf{H}_{1}$ can be written as:

$$ \mathbf{H}_{1} = \frac{\partial \tilde{\mathbf{z}}_{1}}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_{1}}{\partial \tilde{\mathbf{x}}_{I}} & \frac{\partial \tilde{\mathbf{z}}_{1}}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_{1}}{\partial \tilde{\mathbf{x}}^{(3)}_{T}} & \frac{\partial \tilde{\mathbf{z}}_{1}}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.206} $$

where we have:

$$ \mathbf{H}_{C1} = \frac{1}{{}^{I}\hat{z}^{2}_{fs}} \begin{bmatrix} {}^{I}\hat{z}_{fs} & 0 & -{}^{I}\hat{x}_{fs} \\ 0 & {}^{I}\hat{z}_{fs} & -{}^{I}\hat{y}_{fs} \end{bmatrix} \tag{G.207} $$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C1} {}^{I}_{G}\hat{\mathbf{R}} \begin{bmatrix} \lfloor {}^{G}\hat{\mathbf{p}}_{fs} - {}^{G}\hat{\mathbf{p}}_{I} \rfloor {}^{G}_{I}\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & -\mathbf{I}_3 \end{bmatrix} \tag{G.208}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} = \mathbf{H}_{C1} {}^{I}_{G}\hat{\mathbf{R}} \mathbf{I}_3 \tag{G.209}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}^{(3)}_T} = \mathbf{0}_{3\times 9} \tag{G.210}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} = \mathbf{0}_3 \tag{G.211}$$

The target feature measurement Jacobians $\mathbf{H}_2$ can be written as:

$$\mathbf{H}_2 = \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}^{(3)}_T} & \frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.212}$$

where we have:

$$\mathbf{H}_{C2} = \frac{1}{{}^{I}\hat{z}^2_{ft}} \begin{bmatrix} {}^{I}\hat{z}_{ft} & 0 & -{}^{I}\hat{x}_{ft} \\ 0 & {}^{I}\hat{z}_{ft} & -{}^{I}\hat{y}_{ft} \end{bmatrix} \tag{G.213}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C2} {}^{I}_{G}\hat{\mathbf{R}} \begin{bmatrix} \lfloor \Big({}^{T}_{G}\hat{\mathbf{R}}^{\top} {}^{T}\hat{\mathbf{p}}_{ft} - \big({}^{G}\hat{\mathbf{p}}_{I} - {}^{G}\hat{\mathbf{p}}_{T} \big) \Big) \rfloor {}^{G}_{I}\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & -\mathbf{I}_3 \end{bmatrix} \tag{G.214}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} = \mathbf{0}_3 \tag{G.215}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}^{(3)}_T} = \mathbf{H}_{C2} {}^{I}_{G}\hat{\mathbf{R}} \begin{bmatrix} -{}^{T}_{G}\hat{\mathbf{R}}^{\top} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor & \mathbf{0}_{3\times 1} & \mathbf{I}_3 & \mathbf{0}_{3\times 2} \end{bmatrix} \tag{G.216}$$

$$\frac{\partial \tilde{\mathbf{z}}_2}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} = \mathbf{H}_{C2} {}^{I}_{G}\hat{\mathbf{R}} {}^{T}_{G}\hat{\mathbf{R}}^{\top} \tag{G.217}$$

The target representative point measurement Jacobians can be written as:

$$\mathbf{H}_3 = \frac{\partial \tilde{\mathbf{z}}_2}{\partial \tilde{\mathbf{x}}} = \begin{bmatrix} \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}_I} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}^{(3)}_T} & \frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} \end{bmatrix} \tag{G.218}$$

where we have:

$$\mathbf{H}_{C3} = \frac{1}{{}^{I}\hat{z}^2_{T}} \begin{bmatrix} {}^{I}\hat{z}_{T} & 0 & -{}^{I}\hat{x}_{T} \\ 0 & {}^{I}\hat{z}_{T} & -{}^{I}\hat{y}_{T} \end{bmatrix} \tag{G.219}$$

$$\frac{\partial \tilde{\mathbf{z}}_1}{\partial \tilde{\mathbf{x}}_I} = \mathbf{H}_{C3} {}^{I}_{G}\hat{\mathbf{R}} \begin{bmatrix} \lfloor \big({}^{G}\hat{\mathbf{p}}_{T} - {}^{G}\hat{\mathbf{p}}_{I} \big) \rfloor {}^{G}_{I}\hat{\mathbf{R}} & \mathbf{0}_{3\times 9} & \mathbf{0}_3 \end{bmatrix} \tag{G.220}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{G}\tilde{\mathbf{p}}_{fs}} = \mathbf{0}_3 \tag{G.221}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial \tilde{\mathbf{x}}^{(3)}_T} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_{3\times 1} & \mathbf{I}_3 & \mathbf{0}_{3\times 2} \end{bmatrix} \tag{G.222}$$

$$\frac{\partial \tilde{\mathbf{z}}_3}{\partial {}^{T}\tilde{\mathbf{p}}_{ft}} = \mathbf{0}_3 \tag{G.223}$$

G.5.2 State Transition Matrix

The total system state transition matrix can be written as:

$$\boldsymbol{\Phi}^{(3)} = \begin{bmatrix} \boldsymbol{\Phi}_I & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_{fs} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_T^{(3)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_{ft} \end{bmatrix} \tag{G.224}$$

Note that $\boldsymbol{\Phi}_I$, $\boldsymbol{\Phi}_{fs}$ and $\boldsymbol{\Phi}_{ft}$ are the same as previous sections. We still need to solve the target state transition matrix $\boldsymbol{\Phi}_T^{(3)}$. We first linearize the related state evolution functions based on model 3, and get:

$$\dot{\tilde{\boldsymbol{\theta}}} = -\hat{\omega}_z \lfloor \mathbf{e}_3 \rfloor \tilde{\boldsymbol{\theta}} + \mathbf{e}_3 \hat{\omega}_z + \begin{bmatrix} \mathbf{I}_2 \\ \mathbf{0}_{1\times 2} \end{bmatrix} \begin{bmatrix} n_{w_x} \\ n_{w_y} \end{bmatrix} \tag{G.225}$$

$$\dot{\tilde{\mathbf{p}}}_T = -\hat{\mathbf{R}}^\top \lfloor \begin{bmatrix} \hat{v}_x \\ \hat{v}_y \\ 0 \end{bmatrix} \rfloor \tilde{\boldsymbol{\theta}} + \hat{\mathbf{R}}^\top \begin{bmatrix} \tilde{v}_x \\ \tilde{v}_y \\ 0 \end{bmatrix} + \hat{\mathbf{R}}^\top \mathbf{e}_3 n_{vz} \tag{G.226}$$

The linearized system can be written as:

$$\begin{aligned} \dot{\tilde{\mathbf{x}}}_T^{(3)} = \begin{bmatrix} \dot{\tilde{\boldsymbol{\theta}}} \\ \dot{\tilde{\omega}}_z \\ \dot{\tilde{\mathbf{p}}}_T \\ \dot{\tilde{v}}_x \\ \dot{\tilde{v}}_y \end{bmatrix} &= \begin{bmatrix} -\hat{\omega}_z \lfloor \mathbf{e}_3 \rfloor & \mathbf{e}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 0 \\ -\hat{\mathbf{R}}^\top \lfloor \begin{bmatrix} \hat{v}_x \\ \hat{v}_y \\ 0 \end{bmatrix} \rfloor & \mathbf{0}_{3\times 1} & \mathbf{0}_3 & \hat{\mathbf{R}}^\top \mathbf{e}_1 & \hat{\mathbf{R}}^\top \mathbf{e}_2 \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 0 \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 0 \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{\theta}} \\ \tilde{\omega}_z \\ \tilde{\mathbf{p}}_T \\ \tilde{v}_x \\ \tilde{v}_y \end{bmatrix} \\ &+ \begin{bmatrix} \begin{bmatrix} \mathbf{I}_2 \\ \mathbf{0}_{1\times 2} \end{bmatrix} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 2} & \mathbf{0}_1 & 1 & 0 & 0 \\ \mathbf{0}_{3\times 2} & \mathbf{0}_{3\times 1} & \hat{\mathbf{R}}^\top \mathbf{e}_3 & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 2} & \mathbf{0}_1 & 0 & 1 & 0 \\ \mathbf{0}_{1\times 2} & \mathbf{0}_1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} n_{wx} \\ n_{wy} \\ n_{wz} \\ n_{vz} \\ n_{vx} \\ n_{vy} \end{bmatrix} \end{aligned} \tag{G.227}$$

From the linearized system, we can get the state transition matrix $\mathbf{\Phi}_T^{(3)}$ evolution as:

$$\dot{\mathbf{\Phi}}_T^{(3)} = \begin{bmatrix} -\hat{\omega}_z \lfloor \mathbf{e}_3 \rfloor & \mathbf{e}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 0 \\ -{}_G^T\hat{\mathbf{R}}^\top \lfloor \begin{bmatrix} \hat{v}_x \\ \hat{v}_y \\ 0 \end{bmatrix} \rfloor & \mathbf{0}_{3\times 1} & \mathbf{0}_3 & {}_G^T\hat{\mathbf{R}}^\top \mathbf{e}_1 & {}_G^T\hat{\mathbf{R}}^\top \mathbf{e}_2 \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 0 \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}_{11} & \mathbf{\Phi}_{12} & \mathbf{\Phi}_{13} & \mathbf{\Phi}_{14} & \mathbf{\Phi}_{15} \\ \mathbf{\Phi}_{21} & \mathbf{\Phi}_{22} & \mathbf{\Phi}_{23} & \mathbf{\Phi}_{24} & \mathbf{\Phi}_{25} \\ \mathbf{\Phi}_{31} & \mathbf{\Phi}_{32} & \mathbf{\Phi}_{33} & \mathbf{\Phi}_{34} & \mathbf{\Phi}_{35} \\ \mathbf{\Phi}_{41} & \mathbf{\Phi}_{42} & \mathbf{\Phi}_{43} & \mathbf{\Phi}_{44} & \mathbf{\Phi}_{45} \\ \mathbf{\Phi}_{51} & \mathbf{\Phi}_{52} & \mathbf{\Phi}_{53} & \mathbf{\Phi}_{54} & \mathbf{\Phi}_{55} \end{bmatrix} \tag{G.228}$$

Therefore, we can define the state transition matrix as:

$$\mathbf{\Phi}_T^{(3)} = \begin{bmatrix} \mathbf{\Phi}_{T11} & \mathbf{\Phi}_{T12} & \mathbf{0}_3 & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 1 & \mathbf{0}_{1\times 3} & 0 & 0 \\ \mathbf{\Phi}_{T31} & \mathbf{\Phi}_{T32} & \mathbf{I}_3 & \mathbf{\Phi}_{T34} & \mathbf{\Phi}_{T35} \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 1 & 0 \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 1 \end{bmatrix} \tag{G.229}$$

where we have:

$$\mathbf{\Phi}_{T11} = {}_{T_0}^{T_k}\hat{\mathbf{R}} \tag{G.230}$$

$$\mathbf{\Phi}_{T12} = \int_{t_0}^{t} {}_{T_\tau}^{T_k}\mathbf{R} d\tau \mathbf{e}_3 \tag{G.231}$$

$$\dot{\mathbf{\Phi}}_{T31} = -\lfloor {}^G\hat{\mathbf{v}}_T(t) \rfloor {}_{T_0}^{G}\hat{\mathbf{R}} \tag{G.232}$$

$$\dot{\mathbf{\Phi}}_{T32} = -\lfloor {}^G\hat{\mathbf{v}}_T(t) \rfloor \int_{t_0}^{t} {}_{T_\tau}^{G}\hat{\mathbf{R}} d\tau \mathbf{e}_3 \tag{G.233}$$

$$\mathbf{\Phi}_{T31} = -\lfloor {}^G\hat{\mathbf{p}}_{T_k} - {}^G\hat{\mathbf{p}}_{T_0} \rfloor {}_{T_0}^{G}\hat{\mathbf{R}} \tag{G.234}$$

$$\mathbf{\Phi}_{T32} = -\int_{t_0}^{t} \lfloor {}^G\hat{\mathbf{v}}_T(s) \rfloor \int_{t_0}^{s} {}_{T_\tau}^{G}\hat{\mathbf{R}} d\tau ds \mathbf{e}_3 \tag{G.235}$$

$$\mathbf{\Phi}_{T34} = \int_{t_0}^{t} {}_{T_\tau}^{G}\hat{\mathbf{R}} d\tau \mathbf{e}_1 \tag{G.236}$$

$$\mathbf{\Phi}_{T35} = \int_{t_0}^{t} {}_{T_\tau}^{G}\hat{\mathbf{R}} d\tau \mathbf{e}_2 \tag{G.237}$$

Therefore, the target state transition matrix can be written as:

$$\boldsymbol{\Phi}_T^{(3)} = \begin{bmatrix} {}^{T_k}_{T_0}\hat{\mathbf{R}} & \int_{t_0}^{t} {}^{T_k}_{T_\tau}\mathbf{R} d\tau \mathbf{e}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{1\times 3} & 1 & \mathbf{0}_{1\times 3} & 0 & 0 \\ -\lfloor {}^G\hat{\mathbf{p}}_{T_k} - {}^G\hat{\mathbf{p}}_{T_0} \rfloor {}^G_{T_0}\hat{\mathbf{R}} & -\int_{t_0}^{t} \lfloor {}^G\hat{\mathbf{v}}_T(s) \rfloor \int_{t_0}^{s} {}^G_{T_\tau}\hat{\mathbf{R}} d\tau ds \mathbf{e}_3 & \mathbf{I}_3 & \int_{t_0}^{t} {}^G_{T_\tau}\hat{\mathbf{R}} d\tau \mathbf{e}_1 & \int_{t_0}^{t} {}^G_{T_\tau}\hat{\mathbf{R}} d\tau \mathbf{e}_2 \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 1 & 0 \\ \mathbf{0}_{1\times 3} & 0 & \mathbf{0}_{1\times 3} & 0 & 1 \end{bmatrix} \tag{G.238}$$

G.5.3 Observability Matrix

With abuse of notation, the k'th block of the observability matrix can be written as:

$$\mathbf{M}_k^{(3)} = \mathbf{H}_{\mathbf{x}k}\boldsymbol{\Phi}^{(3)}(k,0) \tag{G.239}$$

$$= \mathbf{H}_{\mathbf{x}k} \begin{bmatrix} \boldsymbol{\Phi}_I & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_{fs} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_T^{(3)} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\Phi}_{ft} \end{bmatrix} \tag{G.240}$$

$$= \begin{bmatrix} \mathbf{H}_{C1}{}^I_G\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{H}_{C2}{}^I_G\hat{\mathbf{R}} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{H}_{C3}{}^I_G\hat{\mathbf{R}} \end{bmatrix} \times$$
$$\begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} & \mathbf{M}_{13} & \mathbf{M}_{14} & \mathbf{M}_{15} & \mathbf{M}_{16} & \mathbf{M}_{17} & \mathbf{M}_{18} & \mathbf{M}_{19} & \mathbf{M}_{1,10} & \mathbf{M}_{1,11} & \mathbf{M}_{1,12} \\ \mathbf{M}_{21} & \mathbf{M}_{22} & \mathbf{M}_{23} & \mathbf{M}_{24} & \mathbf{M}_{25} & \mathbf{M}_{26} & \mathbf{M}_{27} & \mathbf{M}_{28} & \mathbf{M}_{29} & \mathbf{M}_{2,10} & \mathbf{M}_{2,11} & \mathbf{M}_{2,12} \\ \mathbf{M}_{31} & \mathbf{M}_{32} & \mathbf{M}_{33} & \mathbf{M}_{34} & \mathbf{M}_{35} & \mathbf{M}_{36} & \mathbf{M}_{37} & \mathbf{M}_{38} & \mathbf{M}_{39} & \mathbf{M}_{3,10} & \mathbf{M}_{3,11} & \mathbf{M}_{3,12} \end{bmatrix} \tag{G.241}$$

Then, we can solve this matrix row by row. The first row of this matrix can be denoted as:

$$\mathbf{M}_{11} = \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_{I_k} \rfloor {}^G_{I_k}\hat{\mathbf{R}}\boldsymbol{\Phi}_{I11} - \boldsymbol{\Phi}_{I51} = \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_{I_0} - {}^G\hat{\mathbf{v}}_{I_0}\Delta t_k + \frac{1}{2}{}^G\mathbf{g}\Delta t_k^2 \rfloor {}^G_{I_0}\hat{\mathbf{R}} \tag{G.242}$$

$$\mathbf{M}_{12} = \lfloor {}^G\hat{\mathbf{p}}_{fs} - {}^G\hat{\mathbf{p}}_{I_k} \rfloor {}^G_{I_k}\hat{\mathbf{R}}\boldsymbol{\Phi}_{I12} - \boldsymbol{\Phi}_{I52} \tag{G.243}$$

$$\mathbf{M}_{13} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3 \Delta t_k \tag{G.244}$$

$$\mathbf{M}_{14} = -\mathbf{\Phi}_{I54} \tag{G.245}$$

$$\mathbf{M}_{15} = -\mathbf{I}_3 \tag{G.246}$$

$$\mathbf{M}_{16} = \mathbf{I}_3 \tag{G.247}$$

$$\mathbf{M}_{17} = \mathbf{M}_{19} = \mathbf{M}_{1,12} = \mathbf{0}_3 \tag{G.248}$$

$$\mathbf{M}_{18} = \mathbf{M}_{1,10} = \mathbf{M}_{1,11} = \mathbf{0}_{3\times 1} \tag{G.249}$$

The second row of this matrix can be described as:

$$\mathbf{M}_{21} = \lfloor {}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor {}^{G}_{I_k}\hat{\mathbf{R}} \mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} \tag{G.250}$$

$$= \lfloor {}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} + {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{I_0} - {}^{G}\hat{\mathbf{v}}_{I_0} \Delta t_k + \frac{1}{2} {}^{G}\mathbf{g} \Delta t_k^2 \rfloor {}^{G}_{I_0}\hat{\mathbf{R}} \tag{G.251}$$

$$\mathbf{M}_{22} = \lfloor {}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor {}^{G}_{I_k}\hat{\mathbf{R}} \mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.252}$$

$$\mathbf{M}_{23} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3 \Delta t_k \tag{G.253}$$

$$\mathbf{M}_{24} = -\mathbf{\Phi}_{I54} \tag{G.254}$$

$$\mathbf{M}_{25} = -\mathbf{I}_3 \tag{G.255}$$

$$\mathbf{M}_{26} = \mathbf{0}_3 \tag{G.256}$$

$$\mathbf{M}_{27} = -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{\Phi}_{T11} + \mathbf{\Phi}_{T31} = \lfloor -{}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{T_k} + {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \tag{G.257}$$

$$\mathbf{M}_{28} = -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{\Phi}_{T12} + \mathbf{\Phi}_{T32} \tag{G.258}$$

$$\mathbf{M}_{29} = \mathbf{I}_3 \tag{G.259}$$

$$\mathbf{M}_{2,10} = \mathbf{\Phi}_{T34} = \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}} \mathbf{d}\tau \mathbf{e}_1 \tag{G.260}$$

$$\mathbf{M}_{2,11} = \mathbf{\Phi}_{T35} = \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}} \mathbf{d}\tau \mathbf{e}_2 \tag{G.261}$$

$$\mathbf{M}_{2,12} = {}^{G}_{T_k}\hat{\mathbf{R}} \tag{G.262}$$

The third row of this matrix can be described as:

$$\mathbf{M}_{31} = \lfloor -{}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor {}^{G}_{I_k}\hat{\mathbf{R}} \mathbf{\Phi}_{I11} - \mathbf{\Phi}_{I51} = \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{I_0} - {}^{G}\hat{\mathbf{v}}_{I_0} \Delta t_k + \frac{1}{2} {}^{G}\mathbf{g} \Delta t_k^2 \rfloor {}^{G}_{I_0}\hat{\mathbf{R}} \tag{G.263}$$

$$\mathbf{M}_{32} = \lfloor -{}^{G}\hat{\mathbf{p}}_{I_k} + {}^{G}\hat{\mathbf{p}}_{T_k} \rfloor {}^{G}_{I_k}\hat{\mathbf{R}}\mathbf{\Phi}_{I12} - \mathbf{\Phi}_{I52} \tag{G.264}$$

$$\mathbf{M}_{33} = -\mathbf{\Phi}_{I53} = -\mathbf{I}_3\Delta t_k \tag{G.265}$$

$$\mathbf{M}_{34} = -\mathbf{\Phi}_{I54} \tag{G.266}$$

$$\mathbf{M}_{35} = -\mathbf{I}_3 \tag{G.267}$$

$$\mathbf{M}_{36} = \mathbf{0}_3 \tag{G.268}$$

$$\mathbf{M}_{37} = \mathbf{\Phi}_{T31} = \lfloor -{}^{G}\hat{\mathbf{p}}_{T_k} + {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \tag{G.269}$$

$$\mathbf{M}_{38} = \mathbf{\Phi}_{T32} \tag{G.270}$$

$$\mathbf{M}_{39} = \mathbf{I}_3 \tag{G.271}$$

$$\mathbf{M}_{3,10} = \mathbf{\Phi}_{T34} = \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau\mathbf{e}_1 \tag{G.272}$$

$$\mathbf{M}_{3,11} = \mathbf{\Phi}_{T35} = \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau\mathbf{e}_2 \tag{G.273}$$

$$\mathbf{M}_{3,12} = \mathbf{0}_3 \tag{G.274}$$

Similarly, based on the constant angular velocity $\hat{\omega}_z$ and linear velocity $\hat{v}_x$ and $\hat{v}_y$ assumption, we have that:

$${}^{T_0}_{T_k}\hat{\mathbf{R}} = \begin{bmatrix} \cos(\hat{\omega}_z\Delta t_k) & -\sin(\hat{\omega}_z\Delta t_k) & 0 \\ \sin(\hat{\omega}_z\Delta t_k) & \cos(\hat{\omega}_z\Delta t_k) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{G.275}$$

$$\mathbf{\Phi}_{T12} = \int_{t_0}^{t} {}^{T_k}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau\mathbf{e}_3 = \int_{t_0}^{t} \begin{bmatrix} \cos(\hat{\omega}_z(\tau - t)) & -\sin(\hat{\omega}_z(\tau - t)) & 0 \\ \sin(\hat{\omega}_z(\tau - t)) & \cos(\hat{\omega}_z(\tau - t)) & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{d}\tau\mathbf{e}_3 = \mathbf{e}_3\Delta t_k \tag{G.276}$$

$$\mathbf{\Phi}_{T34}\hat{\omega}_z = \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau\mathbf{e}_1\hat{\omega}_z = {}^{G}_{T_0}\hat{\mathbf{R}}\int_{t_0}^{t} {}^{T_0}_{T_\tau}\hat{\mathbf{R}}\mathbf{d}\tau\mathbf{e}_1\hat{\omega}_z = {}^{G}_{T_0}\hat{\mathbf{R}}\int_{t_0}^{t} \begin{bmatrix} \cos(\hat{\omega}_z\delta\tau) \\ \sin(\hat{\omega}_z\delta\tau) \\ 0 \end{bmatrix} \mathbf{d}\tau\boldsymbol{\omega}_z \tag{G.277}$$

$$= {}^{G}_{T_0}\hat{\mathbf{R}} \left(- \begin{bmatrix} -\sin\left(\hat{\omega}_z \Delta t_k\right) \\ \cos\left(\hat{\omega}_z \Delta t_k\right) \\ 0 \end{bmatrix} + \mathbf{e}_2 \right) = {}^{G}_{T_0}\hat{\mathbf{R}} \left(\mathbf{I}_3 - {}^{T_0}_{T_k}\hat{\mathbf{R}} \right) \mathbf{e}_2 \tag{G.278}$$

$$\mathbf{\Phi}_{T35}\hat{\omega}_z = \int_{t_0}^{t} {}^{G}_{T_\tau}\hat{\mathbf{R}} \mathbf{d}\tau \mathbf{e}_2 \hat{\omega}_z = {}^{G}_{T_0}\hat{\mathbf{R}} \int_{t_0}^{t} {}^{T_0}_{T_\tau}\hat{\mathbf{R}} \mathbf{d}\tau \mathbf{e}_2 \boldsymbol{\omega}_z = {}^{G}_{T_0}\hat{\mathbf{R}} \int_{t_0}^{t} \begin{bmatrix} -\sin\left(\hat{\omega}_z \delta\tau\right) \\ \cos\left(\hat{\omega}_z \delta\tau\right) \\ 0 \end{bmatrix} \mathbf{d}\tau \hat{\omega}_z \tag{G.279}$$

$$= {}^{G}_{T_0}\hat{\mathbf{R}} \left(\begin{bmatrix} \cos\left(\hat{\omega}_z \Delta t_k\right) \\ \sin\left(\hat{\omega}_z \Delta t_k\right) \\ 0 \end{bmatrix} - \mathbf{e}_1 \right) = {}^{G}_{T_0}\hat{\mathbf{R}} \left({}^{T_0}_{T_k}\hat{\mathbf{R}} - \mathbf{I}_3 \right) \mathbf{e}_1 \tag{G.280}$$

$$\mathbf{M}_{27}\mathbf{e}_3\hat{\omega}_z = \lfloor -{}^{G}_{T_k}\hat{\mathbf{R}}^T \hat{\mathbf{p}}_{ft} - {}^{G}\hat{\mathbf{p}}_{T_k} + {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \mathbf{e}_3 \boldsymbol{\omega}_z \tag{G.281}$$

$$= -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \hat{\omega}_z \mathbf{e}_3 - \lfloor {}^{G}\hat{\mathbf{p}}_{T_k} - {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \mathbf{e}_3 \hat{\omega}_z \tag{G.282}$$

$$= -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \hat{\omega}_z \mathbf{e}_3 - \lfloor {}^{G}\hat{\mathbf{p}}_{T_0} + {}^{G}_{T_0}\hat{\mathbf{R}} \int_{t_0}^{t} {}^{T_0}_{T_\tau}\mathbf{R}^{T_\tau} \mathbf{v}_T \mathbf{d}\tau - {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \mathbf{e}_3 \hat{\omega}_z \tag{G.283}$$

$$= -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \hat{\omega}_z \mathbf{e}_3 + \lfloor {}^{G}_{T_0}\hat{\mathbf{R}} \left({}^{T_0}_{T_k}\hat{\mathbf{R}} - \mathbf{I}_3 \right) \mathbf{e}_2 \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \mathbf{e}_3 \hat{v}_x - \lfloor {}^{G}_{T_0}\hat{\mathbf{R}} \left({}^{T_0}_{T_k}\hat{\mathbf{R}} - \mathbf{I}_3 \right) \mathbf{e}_1 \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} \mathbf{e}_3 \hat{v}_y \tag{G.284}$$

$$= -{}^{G}_{T_k}\hat{\mathbf{R}} \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \hat{\omega}_z \mathbf{e}_3 + {}^{G}_{T_k}\hat{\mathbf{R}} \mathbf{e}_1 \hat{v}_x + {}^{G}_{T_k}\hat{\mathbf{R}} \mathbf{e}_2 \hat{v}_y - {}^{G}_{T_0}\hat{\mathbf{R}} \mathbf{e}_1 \hat{v}_x - {}^{G}_{T_0}\hat{\mathbf{R}} \mathbf{e}_2 \hat{v}_y \tag{G.285}$$

Therefore, we can rewrite the equation as:

$$\mathbf{M}_k^{(3)} = \mathbf{H}_{\mathbf{x}k} \mathbf{\Phi}^{(3)}(k, 0) \tag{G.286}$$

$$= \begin{bmatrix} \mathbf{H}_{C1} {}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{H}_{C2} {}^{I}_{G}\hat{\mathbf{R}} & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{H}_{C3} {}^{I}_{G}\hat{\mathbf{R}} \end{bmatrix} \times$$

$$\begin{bmatrix} \mathbf{M}_{11} & \mathbf{M}_{12} & -\mathbf{I}_3 \Delta t_k & \mathbf{M}_{14} & -\mathbf{I}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_3 & \mathbf{0}_{3\times1} & \mathbf{0}_{3\times1} & \mathbf{0}_3 \\ \mathbf{M}_{21} & \mathbf{M}_{22} & -\mathbf{I}_3 \Delta t_k & \mathbf{M}_{24} & -\mathbf{I}_3 & \mathbf{0}_3 & \mathbf{M}_{27} & \mathbf{M}_{28} & \mathbf{I}_3 & \mathbf{M}_{2,10} & \mathbf{M}_{2,11} & {}^{G}_{T_k}\hat{\mathbf{R}} \\ \mathbf{M}_{31} & \mathbf{M}_{32} & -\mathbf{I}_3 \Delta t_k & \mathbf{M}_{34} & -\mathbf{I}_3 & \mathbf{0}_3 & \lfloor -{}^{G}\hat{\mathbf{p}}_{T_k} + {}^{G}\hat{\mathbf{p}}_{T_0} \rfloor {}^{G}_{T_0}\hat{\mathbf{R}} & \mathbf{M}_{38} & \mathbf{I}_3 & \mathbf{M}_{3,10} & \mathbf{M}_{3,11} & \mathbf{0}_3 \end{bmatrix} \tag{G.287}$$

The unobservable directions $\mathbf{N}^{(3)}$ span the right null space of the observability matrix $\mathbf{M}^{(3)}$, that is: $\mathbf{M}^{(3)}\mathbf{N}^{(3)} = \mathbf{0}$. Based on the derivation of the observability matrix, we can have the unobservable directions as:

$$\mathbf{N}^{(3)} = \begin{bmatrix} \mathbf{N}_1^{(3)} & \mathbf{N}_{2:4}^{(3)} & \mathbf{N}_{{}_G^T\mathbf{R}}^{(3)} \end{bmatrix} = \begin{bmatrix} {}_G^{I_0}\hat{\mathbf{R}}\,{}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{3\times 1} & \mathbf{0}_3 & \mathbf{0}_3 \\ -\lfloor {}^G\hat{\mathbf{v}}_{I_0} \rfloor {}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{0}_{3\times 1} \\ \mathbf{0}_{3\times 1} & \mathbf{0}_3 & \mathbf{0}_{3\times 1} \\ -\lfloor {}^G\hat{\mathbf{p}}_{I_0} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_{3\times 1} \\ -\lfloor {}^G\hat{\mathbf{p}}_{fs} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_{3\times 1} \\ -{}_G^{T_0}\hat{\mathbf{R}}\,{}^G\mathbf{g} & \mathbf{0}_3 & \mathbf{e}_3 \\ \mathbf{0}_1 & \mathbf{0}_3 & \mathbf{0}_1 \\ -\lfloor {}^G\hat{\mathbf{p}}_{T_0} \rfloor {}^G\mathbf{g} & \mathbf{I}_3 & \mathbf{0}_{3\times 1} \\ \mathbf{0}_1 & \mathbf{0}_{1\times 3} & \hat{v}_y \\ \mathbf{0}_1 & \mathbf{0}_{1\times 3} & -\hat{v}_x \\ \mathbf{0}_{3\times 1} & \mathbf{0}_3 & \lfloor {}^T\hat{\mathbf{p}}_{ft} \rfloor \mathbf{e}_3 \end{bmatrix} \tag{G.288}$$

However, without the measurement of the target representative point, the system will have additional 3 unobservable directions related to the target representative point position as:

$$\mathbf{N}_{{}_G\mathbf{p}_T}^{(3)} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}_{T_0}^G\hat{\mathbf{R}}\mathbf{e}_3\right)^\top & \mathbf{0}_1 & \mathbf{0}_1 & -\mathbf{e}_3^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}_{T_0}^G\hat{\mathbf{R}}\mathbf{e}_2\right)^\top & -\hat{\omega}_z & \mathbf{0}_1 & -\mathbf{e}_2^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}_{T_0}^G\hat{\mathbf{R}}\mathbf{e}_1\right)^\top & \mathbf{0}_1 & \hat{\omega}_z & -\mathbf{e}_1^\top \end{bmatrix}^\top \tag{G.289}$$

G.5.4 Geometric Interpretation - Verification of $\mathbf{N}_{{}_G^T\mathbf{R}}^{(3)}$

We assume there is a small angle $\delta\theta$ change to the orientation, then, the disturbance to the state can be written as:

$${}_G^{T'}\mathbf{R} = {}_T^{T'}\mathbf{R}\,{}_G^T\mathbf{R} = \left(\mathbf{I}_3 - \lfloor \mathbf{e}_3 \rfloor \delta\theta\right) {}_G^T\mathbf{R} \tag{G.290}$$

$${}^{T'}\mathbf{p}_{ft} = {}_T^{T'}\mathbf{R}\,{}^T\mathbf{p}_{ft} = \left(\mathbf{I}_3 - \lfloor \mathbf{e}_3 \rfloor \delta\theta\right) {}^T\mathbf{p}_{ft} \simeq {}^T\mathbf{p}_{ft} + \lfloor {}^T\mathbf{p}_{ft} \rfloor \mathbf{e}_3 \delta\theta \tag{G.291}$$

$$ {}^{T'}\mathbf{v}_{T'} = {}^{T'}_{T}\mathbf{R}^{T}\mathbf{v}_{T} = (\mathbf{I}_3 - \lfloor \mathbf{e}_3 \delta\theta \rfloor)\, {}^{T}\mathbf{v}_{T} = {}^{T}\mathbf{v}_{T} + \lfloor {}^{T}\mathbf{v}_{T} \rfloor \mathbf{e}_3 \delta\theta = {}^{T}\mathbf{v}_{T} + \begin{bmatrix} v_y \\ -v_x \\ 0 \end{bmatrix} \delta\theta \tag{G.292} $$

Similar to previous sections, the target feature measurements and the target representative point measurements will remain the same:

$$ {}^{I}\mathbf{p}_{ft'} = {}^{I}_{G}\mathbf{R} \left({}^{G}_{T'}\mathbf{R}^{T'} \mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T'} \right) = {}^{I}_{G}\mathbf{R} \left({}^{G}_{T}\mathbf{R}^{T}_{T'}\mathbf{R}^{T'}_{T}\mathbf{R}^{T} \mathbf{p}_{ft} - {}^{G}\mathbf{p}_{I} + {}^{G}\mathbf{p}_{T} \right) \tag{G.293} $$

$$ = {}^{I}\mathbf{p}_{ft} \tag{G.294} $$

$$ {}^{I}\mathbf{p}_{T'} = {}^{I}_{G}\mathbf{R} \left({}^{G}\mathbf{p}_{T'} - {}^{G}\mathbf{p}_{I} \right) = {}^{I}_{G}\mathbf{R} \left({}^{G}\mathbf{p}_{T} - {}^{G}\mathbf{p}_{I} \right) \tag{G.295} $$

$$ = {}^{I}\mathbf{p}_{T} \tag{G.296} $$

Hence, the disturbed error states can be written as:

$$ \tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}'_I \\ {}^{G}\tilde{\mathbf{p}}'_{fs} \\ {}^{T'}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}'_z \\ {}^{G}\tilde{\mathbf{p}}_{T'} \\ \tilde{v}'_x \\ \tilde{v}'_y \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_I \\ {}^{G}\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} + \mathbf{e}_3\delta\theta \\ \tilde{\omega}_z \\ {}^{G}\tilde{\mathbf{p}}_{T} \\ \tilde{v}_x + \hat{v}_y\delta\theta \\ \tilde{v}_y - \hat{v}_x\delta\theta \\ {}^{T}\tilde{\mathbf{p}}_{ft} + \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{e}_3\delta\theta \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times1} \\ \mathbf{0}_{3\times1} \\ \mathbf{e}_3 \\ \mathbf{0}_1 \\ \mathbf{0}_{3\times1} \\ \hat{v}_y \\ -\hat{v}_x \\ \lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{e}_3 \end{bmatrix} \delta\theta = \tilde{\mathbf{x}} + \mathbf{N}^{(3)}_{{}^{T}_{G}\mathbf{R}}\delta\theta \tag{G.297} $$

G.5.5 Geometric Interpretation - Verification of $\mathbf{N}^{(3)}_{{}^{G}\mathbf{p}_T}$

If we disturb the target position by δp along the direction of ${}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_3$, then we will have:

$$ {}^{G}\mathbf{p}_{T'} = {}^{G}\mathbf{p}_{T} + {}^{G}_{T_0}\mathbf{R}\mathbf{e}_3\delta p \tag{G.298} $$

$$ {}^{T}\mathbf{p}_{T'} = {}^{T}_{G}\mathbf{R} \left({}^{G}\mathbf{p}_{T'} - {}^{G}\mathbf{p}_{T} \right) = \mathbf{e}_3\delta p \tag{G.299} $$

$$ {}^{T'}\mathbf{p}_{ft} = {}^{T'}_{T}\mathbf{R} \left({}^{T}\mathbf{p}_{ft} - \mathbf{e}_3\delta p \right) \tag{G.300} $$

Similar to previous sections, the target feature measurements will remain the same. Hence, the disturbed error state can be written as:

$$\tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}_I' \\ {}^G\tilde{\mathbf{p}}_{fs}' \\ {}^{T'}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}_z' \\ {}^G\tilde{\mathbf{p}}_{T'} \\ \tilde{v}_x' \\ \tilde{v}_y' \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_I \\ {}^G\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}_z \\ {}^G\tilde{\mathbf{p}}_T + {}^{G}_{T0}\hat{\mathbf{R}}\mathbf{e}_3\delta p \\ \tilde{v}_x \\ \tilde{v}_y \\ {}^T\tilde{\mathbf{p}}_{ft} - \mathbf{e}_3\delta p \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times 1} \\ \mathbf{0}_{3\times 1} \\ \mathbf{0}_{3\times 1} \\ 0 \\ {}^{G}_{T0}\hat{\mathbf{R}}\mathbf{e}_3 \\ 0 \\ 0 \\ -\mathbf{e}_3 \end{bmatrix} \delta p \tag{G.301}$$

Follow the same way, if we disturb the target position by δp along the direction of ${}^{G}_{T}\mathbf{R}\mathbf{e}_2$, then we will have the disturbed state as:

$${}^G\mathbf{p}_{T'} = {}^G\mathbf{p}_T + {}^{G}_{T}\mathbf{R}\mathbf{e}_2\delta p \tag{G.302}$$

$${}^T\mathbf{p}_{T'} = {}^{T}_{G}\mathbf{R}\left({}^G\mathbf{p}_{T'} - {}^G\mathbf{p}_T\right) = \mathbf{e}_2\delta p \tag{G.303}$$

$${}^{T'}\mathbf{p}_{ft} = {}^{T'}_{T}\mathbf{R}\left({}^T\mathbf{p}_{ft} - \mathbf{e}_2\delta p\right) \tag{G.304}$$

$${}^{T'}\mathbf{v}_{T'} = {}^{T'}_{T}\mathbf{R}\left({}^T\mathbf{v}_T + \lfloor\omega_z\mathbf{e}_3\rfloor\mathbf{e}_2\delta p\right) \tag{G.305}$$

Similarly, the target feature measurements will remain the same. Hence, the disturbed error state can be written as:

$$\tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}_I' \\ {}^G\tilde{\mathbf{p}}_{fs}' \\ {}^{T'}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}_z' \\ {}^G\tilde{\mathbf{p}}_{T'} \\ \tilde{v}_x' \\ \tilde{v}_y' \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_I \\ {}^G\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}_z \\ {}^G\tilde{\mathbf{p}}_T + {}^{G}_{T}\hat{\mathbf{R}}\mathbf{e}_2\delta p \\ \tilde{v}_x - \hat{\omega}_z\delta p \\ \tilde{v}_y \\ {}^T\tilde{\mathbf{p}}_{ft} - \mathbf{e}_2\delta p \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times 1} \\ \mathbf{0}_{3\times 1} \\ \mathbf{0}_{3\times 1} \\ 0 \\ {}^{G}_{T0}\hat{\mathbf{R}}\mathbf{e}_2 \\ -\hat{\omega}_z \\ 0 \\ -\mathbf{e}_2 \end{bmatrix} \delta p \tag{G.306}$$

Again, if we disturb the target position by δp along the direction of ${}^{G}_{T}\mathbf{R}\mathbf{e}_1$, then we will have the disturbed state and error states as:

$$
\begin{aligned}
{}^{G}\mathbf{p}_{T'} &= {}^{G}\mathbf{p}_{T} + {}^{G}_{T}\mathbf{R}\mathbf{e}_1\delta p && \text{(G.307)}\\
{}^{T}\mathbf{p}_{T'} &= {}^{T}_{G}\mathbf{R}\left({}^{G}\mathbf{p}_{T'} - {}^{G}\mathbf{p}_{T}\right) = \mathbf{e}_1\delta p && \text{(G.308)}\\
{}^{T'}\mathbf{p}_{ft} &= {}^{T'}_{T}\mathbf{R}\left({}^{T}\mathbf{p}_{ft} - \mathbf{e}_1\delta p\right) && \text{(G.309)}\\
{}^{T'}\mathbf{v}_{T'} &= {}^{T'}_{T}\mathbf{R}\left({}^{T}\mathbf{v}_{T} + \lfloor\omega_z\mathbf{e}_3\rfloor\mathbf{e}_1\delta p\right) && \text{(G.310)}
\end{aligned}
$$

Similarly, the target feature measurements will remain the same. Hence, the disturbed error state can be written as:

$$
\tilde{\mathbf{x}}' = \begin{bmatrix} \tilde{\mathbf{x}}'_I \\ {}^{G}\tilde{\mathbf{p}}'_{fs} \\ {}^{T'}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}'_z \\ {}^{G}\tilde{\mathbf{p}}_{T'} \\ \tilde{v}'_x \\ \tilde{v}'_y \\ {}^{T'}\tilde{\mathbf{p}}_{ft} \end{bmatrix} = \begin{bmatrix} \tilde{\mathbf{x}}_I \\ {}^{G}\tilde{\mathbf{p}}_{fs} \\ {}^{T}_{G}\tilde{\boldsymbol{\theta}} \\ \tilde{\omega}_z \\ {}^{G}\tilde{\mathbf{p}}_{T} + {}^{G}_{T}\hat{\mathbf{R}}\mathbf{e}_2\delta p \\ \tilde{v}_x \\ \tilde{v}_y + \hat{\omega}_z\delta p \\ {}^{T}\tilde{\mathbf{p}}_{ft} - \mathbf{e}_3\delta p \end{bmatrix} = \tilde{\mathbf{x}} + \begin{bmatrix} \mathbf{0}_{15\times1} \\ \mathbf{0}_{3\times1} \\ \mathbf{0}_3 \\ 0 \\ {}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_1 \\ 0 \\ \hat{\omega}_z \\ -\mathbf{e}_1 \end{bmatrix}\delta p \qquad \text{(G.311)}
$$

G.6 Summary and Discussion

Given motion model 1 (constant ${}^{G}\mathbf{v}_T$ and constant ${}^{T}\boldsymbol{\omega}_T$), with static feature, target feature, and representative point measurements, the system will have at least 7 unobservable directions related to global yaw, global IMU position ${}^{G}\mathbf{p}_I$, and the target orientation ${}^{T}_{G}\mathbf{R}$. The first 4 unobservable directions are inherited from VINS [58]. If measurements of the target's representative point are *unavailable* (due to occlusion), the system will have one more unobservable direction related to the representative point position along the rotation axis of ${}^{T}\boldsymbol{\omega}_T$. The standard and extra unobservable directions related to the target can be respectively written as:

$$
\mathbf{N}^{(1)}_{{}^{T}_{G}\mathbf{R}} = \begin{bmatrix} \mathbf{0}_{3\times15} & \mathbf{0}_3 & \mathbf{I}_3 & \left(\lfloor{}^{T}\hat{\boldsymbol{\omega}}_T\rfloor\right)^{\top} & \mathbf{0}_3 & \mathbf{0}_3 & \left(\lfloor{}^{T}\hat{\mathbf{p}}_{ft}\rfloor\right)^{\top} \end{bmatrix}^{\top} \qquad \text{(G.312)}
$$

$$\mathbf{N}_{{}^{G}\hat{\mathbf{p}}_T}^{(1)} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \left({}^{G}_{T_0}\hat{\mathbf{R}}\,{}^{T}\hat{\boldsymbol{\omega}}_T\right)^\top & \mathbf{0}_{1\times 3} & \left(-{}^{T}\hat{\boldsymbol{\omega}}_T\right)^\top \end{bmatrix}^\top \tag{G.313}$$

For motion model 2 (constant ${}^{T}\mathbf{v}_T$ and constant ${}^{T}\boldsymbol{\omega}_T$), with all measurements, the system will also have at least 7 unobservable directions, which have the same geometric interpretation as those of model 1. Similarly, without representative point measurements, the system will have 3 additional unobservable directions related to the full 3D position of the representative point. The standard and extra unobservable directions related to the target can be respectively written as:

$$\mathbf{N}_{{}^{T}_{G}\mathbf{R}}^{(2)} = \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{I}_3 & \left(\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor\right)^\top & \mathbf{0}_3 & \left(\lfloor {}^{T}\hat{\mathbf{v}}_T \rfloor\right)^\top & \left(\lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor\right)^\top \end{bmatrix}^\top \tag{G.314}$$

$$\mathbf{N}_{{}^{G}\mathbf{p}_T}^{(2)} = \begin{bmatrix} \mathbf{0}_{3\times 15} & \mathbf{0}_3 & \mathbf{0}_3 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\right)^\top & \left(\lfloor {}^{T}\hat{\boldsymbol{\omega}}_T \rfloor\right)^\top & -\mathbf{I}_3 \end{bmatrix}^\top \tag{G.315}$$

For motion model 3 (planar motion with constant ω_z, v_x, and v_y), with all measurements, unlike with the above two models, the target's roll and pitch become observable and the system has at least 5 unobservable directions where 4 are inherited from VINS and 1 is related to target orientation yaw. Without the representative point measurements, similar to model 2, the system will also gain additional 3 unobservable directions related to the full 3D position of the representative point. The standard and extra unobservable directions related to the target can be respectively written as:

$$\mathbf{N}_{{}^{T}_{G}\mathbf{R}}^{(3)} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{e}_3^\top & \mathbf{0}_1 & \mathbf{0}_{3\times 1} & \hat{v}_y & -\hat{v}_x & \left(\lfloor {}^{T}\hat{\mathbf{p}}_{ft} \rfloor \mathbf{e}_3\right)^\top \end{bmatrix}^\top \tag{G.316}$$

$$\mathbf{N}_{{}^{G}\mathbf{p}_T}^{(3)} = \begin{bmatrix} \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_3\right)^\top & \mathbf{0}_1 & \mathbf{0}_1 & -\mathbf{e}_3^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_2\right)^\top & -\hat{\omega}_z & \mathbf{0}_1 & -\mathbf{e}_2^\top \\ \mathbf{0}_{1\times 15} & \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \mathbf{0}_1 & \left({}^{G}_{T_0}\hat{\mathbf{R}}\mathbf{e}_1\right)^\top & \mathbf{0}_1 & \hat{\omega}_z & -\mathbf{e}_1^\top \end{bmatrix}^\top \tag{G.317}$$

Appendix H

PERMISSIONS

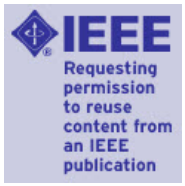
Closed-form preintegration methods for graph-based visual–inertial navigation

Author: Kevin Eckenhoff, Patrick Geneva, Guoquan Huang

Publication: The International Journal of Robotics Research

Publisher: SAGE Publications

Date: 04/01/2019



Multi-Camera Visual-Inertial Navigation with Online Intrinsic and Extrinsic Calibration

Conference Proceedings: 2019 International Conference on Robotics and Automation (ICRA)

Author: Kevin Eckenhoff

Publisher: IEEE

Date: May 2019

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

IEEE Requesting permission to reuse content from an IEEE publication

Sensor-Failure-Resilient Multi-IMU Visual-Inertial Navigation

Conference Proceedings: 2019 International Conference on Robotics and Automation (ICRA)

Author: Kevin Eckenhoff

Publisher: IEEE

Date: May 2019

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

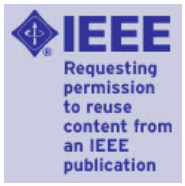
1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW



Tightly-Coupled Visual-Inertial Localization and 3-D Rigid-Body Target Tracking

Author: Kevin Eckenhoff

Publication: IEEE Robotics and Automation Letters

Publisher: IEEE

Date: April 2019

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW