MINS: Efficient and Robust Multisensor-aided Inertial Navigation System

Journal Title
XX(X):1–29
©The Author(s) 2023
Reprints and permission:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Woosik Lee, Patrick Geneva, Chuchu Chen, and Guoquan Huang

Abstract

Robust multisensor fusion of multi-modal measurements such as IMUs, wheel encoders, cameras, LiDARs, and GPS holds great potential due to its innate ability to improve resilience to sensor failures and measurement outliers, thereby enabling robust autonomy. To the best of our knowledge, this work is among the first to develop a consistent tightly-coupled Multisensor-aided Inertial Navigation System (MINS) that is capable of fusing the most common navigation sensors in an efficient filtering framework, by addressing the particular challenges of computational complexity, sensor asynchronicity, and intra-sensor calibration. In particular, we propose a consistent high-order on-manifold interpolation scheme to enable efficient asynchronous sensor fusion and state management strategy (i.e. dynamic cloning). The proposed dynamic cloning leverages motion-induced information to adaptively select interpolation orders to control computational complexity while minimizing trajectory representation errors. We perform online intrinsic and extrinsic (spatiotemporal) calibration of all onboard sensors to compensate for poor prior calibration and/or degraded calibration varying over time. Additionally, we develop an initialization method with only proprioceptive measurements of IMU and wheel encoders, instead of exteroceptive sensors, which is shown to be less affected by the environment and more robust in highly dynamic scenarios. We extensively validate the proposed MINS in simulations and large-scale challenging real-world datasets, outperforming the existing state-of-the-art methods, in terms of localization accuracy, consistency, and computation efficiency. We have also open-sourced our algorithm, simulator, and evaluation toolbox for the benefit of the community: https://github.com/rpng/mins.

Keywords

Multisensor system, inertial navigation, sensor fusion, sensor calibration, Monte-Carlo analysis

1 Introduction

Robust, accurate, real-time localization is a fundamental capability for autonomous vehicles. This is often addressed by leveraging multisensor fusion approaches, in part due to the fact that multi-modal sensors can offer complementary and/or redundant sensing under different environmental conditions (Lynen et al. 2013; Hackett and Shah 1990). Among all possible navigation sensors, inertial measurement units (IMUs), cameras, wheel encoders, global navigation satellite systems (GNSS), and LiDARs prevail in recent literature (Cao et al. 2022; Lv et al. 2023; Wang et al. 2021a). While it appears to be straightforward in principle to fuse all these heterogeneous sensors to achieve optimal localization performance, there is only limited work that has used greater than three sensing modalities because of the inherent difficulties of robust and efficient multisensor fusion.

As multisensor systems – in particular, those that forgo hardware-synchronization due to its significant cost – inevitably produce asynchronous or out-of-sequence data, it remains challenging to optimally model and update such asynchronous and delayed multi-modal information. The ability to seamlessly handle these is a key attribute of multisensor systems as it endows the systems with greater flexibility to use more sensors. To this end, multisensor calibration is another essential but difficult component. Many existing methods assume perfect offline calibration (Sun et al. 2022; Nguyen et al. 2021b; Meng et al. 2017), which, however, inject unmodelled errors into the estimator and thus degrade localization performance if the prior calibration was poor or inevitably changes during long-term operation. As incorporating more sensors increases not only the number of measurements to process but also the size of the state to be estimated, highly-efficient fusion of all available multi-modal measurements is always challenging given stringent computational resources and latency requirements. Additionally, system initialization is necessary for robust estimation and failure recovery but practically is challenging due to constrained under-actuated motion and short time horizons.

To address the aforementioned challenges, in this work, we develop a novel Multisensor-aided Inertial Navigation System (MINS), which builds upon the inertial navigation system (INS) as the backbone and efficiently fuses the multi-modal measurements of wheel encoders, cameras, LiDARs, and GNSS while performing full online spatiotemporal calibration. The proposed MINS employs a continuous-time

Robot Perception and Navigation Group, University of Delaware, Newark, DE 19716, USA.

Corresponding author:
Woosik Lee, Department of Mechanical Engineering, University of Delaware, 126 Spencer Lab, Newark, DE 19716, USA.
Email: woosik@udel.edu

state representation to reduce the computational complexity of asynchronous and delayed measurements and proposes robust initialization in challenging dynamic scenes, while achieving accurate real-time localization.

In particular, the main contributions of this work include:

- We, for the first time, design an efficient and robust filtering-based MINS estimator that fuses five most commonly-seen navigation sensors in a tightly-coupled fashion, including an IMU, a pair of wheel encoders, and an arbitrary number of cameras, LiDARs, and GNSS receivers. Thanks to the complementary and redundant sensing capabilities, the proposed approach is resilient to sensor failures and measurement depletion (e.g. in a dark room for cameras, open field for LiDARs, or indoors for GNSS) as well as measurement outliers, enabling it to output continuous and uninterrupted estimates for downstream applications.
- The proposed MINS introduces a consistent high-order on-manifold state interpolation methodology to process asynchronous and delayed measurements, which enables flexibility in adding auxiliary sensors while ensuring computational efficiency. Furthermore, we investigate the long-standing loss of trajectory fidelity introduced by continuous-time representations and reformulate the measurement model to account for trajectory representation errors. A dynamic cloning algorithm which adaptively balances the computational complexity, estimation accuracy, and trajectory representation errors is developed and validated.
- The proposed MINS performs online intrinsic and extrinsic (spatiotemporal) calibration of all sensors onboard, which enables the system to be robust to poor prior calibration or time-varying calibration. Moreover, we develop a robust initialization technique using only the IMU and wheel-encoder proprioceptive measurements for the case of high speed and dynamic object filled environments, where conventional methods which leverage exteroceptive sensors fail to initialize.
- We validate the proposed MINS extensively in both realistic simulations and large-scale real-world datasets, showing that the proposed method can achieve high efficiency, robustness, consistency, and accuracy in many challenging scenarios. We have also open-sourced our codebase, simulator, and evaluation toolbox for the benefit of the research community.

2 Related Work

Although there is a rich literature on multi-sensor fusion (Chen et al. 2022b; Xu et al. 2022b), we here only review the most relevant multi-sensor systems with three or more sensing modalities, while focusing on sensor asynchronicity, delayed measurements, and online sensor calibration.

2.1 Estimation with Three Sensing Modalities

Visual-Inertial Navigation Systems (VINS) that combine the inertial reading of the IMU and the visual bearing information of the camera have become popular due to their complementary nature (Huang 2019). Many multi-sensor navigation systems encompass VINS as a key module of the whole system due to its compactness and rich information.

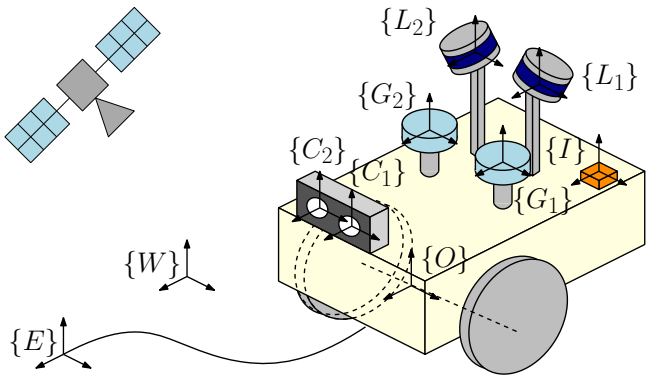


Figure 1. Frames used in the proposed MINS. In this example, 2 cameras $\{C\}$, 2 GNSS sensors $\{G\}$, 2 LiDARs $\{L\}$, 1 IMU $\{I\}$, and 1 wheel odometry frame $\{O\}$ are shown along with local world frame $\{W\}$ and global ENU frame $\{E\}$.

2.1.1 LiDAR-VINS

LiDAR is one of the popular sensors to be fused with VINS due to its measurement being invariant to the light and its ability to provide depth information directly, which can compensate for the degeneracy of the camera in many scenarios, such as in a dark cave or facing featureless wall (Xu et al. 2022b). V-LOAM (Zhang and Singh 2018) is one of the early works that loosely couples a LiDAR to the camera and IMU frame-to-frame leveraging VINS to get the initial matching of the LiDAR point cloud. Wang et al. (2019) improved this by designing a graph-based estimator including loop closure, while similar approaches can be found in many other works (Camurri et al. 2020; Khattak et al. 2020). A focus of loosely coupled systems has been to address the degeneracy issues of the independent camera and LiDAR odometry algorithms. For example LiDAR (+ IMU) odometry is combined with VINS in a graph formulation as separate sub-systems to enable robustness (Shan et al. 2021; Shao et al. 2019; Zhao et al. 2021). Many studies adapt filtering frameworks to design more tightly coupled systems which come with the estimate consistency and accuracy gains. MSCKF-based (Mourikis and Roumeliotis 2007) designs are popular and adapted by many works to great success (Zuo et al. 2019a, 2020; Lee et al. 2021; Yang et al. 2019b). In those works, features, such as planes or lines, are extracted from the LiDAR measurements and fused with camera measurements in a tightly coupled way. Similarly, the error state iterated Kalman filter (ESIKF) is adapted (Lin et al. 2021; Lin and Zhang 2022; Zheng et al. 2022) while fusing LiDAR without extracting the features (a.k.a. direct method) was also studied (Lin and Zhang 2022; Zheng et al. 2022).

2.1.2 GNSS-VINS

A GNSS sensor directly provides absolute position information, although the accuracy of it is highly dependent on the surrounding environment and the availability of external correction data. For these reasons, GNSS fusion with VINS has been investigated to provide locally precise

and globally accurate localization. There are many works fusing GNSS with VINS in a graph formulation via both loosely (Qin et al. 2019; Merfels and Stachniss 2016, 2017) and tightly (Mascaro et al. 2018; Cioffi and Scaramuzza 2020; Rehder et al. 2012; Wang et al. 2021b; Han et al. 2022). One of the most popular graph-based works is VINS-Fusion (Qin et al. 2019) which takes both VINS estimation and GNSS measurements to build a pose graph for optimization. However, GNSS measurements are often intermittent and asynchronous to other sensors which max graph construction a complex design. Cioffi and Scaramuzza (2020) addressed this issue by creating synthetic synchronized global measurements using both IMU and GNSS measurements, but introduced inconsistent measurement re-use of inertial information. There exist many filter-based techniques (Lee et al. 2016; Ramanandan et al. 2019; Chambers et al. 2014; Lynen et al. 2013) which fused GNSS with VINS, with Lee et al. (2020a) being the first to address the asynchronicity issue by interpolating historical stochastically cloned IMU poses. More recent works have focused on fusing lower-level GNSS measurements (GNSS satellite signals) with VINS (Dong and Zhang 2022; Liu et al. 2021b; Lee et al. 2022; Cao et al. 2022) showing robustness in limited FOV scenarios and improved localization performance.

2.1.3 Wheel-VINS

Many of the simultaneous localization and mapping (SLAM) applications are on ground vehicles. Therefore, it appears to be straightforward in principle to fuse the wheel (encoder) with other sensors for ground robot navigation. Especially for VINS on wheeled robots, it is known that their estimated state can have addititional unobservable directions beyond the standard 4 degrees of freedom (DOF) depending on the motion (Wu et al. 2017) (calibration parameters have additional challenges, (Yang et al. 2019a)). Thus the fusion of additional scale information from other sensors is necessary to build a consistent estimator. To this end, wheel odometry fusion with VINS has been investigated in both graph form (Kang et al. 2019; Dang et al. 2018; Zheng and Liu 2019; Liu et al. 2021a; Zuo et al. 2019b) and the filter form (Wu et al. 2017; Lee et al. 2020b; Serov et al. 2021; Ma et al. 2019, 2020).

One of the problems of wheel fusion is that the readings typically provide 2D motion information while the estimated robot state is generally in 3D space. Many studies address this issue through either tightly coupled integration of inertial and the wheel in 3D (Liu et al. 2019; Su et al. 2021; Quan et al. 2019; Liu et al. 2021a; Jung et al. 2020), projecting the 3D state onto the local 2D plane (Wu et al. 2017; Geneva et al. 2020b; Lee et al. 2020b), or applying planar motion constraint (Kang et al. 2019; Zheng and Liu 2019). Leveraging the ground shape information, such as plane model (Wu et al. 2017) or quadratic polynomial (Zhang et al. 2021; Zuo et al. 2019b), has shown to improve the robot pose estimation. The most common kinematic model of the wheel adapted is the differential drive (Wu et al. 2017; Lee et al. 2020b) while other models, such as 2 DOF vehicle dynamics model (Kang et al. 2019), Ackermann model (Ma et al. 2019, 2020), or ICR model (Zuo et al. 2019b) are used for different types of platforms.

2.2 Estimation with Four or More Sensing Modalities

Despite many advantages, such as robustness, accuracy, and applicability, a relatively small number of systems equipped with 4 or more sensor modalities have been introduced in part due to the difficulty of handling large computation and sensor measurements that are either asynchronous, delayed, or both. However, the ability to properly handle asynchronous and delayed sensor measurements is important for multi-sensor systems because it enables the system to flexibly add an arbitrary number of sensors that are either homogeneous or heterogeneous, model measurements more precisely, and include all the information without naively dropping it. Table 1 summarizes the related literature.

2.2.1 Graph-based Systems

The graph formulations can easily handle the delayed measurements owing to their nature of carrying all/part of the history of measurements which allows for delayed states to be inserted into the optimization problem (Chiu et al. 2014). The more recent VINS-Fusion (Qin et al. 2019) employs a pose graph which loosely couples VINS (tightly coupled IMU and camera) with other global sensors, such as GNSS, magnetometer, or barometer but assumes sensors are perfectly synchronized. There exist other graph formulations, such as SVIn2 (Rahman et al. 2022), which fused IMU, camera, water pressure sensor, and sonar to perform underwater SLAM in pose graph formulations, Lvio-fusion (Jia et al. 2021) tightly fused IMU, camera, and LiDAR for trajectory estimation and in turn, loosely fused GNSS in a pose graph, or GR-Fusion (Wang et al. 2021a), which tightly fused IMU, camera, LiDAR, wheel, and GNSS in the graph along with ground estimation and mapping. However, all of these works assumed the measurements are perfectly synchronized which prevents their plug-and-play use. The direct extension of these graph-based methods to asynchronous systems requires the estimation of the state at every measurement time, which requires techniques to bound computation, such as performing updates with only a subset of sensors (Chiu et al. 2014), to maintain real-time performance. Another challenge is the inclusion of temporal calibration parameter of sensors, which typically has been ignored or handled by hardware-synchronization (i.e. PTP network time protocol (Eidson et al. 2002)).

On the other hand, VIRAL SLAM (Nguyen et al. 2021b) which fused IMU and stereo camera with multiple UWBs and LiDARs handled asynchronicity by synchronizing all the sensors to the base LiDAR measurement time. To do so, IMU and UWBs are interpolated, non-base LiDAR point clouds are unwarped to the base LiDAR measurement time, and camera measurements that are closest to the time are admitted while others are dropped. The system is improved to a more general multi-sensor system, VIRAL-Fusion (Nguyen et al. 2021a), synchronizing the sensors by creating the state independent of the sensor measurement timings, interpolating the states to handle UWB measurements, and generating synthetic measurements at the state timestamp with measurement interpolation. Similarly, VILENS (Wisth et al. 2022) which fused IMU, leg odometry, camera, and LiDAR resolved the asynchronicity issue by synchronizing all the sensors to the camera time, interpolating IMU and

Table 1. Summary of the state-of-the-art multisensor navigation systems

	Algorithms	IMU	Camera	GNSS	Wheel	LiDAR	Asynchronous Sensors	Delayed Measurements	Online Calibration
Filter	**MINS** (proposed)	1	N	N	1	N	◯	◯	◯
	Kubelka et al. (2015)	1	1	×	1	1	×	×	×
	Simanek et al. (2015)	1	1	×	1	1	×	×	×
	Meng et al. (2017)	1	×	1	1	1	×	×	×
	Suhr et al. (2016)	1	1	1	1	×	×	×	×
	MSF-EKF	1	1	N	×	×	◯	◯	△
	Hausman et al. (2016)	1	1	1	×	×	◯	◯	△
	Tessier et al. (2006)	1	×	1	×	1	◯	◯	×
	MaRS	1	2	1	×	×	◯	◯	△
	Pronto	1	2	×	1	1	◯	◯	×
	Shen et al. (2014)	1	2	1	×	1	◯	◯	×
	Wu et al. (2022)	1	2	1	1	1	◯	◯	×
	Lee et al. (2021)	1	2	1	1	1	◯	◯	◯
Graph	VINS-Fusion	1	2	N	×	×	×	◯	△
	SVIn2	1	N	×	×	×	×	◯	×
	Lvio-Fusion	1	2	1	×	1	×	◯	×
	GR-Fusion	1	2	1	1	1	×	◯	△
	Chiu et al. (2014)	1	N	1	×	1	◯	◯	×
	VIRAL SLAM	1	2	×	×	N	△	◯	×
	VIRAL-Fusion	1	N	×	×	N	◯	◯	×
	VILENS	1	2	×	1	1	◯	◯	×
	VIL-Fusion	1	2	1	×	1	◯	◯	×

* N: arbitrary number of sensors supported, ◯: fully supported, △: partially supported, ×: not supported.

leg measurements, and unwarping LiDAR point cloud to the camera timestamp using the integrated IMU factor. VIL-Fusion (Sun et al. 2022) which fused IMU, camera, LiDAR, and GNSS also synchronized IMU, camera, and LiDAR in the same fashion while linear interpolation was used to handle GNSS. While these works look to address the asynchronicity problem without increasing the computational complexity, they introduce an additional unmodeled source of error from the approximation.

2.2.2 *Filter-based Systems*

Compared to graph-based systems, filter-based methods remain popular due to their efficiency, which is crucial to computationally complex multi-sensor systems. However, handling asynchronous and delayed sensor measurements presents challenges as filters only keep the latest information and maybe a short period of historical stochastic poses. In many early filtering approaches, such as Kubelka et al. (2015) and Simanek et al. (2015) which fused IMU, omnidirectional camera, wheel, and LiDAR in EKF, Meng et al. (2017) which fused IMU, GNSS, wheel, and LiDAR in unscented Kalman filter (UKF), or Suhr et al. (2016) which fused IMU, camera, GNSS, wheel, symbolic road map in particle filter (PF), the sensor measurements are assumed to be synchronized and time-ordered which injects unmodeled error to the estimator in case the assumption does not hold.

MSF-EKF (Lynen et al. 2013) is among the first work to fuse generic relative and absolute measurements from IMU, camera, GNSS, and pressure sensor in EKF and also handled asynchronous and delayed sensor measurements. In their work, the current state is re-predicted forward in time from the queried state when a delayed measurement comes in, which requires all subsequent measurement updates to be reapplied, and asynchronicity is handle by propagating to the temporally closest queried state to the measurement time Similarly spirited works that handle asynchronous and delayed sensor measurements are by Hausman et al. (2016) which fused IMU, camera, GNSS, and UWB in iterated EKF, Tessier et al. (2006) which fused magnetometer, GNSS, radar, and gyroscope in PF, MaRS (Brommer et al. 2020) which fused IMU, stereo camera, GNSS, barometer in EKF, and Pronto (Camurri et al. 2020) which fused IMU, leg odometry, camera, and LiDAR in EKF keeping 10 seconds of worth measurements, prior & posterior states, and the corresponding covariances. However, keeping the previous states and the measurements, and re-processing those measurements every time a delayed measurement is received, incurs a higher than needed computational cost which may result in losing real-time estimation performance.

One of the popular methods to improve efficiency by avoiding this re-computation is using a buffer to hold the measurements and processing them in a time-ordered manner. Shen et al. (2014) fused IMU, camera, GNSS, 2D LiDAR, pressure altimeter, and magnetometer in UKF buffering 0.1 seconds of the measurements to handle asynchronous and delayed sensor measurements. A similar approach can be found in the work of Wu et al. (2022) which fused IMU, stereo camera, wheel, LiDAR, and AHRS via a system which is composed of 3 subsystems (wheel+AHRS, VINS, LiDAR odometry) where the sensor measurements are abstracted into pose or velocity information and fused in EKF formulation. The buffer logic is useful to sort the unordered measurements, but the buffering window cannot be too large because it delays every measurement

update, thereby introducing a large latency to the real-time robotic system and is unacceptable in safety critical systems. Additionally, fusing sensors with large delays may not be feasible with a buffer approach (e.g. GNSS measurement delay can be on the order of 0.85 seconds (Lee et al. 2020a)).

2.2.3 Continuous-time Estimation

An alternative paradigm that elegantly allows the incorporation of asynchronous and delayed sensor measurements is the continuous-time estimation which represents the robot state at an arbitrary time using temporal basis functions. A popular temporal basis functions is the uniform B-spline (Furgale et al. 2013; Zheng et al. 2015; Droeschel and Behnke 2018; Cioffi et al. 2022; Lv et al. 2022) while there are other approaches, such as Gaussian process (Barfoot 2017; Anderson et al. 2015), wavelets (Anderson et al. 2014), or Taylor-series (Li and Mourikis 2014b). However, uniform B-spline, in general, requires a fixed frequency between control nodes and requires sufficiently high frequency to ensure accurate trajectory fidelity in highly dynamic motions. Too high of control node frequency may inefficiently increase computational complexity due to the estimation of a larger state vector, while a too low frequency may hurt the estimation performance due to inaccurate trajectory representation (see (Cioffi et al. 2022) for comparison to discrete-time formulations). In part due to these reasons, many methods that utilize B-spline for the continuous-time estimation are offline (Rehder et al. 2016a; Furgale et al. 2012; Lang et al. 2022) or are applied to small systems (Mueggler et al. 2018; Quenzel and Behnke 2021; Rodrigues et al. 2021).

Recently, MIMC-VINS (Eckenhoff et al. 2021) investigated a trajectory formulation using high-order polynomials over stochastic poses. The major benefits of this method are that it can handle a non-uniform temporal pose distribution and has a low computational cost to reconstruct different orders of polynomials as the exact robot poses are the control nodes. The 1-order linear interpolation variant of this has been leveraged by many works to great success (Paul and Roumeliotis 2018; Geneva et al. 2018a; Lee et al. 2020a; Sun et al. 2022). Delayed measurements can be naturally incorporated since the pose at *any* time instance within the historical stochastic clone window can be found. However, in the case of highly dynamic motions, the polynomial trajectory representation still has a loss of fidelity, as do temporal basis functions, as it is still limited by the clone rate and can introduce unmodeled errors. We will directly investigate the magnitude and impact of these unmodeled errors and properly model their characteristics in the proposed MINS to enable consistent estimation.

2.3 Online Sensor Calibration

Calibration methods can be classified into two methodologies: offline and online. Offline methods estimate the calibration parameters in batch optimization fashion with the aim to provide the most accurate results at the cost of high computation and time. There are few works that jointly calibrate 4 or more sensing modalities offline, such as a general framework (Rehder et al. 2016b) for the calibration of the spatiotemporal parameters, iCalib (Yang et al. 2021) for IMU, camera, LiDAR, and wheel calibration, or ATOM (Oliveira et al. 2022) for the camera, LiDAR, and RaDAR calibration. One can find more offline calibration methods in pair-wise systems, such as Kalibr (Rehder et al. 2016a), COC (Heng et al. 2013), MSG-cal (Owens et al. 2015), or MVIS (Yang et al. 2023). However, the offline calibration results may still have some errors that may result in estimator inconsistency if not properly modeled and are unable to cope with only platform reconfigurations. To this end, instead of blindly trusting them as true, many multi-sensor systems append the calibration parameters in the state to improve robustness by modeling their uncertainties and jointly estimating them while performing navigation which is called online calibration.

Among those 4 or more sensor fusion systems introduced (Sec. 2.2), MSF-EKF (Lynen et al. 2013), MaRS (Brommer et al. 2020), and the work of Hausman et al. (2016) supports the spatial calibration of onboard sensors while VINS-Fusion (Qin et al. 2019; Xu et al. 2022b) only supports the spatiotemporal extrinsic calibration between the camera and IMU. GR-Fusion (Wang et al. 2021a) further calibrated the spatiotemporal extrinsic of the LiDAR, intrinsic and spatial extrinsic of the wheel, and temporal extrinsic of the camera. However, the camera intrinsic and spatial extrinsic calibration was done offline and the temporal offset of the wheel was not calibrated. Our previous work (Lee et al. 2021) was the only work that showed full spatiotemporal calibration of all onboard sensors (camera, GNSS, wheel, and LiDAR), but it was limited to a single sensor per modality besides cameras. There exist systems that fully/partially calibrate the calibration parameters of smaller sensing modalities online, such as LiDAR-camera-IMU (Zuo et al. 2019a, 2020; Lee et al. 2021; Ye et al. 2021), GNSS-camera-IMU (Girrbach et al. 2019; Lee et al. 2020a; Han et al. 2022), wheel-camera-IMU (Liu et al. 2019; Jung et al. 2020; Lee et al. 2020b).

2.4 Extension of Our Previous Publications

While this article evolved from our prior works (Lee et al. 2020a,b, 2021; Geneva et al. 2020a; Eckenhoff et al. 2021), there are significant contributions differentiating this work. To be more specific, MINS leverages the high-order polynomial ideology of MIMC-VINS to enable high-frequency fusion of asynchronous and delayed arbitrary numbers of sensors within a unified continuous-time architecture. On top of this, we provide a thorough investigation of the error induced by the interpolation, propose the incorporation of the error model that improves system consistency, and further extend this idea to dynamic cloning to enable significant computational saving while maintaining localization accuracy. MINS enables spatiotemporal calibration of all onboard sensors, which differentiates MINS from OpenVINS (Geneva et al. 2020a) which only focused on VIO intrinsics and spatiotemporal and Lee et al. (2020a) which performed GNSS fusion with linear interpolation alongside VIO. Compared to VIWO (Lee et al. 2020b) which presented wheel encoder angular velocity fusion with VINS, MINS is additionally capable of fusing linear velocities of the wheel encoder and angular & linear velocities of wheel odometry frame along with planar motion constraint. Furthermore, this paper provides a novel IMU-wheel-based system initialization technique that is robust to very challenging scenarios. The LiDAR

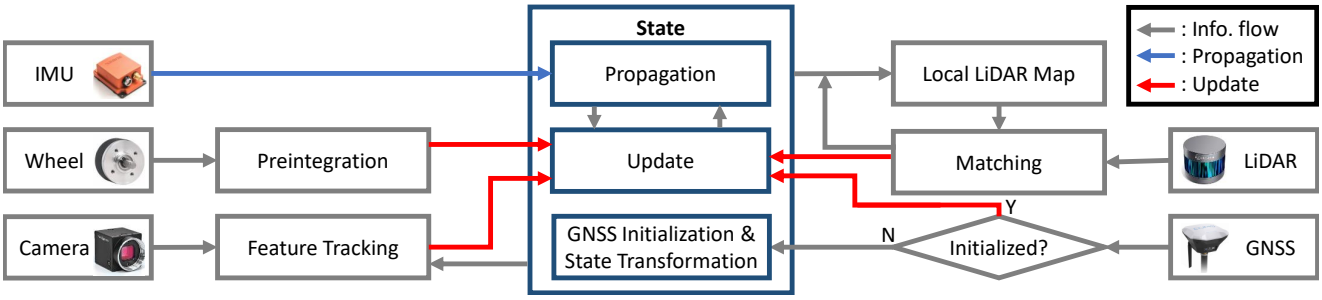


Figure 2. Overview of the proposed MINS showing each sensor component and their roles in state estimation.

fusion technique of MINS has been completely redesigned as compared to Lee et al. (2021) to enable multi-LiDARs fusion along with spatiotemporal calibration. MINS adapts the direct method (Xu et al. 2022a) of alignment instead of extracting LOAM features, improving the efficiency and consistency of the update process, and adding a mapping functionality. We additionally stress that the comprehensive accuracy, consistency, computation, and ablation studies of each part of the system in realistic simulations and in real-world experiments are non-trivial for multi-sensor fusion frameworks.

3 MINS State Estimation

Building upon the backbone INS, the proposed MINS propagates the state with the IMU measurements and updates in an efficient filtering framework with multi-modal measurements of a number of exteroceptive sensors such as cameras, LiDARs, GNSS receivers, and wheel encoders. As shown in Fig. 1, we use $\{I\}$ to denote the IMU frame, $\{C\}$ for the camera frame, $\{O\}$ for the wheel odometry frame, $\{L\}$ for the LiDAR frame, $\{G\}$ for the GNSS frame, while $\{E\}$ for the East-North-Up (ENU) frame corresponding to the GNSS. Fig. 2 depicts the the proposed MINS architecture.

Extended from the MSCKF-based VINS (Mourikis and Roumeliotis 2007; Geneva et al. 2020a), the state vector of the proposed MINS includes the IMU navigation state $\mathbf{x}_{I_k}$ and a set of historical IMU poses $\mathbf{x}_{H_k}$ (a.k.a. clones):

$$\mathbf{x}_k = (\mathbf{x}_{I_k},\ \mathbf{x}_{H_k}) \tag{1}$$

$$\mathbf{x}_{I_k} = ({}^{I_k}_{E}\mathbf{R},\ {}^{E}\mathbf{p}_{I_k},\ {}^{E}\mathbf{v}_{I_k},\ \mathbf{b}_g,\ \mathbf{b}_a) \tag{2}$$

$$\mathbf{x}_{H_k} = ({}^{I_{k-1}}_{E}\mathbf{R},\ {}^{E}\mathbf{p}_{I_{k-1}},\ \cdots,\ {}^{I_{k-h}}_{E}\mathbf{R},\ {}^{E}\mathbf{p}_{I_{k-h}}) \tag{3}$$

where ${}^{B}_{A}\mathbf{R}$ is a rotation matrix from $\{A\}$ to $\{B\}$, ${}^{B}\mathbf{p}_A$ and ${}^{B}\mathbf{v}_A$ are the position and linear velocity of $\{A\}$ in $\{B\}$, $\mathbf{b}_g$ and $\mathbf{b}_a$ are the biases of the gyroscope and accelerometer. We define $\mathbf{x} = \hat{\mathbf{x}} \boxplus \tilde{\mathbf{x}}$, where $\mathbf{x}$ is the true state, $\hat{\mathbf{x}}$ is its estimate, $\tilde{\mathbf{x}}$ is the error state, and the operation $\boxplus$ which maps the error state to its corresponding manifold (Hertzberg et al. 2013). Note that we represent the state variables in $\{E\}$ as the "global" GNSS coordinate after the system is initialized using the available GNSS measurements (see Sec. 5.2), otherwise, the state is expressed in the local world frame $\{W\}$ (which is often the "global" frame in local navigation systems (Qin et al. 2018; Xu et al. 2022a)).

In what follows, we present in detail each of the five sensing modalities and the proposed fusion methods to propagate and update the state with their measurements.

3.1 IMU

High-rate IMU measurements typically include the angular velocity $\boldsymbol{\omega}_m$ and local linear acceleration $\mathbf{a}_m$:

$$\boldsymbol{\omega}_{m_k} = \boldsymbol{\omega}_k + \mathbf{b}_g + \mathbf{n}_g \tag{4}$$

$$\mathbf{a}_{m_k} = \mathbf{a}_k + {}^{I_k}_{E}\mathbf{R}\,{}^{E}\mathbf{g} + \mathbf{b}_a + \mathbf{n}_a \tag{5}$$

where $\boldsymbol{\omega}_k$ and $\mathbf{a}_k$ are the true angular velocity and linear acceleration at time t_k; $\mathbf{b}_g$ and $\mathbf{b}_a$ are the biases of the gyroscope and the accelerometer; ${}^{E}\mathbf{g} \simeq [0\ 0\ 9.81]^\top$ is the global gravity; $\mathbf{n}_g$ and $\mathbf{n}_a$ are zero mean Gaussian noises. These measurements are used to propagate the IMU state $\mathbf{x}_I$ from t_k to t_{k+1} based on the following generic nonlinear kinematic model (Trawny and Roumeliotis 2005):

$$\hat{\mathbf{x}}_{I_{k+1}} = f(\hat{\mathbf{x}}_{I_k}, \mathbf{a}_{m_k}, \boldsymbol{\omega}_{m_k}) \tag{6}$$

In contrast, the historical pose state $\mathbf{x}_{H_k}$ is static and does not evolve over time. Thus, we only need to linearize Eq. (6) to propagate the corresponding covariance matrix, $\mathbf{P}_{I_k}$, of the IMU state, as follows:

$$\mathbf{P}_{I_{k+1}} = \boldsymbol{\Phi}_I(t_{k+1}, t_k)\mathbf{P}_{I_k}\boldsymbol{\Phi}_I(t_{k+1}, t_k)^\top + \mathbf{G}_k\mathbf{Q}_d\mathbf{G}_k^\top \tag{7}$$

where $[\mathbf{n}_g^\top\ \mathbf{n}_a^\top\ \mathbf{n}_{\omega g}^\top\ \mathbf{n}_{\omega a}^\top]^\top \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_d)$; $\mathbf{n}_{\omega g}$ and $\mathbf{n}_{\omega a}$ are the white Gaussian noises of gyroscope and accelerometer bias random walk model; $\boldsymbol{\Phi}_I(t_{k+1}, t_k)$ and $\mathbf{G}_k$ are the Jacobians of $f(\cdot)$ respect to the state and noise vector, respectively, which can be found in (Hesch et al. 2012).

3.2 Camera

Consider a 3D feature is detected from an arbitrary camera image at time t_k, whose uv measurement (i.e. the corresponding pixel coordinates) on the image plane is given by (see Geneva et al. (2020a)):

$$\mathbf{z}_C = \mathbf{h}_C(\mathbf{x}_k) + \mathbf{n}_C \tag{8}$$

$$= \mathbf{h}_d(\mathbf{h}_\rho(\mathbf{h}_t({}^{E}\mathbf{p}_F,\ {}^{C_k}_{E}\mathbf{R},\ {}^{E}\mathbf{p}_{C_k})),\ \mathbf{x}_{CI}) + \mathbf{n}_C \tag{9}$$

where $\mathbf{n}_C$ is the zero mean white Gaussian noise; $\mathbf{z}_n$ is the normalized undistorted uv measurement; $\mathbf{x}_{CI}$ is the camera intrinsic parameters such as focal length and distortion parameters; ${}^{E}\mathbf{p}_F$ is the feature position in global; $({}^{C_k}_{E}\mathbf{R},\ {}^{E}\mathbf{p}_{C_k})$ denotes the current camera pose in the global. In the above expression, we decompose the measurement function into multiple concatenated functions corresponding to different operations, which map the states into the raw uv measurement on the image plane. Each function (i.e. $\mathbf{h}_d,\ \mathbf{h}_\rho,\ \mathbf{h}_t$) is explained in the following:

3.2.1 Distortion Function $\mathbf{h}_d$ To get the normalized coordinates of the 3D feature on the image plane $\mathbf{z}_n = [x_n \ y_n]^\top$, we apply a distortion model which depends on the camera lens type. To be more specific, MINS supports radial-tangential and fisheye camera models (Kannala and Brandt 2006) which map the normalized coordinates into the raw pixel coordinates. As an example, we employ distortion function $\mathbf{h}_d$ with the radial-tangential model:

$$\begin{bmatrix} u \\ v \end{bmatrix} := \mathbf{z}_C = \mathbf{h}_d(\mathbf{z}_n, \mathbf{x}_{CI}) = \begin{bmatrix} f_x x + c_x \\ f_y y + c_y \end{bmatrix} \tag{10}$$

where

$$x = x_n(1 + k_1 r^2 + k_2 r^4) + 2p_1 x_n y_n + p_2(r^2 + 2x_n^2) \tag{11}$$
$$y = y_n(1 + k_1 r^2 + k_2 r^4) + p_1(r^2 + 2y_n^2) + 2p_2 x_n y_n \tag{12}$$
$$r^2 = x_n^2 + y_n^2 \tag{13}$$
$$\mathbf{x}_{CI} = (f_x, \ f_y, \ c_x, \ c_y, \ k_1, \ k_2, \ p_1, \ p_2) \tag{14}$$

See Kannala and Brandt (2006) for the camera intrinsic parameter $\mathbf{x}_{CI}$ definition.

3.2.2 Perspective Projection Function $\mathbf{h}_\rho$ The standard pinhole camera model is used to project a 3D point ${}^{C_k}\mathbf{p}_F = \begin{bmatrix} {}^{C_k}x & {}^{C_k}y & {}^{C_k}z \end{bmatrix}^\top$ into the normalized image plane (with unit depth):

$$\mathbf{z}_n = \mathbf{h}_\rho({}^{C_k}\mathbf{p}_F) = \begin{bmatrix} {}^{C_k}x/{}^{C_k}z \\ {}^{C_k}y/{}^{C_k}z \end{bmatrix} \tag{15}$$

3.2.3 Euclidean Transformation $\mathbf{h}_t$ We employ the 6DOF rigid-body Euclidean transformation to transform the 3D feature position in $\{E\}$ to the current camera frame $\{C_k\}$ based on the current camera pose:

$${}^{C_k}\mathbf{p}_F = \mathbf{h}_t({}^{E}\mathbf{p}_F, \ {}^{C_k}_{E}\mathbf{R}, \ {}^{E}\mathbf{p}_{C_k}) \tag{16}$$
$$= {}^{C_k}_{E}\mathbf{R}({}^{E}\mathbf{p}_F - {}^{E}\mathbf{p}_{C_k}) \tag{17}$$

where we in turn represent the camera pose $({}^{C_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{C_k})$ using camera extrinsic $\mathbf{x}_{CE} = ({}^{C}_{I}\mathbf{R}, \ {}^{C}\mathbf{p}_I)$ and IMU pose $({}^{I_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{I_k})$:

$${}^{C_k}_{E}\mathbf{R} = {}^{C}_{I}\mathbf{R}\,{}^{I_k}_{E}\mathbf{R} \tag{18}$$
$${}^{E}\mathbf{p}_{C_k} = {}^{E}\mathbf{p}_{I_k} + {}^{E}_{I_k}\mathbf{R}\,{}^{I}\mathbf{p}_C \tag{19}$$

3.2.4 MSCKF Update To perform the MSCKF update (Mourikis and Roumeliotis 2007), we linearize the measurement function (see Eq. (8)) at current state estimate $\hat{\mathbf{x}}_k$ and ${}^{E}\hat{\mathbf{p}}_F$ to get the following residual:

$$\tilde{\mathbf{z}}_C := \mathbf{z}_C - \mathbf{h}_C(\hat{\mathbf{x}}_k, {}^{E}\hat{\mathbf{p}}_F) \tag{20}$$
$$\approx \mathbf{H}_C\tilde{\mathbf{x}}_k + \mathbf{H}_F\,{}^{E}\tilde{\mathbf{p}}_F + \mathbf{n}_C \tag{21}$$

where $\mathbf{H}_{C_k}$ and $\mathbf{H}_F$ are the Jacobian matrix of $\mathbf{h}_C(\cdot)$ in respect to $\hat{\mathbf{x}}_k$ and ${}^{E}\hat{\mathbf{p}}_F$. The detailed derivation of these Jacobians can be found in Appendix A. After stacking the Jacobians and residuals for all camera measurements, we find the left nullspace of $\mathbf{H}_F$ and project Eq. (21) onto the nullspace to obtain a new measurement residual that is independent of the feature position:

$$\tilde{\mathbf{z}}'_C = \mathbf{H}'_C\tilde{\mathbf{x}}_k + \mathbf{n}'_C \tag{22}$$

Then we perform measurement compression (Golub and Van Loan (2013), Algorithm 5.2.4) which leads to substantial computational savings before the EKF update.

3.3 Wheel Encoder

2D wheel encoder measurements are commonplace on a ground vehicle that is often driven by two differential wheels (left and right) mounted on a common axis (baselink), each equipped with an encoder providing local angular rate readings (Siegwart et al. 2011):

$$\omega_{ml_k} = \omega_{l_k} + n_{\omega_l}, \quad \omega_{mr_k} = \omega_{r_k} + n_{\omega_r} \tag{23}$$

where ω_{l_k} and ω_{r_k} are the true angular velocities of each wheel at time t_k, and n_{ω_l} and n_{ω_r} are the corresponding zero-mean white Gaussian noises. These encoder readings can be combined to provide 2D linear and angular velocities about odometer frame $\{O\}$ at the center of the baselink:

$${}^{O_k}\omega = (\omega_{r_k} r_r - \omega_{l_k} r_l)/b \ , \quad {}^{O_k}v = (\omega_{r_k} r_r + \omega_{l_k} r_l)/2 \tag{24}$$

where $\mathbf{x}_{OI} := [r_l \ r_r \ b]^\top$ are the left and right wheel radii and the baselink length, respectively. Note that instead of (23), different forms of wheel encoder's measurements might be used, for example, linear velocities of the left and right wheels (i.e. wheel radii have been taken into account), or linear and angular velocities directly of the baselink.

3.3.1 Wheel Odometry Integration As the wheel encoders typically provide measurements of a high rate (e.g. 100-500 Hz), it would be too expensive to perform EKF updates at their rate. On the other hand, as the state (see Eq. (1)) has the historical poses, we naturally integrate the wheel odometry measurements (see Eq. (24)) between the two latest poses and then use this integrated 2D motion measurement for the update.

Consider integrating wheel odometry measurements between two IMU times t_{k-1} and t_k. The continuous-time 2D kinematic model for $t_\tau \in [t_{k-1}, t_k]$ is given by:

$$\begin{bmatrix} {}^{O_\tau}_{O_{k-1}}\dot{\theta} \\ {}^{O_{k-1}}\dot{x}_{O_\tau} \\ {}^{O_{k-1}}\dot{y}_{O_\tau} \end{bmatrix} := \begin{bmatrix} -{}^{O_\tau}\omega \\ {}^{O_\tau}v\cos({}^{O_{k-1}}_{O_\tau}\theta) \\ {}^{O_\tau}v\sin({}^{O_{k-1}}_{O_\tau}\theta) \end{bmatrix} = \begin{bmatrix} -{}^{O_\tau}\omega \\ {}^{O_\tau}v\cos({}^{O_\tau}_{O_{k-1}}\theta) \\ -{}^{O_\tau}v\sin({}^{O_\tau}_{O_{k-1}}\theta) \end{bmatrix} \tag{25}$$

where ${}^{O_\tau}_{O_{k-1}}\theta$ is the local yaw angle, ${}^{O_{k-1}}x_{O_\tau}$ and ${}^{O_{k-1}}y_{O_\tau}$ are the 2D position of $\{O_\tau\}$ in the starting integration frame $\{O_{k-1}\}$. Note that we use $-{}^{O_\tau}\omega$ and $-{}^{O_\tau}v\sin({}^{O_\tau}_{O_{k-1}}\theta)$ because we follow global-to-local orientation representation. We then integrate the model from t_{k-1} to t_k and obtain the 2D relative pose measurement as follows:

$$\mathbf{z}_O =: \begin{bmatrix} {}^{O_k}_{O_{k-1}}\theta \\ {}^{O_{k-1}}\mathbf{d}_{O_k} \end{bmatrix} = \begin{bmatrix} -\int_{t_{k-1}}^{t_k} {}^{O_t}\omega dt \\ \int_{t_{k-1}}^{t_k} {}^{O_t}v\cos({}^{O_t}_{O_{k-1}}\theta)dt \\ -\int_{t_{k-1}}^{t_k} {}^{O_t}v\sin({}^{O_t}_{O_{k-1}}\theta)dt \end{bmatrix} \tag{26}$$
$$=: \mathbf{g}_O(\{\omega_l, \omega_r\}_{k-1:k}, \mathbf{x}_{OI}) \tag{27}$$

where $\{\omega_l, \omega_r\}_{k-1:k}$ denote all the wheel angular velocities integrated between t_{k-1} and t_k.

3.3.2 Measurement Update Note that the integrated wheel measurement, Eq. (27), provides *only* the 2D relative motion on the odometer's plane, while the state (see Eq. (1)) contains the 3D poses. The measurement can be expressed as a function of the relative pose of the odometer frame by

projection:

$$\mathbf{z}_O = \mathbf{h}_O(\mathbf{x}_k) := \begin{bmatrix} \mathbf{e}_3^\top \mathrm{Log}({}^{O_k}_{E}\mathbf{R}\, {}^{E}_{O_{k-1}}\mathbf{R}) \\ \Lambda\, {}^{O_{k-1}}_{E}\mathbf{R}({}^{E}\mathbf{p}_{O_k} - {}^{E}\mathbf{p}_{O_{k-1}}) \end{bmatrix} \tag{28}$$

where $\Lambda = [\mathbf{e}_1 \ \mathbf{e}_2]^\top$, $\mathbf{e}_i$ is the i-th standard unit basis vector, and $\mathrm{Log}(\cdot)$ is the $SO(3)$ matrix logarithm function (Chirikjian 2011). The odometry pose $({}^{O_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{O_k})$ can be derived with IMU pose $({}^{I_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{I_k})$ and extrinsic $({}^{O}_{I}\mathbf{R}, {}^{O}\mathbf{p}_I)$:

$${}^{O_k}_{E}\mathbf{R} = {}^{O}_{I}\mathbf{R}\, {}^{I_k}_{E}\mathbf{R} \tag{29}$$

$${}^{E}\mathbf{p}_{O_k} = {}^{E}\mathbf{p}_{I_k} + {}^{E}_{I_k}\mathbf{R}\, {}^{I}\mathbf{p}_O \tag{30}$$

At this point, we have obtained the integrated wheel odometry measurements along with their corresponding Jacobians which are readily used for the EKF update after linearization:

$$\tilde{\mathbf{z}}_O := \mathbf{g}_O(\{\omega_{ml}, \omega_{mr}\}_{k\text{-}1:k}, \hat{\mathbf{x}}_{OI}) - \mathbf{h}_O(\hat{\mathbf{x}}_k) \tag{31}$$

$$\approx \mathbf{H}_O \tilde{\mathbf{x}}_k + \mathbf{n}_O \tag{32}$$

where $\mathbf{H}_O$ is the Jacobian matrix of $(\mathbf{h}_O(\cdot) - \mathbf{g}_O(\cdot))$ in respect to the state $\hat{\mathbf{x}}_k$ and $\mathbf{n}_O$ is the zero-mean white Gaussian noise. Detailed derivations of $\mathbf{H}_O$ and $\mathbf{n}_O$ can be found in Appendix B.

3.4 LiDAR

LiDAR provides 3D point clouds of the surroundings. Given a new measurement point cloud in LiDAR frame $\{L\}$ at time t_k, for each point ${}^{L_k}\mathbf{p}_F$ we transform it to a local map frame $\{M\}$ and find a number of neighboring points, ${}^{M}\mathbf{p}_{n_i} = [x_{n_i} \ y_{n_i} \ z_{n_i}]^\top, i \in \{1, \cdots, m\}$. We compute the plane ${}^{M}\boldsymbol{\Pi} = [a \ b \ c]^\top$ where the neighboring points are residing as follows:

$$\underbrace{\begin{bmatrix} x_{n_1} & y_{n_1} & z_{n_1} \\ \vdots & \vdots & \vdots \\ x_{n_m} & y_{n_m} & z_{n_m} \end{bmatrix}}_{\mathbf{A}_{m\times 3}} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \underbrace{\begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}}_{\mathbf{b}_{m\times 1}} \tag{33}$$

Its linear least-square solution is given by ${}^{M}\boldsymbol{\Pi} = (\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{b}$. After finding the plane ${}^{M}\boldsymbol{\Pi}$, we formulate the following point-on-plane measurement for all the planar points including ${}^{L_k}\mathbf{p}_F$ and ${}^{M}\mathbf{p}_{n_i}$:

$$\mathbf{z}_L := \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} = \mathbf{h}_L(\mathbf{x}_k) = \begin{bmatrix} {}^{M}\boldsymbol{\Pi}^\top {}^{M}\mathbf{p}_{n_1} - 1 \\ \vdots \\ {}^{M}\boldsymbol{\Pi}^\top {}^{M}\mathbf{p}_{n_m} - 1 \\ {}^{M}\boldsymbol{\Pi}^\top {}^{M}\mathbf{p}_F - 1 \end{bmatrix} \tag{34}$$

where ${}^{M}\mathbf{p}_F$ can be computed by transforming ${}^{L_k}\mathbf{p}_F$ to the map frame using the IMU-LiDAR extrinsic calibration $\mathbf{x}_L = ({}^{L}_{I}\mathbf{R}, {}^{L}\mathbf{p}_I)$, the IMU pose $({}^{I_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{I_k})$, and the map pose $({}^{M}_{E}\mathbf{R}, {}^{E}\mathbf{p}_M)$:

$${}^{M}\mathbf{p}_F = {}^{M}_{E}\mathbf{R}({}^{E}\mathbf{p}_{I_k} + {}^{E}_{I_k}\mathbf{R}({}^{I}\mathbf{p}_L + {}^{I}_{L}\mathbf{R}\, {}^{L_k}\mathbf{p}_F) - {}^{E}\mathbf{p}_M) \tag{35}$$

To perform EKF update, we linearize Eq. (34) and have the following residual:

$$\tilde{\mathbf{z}}_L = \begin{bmatrix} -{}^{M}\hat{\boldsymbol{\Pi}}^\top {}^{M}\hat{\mathbf{p}}_{n_1} + 1 \\ \vdots \\ -{}^{M}\hat{\boldsymbol{\Pi}}^\top {}^{M}\hat{\mathbf{p}}_{n_m} + 1 \\ -{}^{M}\hat{\boldsymbol{\Pi}}^\top {}^{M}\hat{\mathbf{p}}_F + 1 \end{bmatrix} \tag{36}$$

$$\approx \mathbf{H}_L \tilde{\mathbf{x}}_k + \mathbf{H}_{\boldsymbol{\Pi}}\, {}^{M}\tilde{\boldsymbol{\Pi}} + \mathbf{n}_L \tag{37}$$

where $\mathbf{H}_L$ and $\mathbf{H}_{\boldsymbol{\Pi}}$ are the Jacobian matrix of $\mathbf{h}_L(\cdot)$ in respect to the state $\mathbf{x}_k$ and the plane ${}^{M}\boldsymbol{\Pi}$; $\mathbf{n}_L$ is zero-mean Gaussian noise. The detailed derivations of the Jacobians and the noise can be found in Appendix C.

In analogy to the visual feature marginalization in the MSCKF, we now perform null space projection of Eq. (37) onto the left nullspace of $\mathbf{H}_{\boldsymbol{\Pi}}$ to reduce the computation and bound the size of the state. Note that it is possible that not all the points ${}^{M}\mathbf{p}_{n_i}$ are actually on the plane. We therefore only pass the plane to the next step after checking the condition number of the matrix $\mathbf{A}$ and the point-to-plane distance of each points ${}^{M}\mathbf{p}_{n_i}$ on ${}^{M}\boldsymbol{\Pi}$. Also note that this process is actually equivalent to initialize ${}^{M}\boldsymbol{\Pi}$ in the state with infinite covariance, update with Eq. (37), and marginalize (Yang et al. 2017), thus allowing us to properly handle the uncertainty of the plane without explicitly computing it. Then we have the following measurement model dependent only on the state ready for the EKF update:

$$\tilde{\mathbf{z}}'_L = \mathbf{H}'_L \tilde{\mathbf{x}}_k + \mathbf{n}'_L \tag{38}$$

Note that we also perform measurement compression in the same way as in MSCKF-based VINS for further efficiency.

3.4.1 Local Mapping

As having a global map has major drawbacks as the map grows: (i) the computational complexity of tracking the uncertainty of the map points, (ii) the computational complexity of finding the matches between the map and a new point cloud, and (iii) the size of the memory to store all the map points, many existing approaches keep the map points sparse to slow down the map growth to mitigate these issues. However, the sparse map may break the measurement model assumptions. For example, the planes extracted from a sparse map of a forest would not accurately represent the plane where the measurement point is on. We hence chose to have a dense map to ensure accurate matching while keeping only a local map to address the aforementioned issues.

Specifically, we employ the ikd-tree (Cai et al. 2021) to manage the map points and keep them for a fixed temporal window. We assume the map frame $\{M\}$ as the LiDAR pose anchored at one of our historical IMU poses $\{I_i\}$ (see Eq. (3)). and its pose in the global frame, $({}^{M}_{E}\mathbf{R}, {}^{E}\mathbf{p}_M)$, can be represented using the extrinsic calibration $\mathbf{x}_L$:

$${}^{M}_{E}\mathbf{R} = {}^{L}_{I}\mathbf{R}\, {}^{I_i}_{E}\mathbf{R} \tag{39}$$

$${}^{E}\mathbf{p}_M = {}^{E}\mathbf{p}_{I_i} + {}^{E}_{I_i}\mathbf{R}\, {}^{I}\mathbf{p}_L \tag{40}$$

As the map frame can be represented with our historical IMU pose and the LiDAR extrinsics, there is no need to add the map pose in the state, which saves computation. Note that the map frame should change over time as the MSCKF marginalizes the old IMU poses. In this case, we change the map frame anchored to the latest IMU frame (e.g. $\{I_j\}$) using the following relative pose:

$${}^{M_j}_{M_i}\mathbf{R} = {}^{L}_{I}\mathbf{R}\, {}^{I_j}_{E}\mathbf{R}\, {}^{E}_{I_i}\mathbf{R}\, {}^{I}_{L}\mathbf{R} \tag{41}$$

$${}^{M_j}\mathbf{p}_{M_i} = {}^{L}_{I}\mathbf{R}\, {}^{I_j}_{E}\mathbf{R}({}^{E}\mathbf{p}_{I_j} - {}^{E}\mathbf{p}_{I_i} + ({}^{E}_{I_j}\mathbf{R} - {}^{E}_{I_i}\mathbf{R})\, {}^{I}\mathbf{p}_L) \tag{42}$$

After using a new point cloud to update the state, we register the points ${}^{L_k}\mathbf{p}_F$ in the map by transforming them to the map

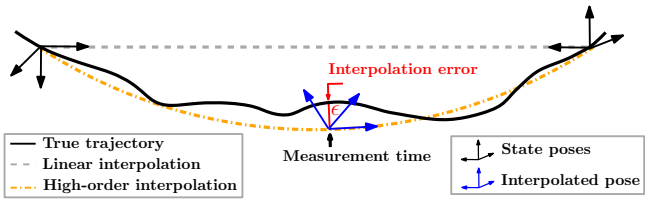


Figure 3. Different interpolation methods.

frame (see Eq. (35)). By doing so, the MINS estimator can maintain dense local maps, enabling consistent point-cloud matching and reducing outliers.

3.5 GNSS

A GNSS receiver such as GPS provides latitude, longitude, and altitude readings in a geodetic coordinate frame, which are typically converted to a local ENU or NED frame $\{E\}$ for outdoor navigation, e.g. by simply setting the first measurement location as the datum or using a base station. A GNSS measurement $\mathbf{z}_G$ at time t_k is the receiver's global position ${}^{E}\mathbf{p}_{G_k}$ – we here do not consider deeply-coupled fusion where pseudo-range and/or carrier phase measurements are used – which can be modeled with the IMU pose $({}^{I_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{I_k})$ and the extrinsic calibration ${}^{I}\mathbf{p}_G$:

$$\mathbf{z}_G = \mathbf{h}_G(\mathbf{x}_k) = {}^{E}\mathbf{p}_{G_k} + \mathbf{n}_G \tag{43}$$

$$= {}^{E}\mathbf{p}_{I_k} + {}^{E}_{I_k}\mathbf{R}\,{}^{I}\mathbf{p}_G + \mathbf{n}_G \tag{44}$$

where $\mathbf{n}_G$ is a white Gaussian noise. Now we linearize this measurement to perform the EKF update:

$$\tilde{\mathbf{z}}_G := \mathbf{z}_G - \mathbf{h}_G(\hat{\mathbf{x}}_k) \approx \mathbf{H}_G\tilde{\mathbf{x}}_k + \mathbf{n}_G \tag{45}$$

where $\mathbf{H}_G$ is the Jacobian matrix of $\mathbf{h}_G(\cdot)$ in respect to $\mathbf{x}_k$ (see Appendix D.1).

Note that the proposed MINS also supports general global pose measurements, which may be provided by sensors such as other estimators running independently or loop closure constraints. Assuming that the measurement carries information of a sensor X pose $({}^{X}_{E}\mathbf{R}, {}^{E}\mathbf{p}_X)$, we model the measurement using the IMU pose $({}^{I_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{I_k})$ and the corresponding extrinsic calibration $({}^{X}_{I}\mathbf{R}, {}^{X}\mathbf{p}_I)$:

$$\mathbf{z}_X := \begin{bmatrix} {}^{X_k}_{E}\boldsymbol{\theta} \\ {}^{E}\mathbf{p}_{X_k} \end{bmatrix} = \mathbf{h}_X(\mathbf{x}_k) = \begin{bmatrix} \mathrm{Log}({}^{X}_{I}\mathbf{R}\,{}^{I_k}_{E}\mathbf{R}) \\ {}^{E}\mathbf{p}_{I_k} + {}^{E}_{I_k}\mathbf{R}\,{}^{I}\mathbf{p}_X \end{bmatrix} + \mathbf{n}_X \tag{46}$$

where $\mathbf{n}_X$ is the zero-mean Gaussian noise. We again linearize this global-pose measurement for the EKF update:

$$\tilde{\mathbf{z}}_X := \mathbf{z}_X - \mathbf{h}_X(\hat{\mathbf{x}}_k) \approx \mathbf{H}_X\tilde{\mathbf{x}}_k + \mathbf{n}_X \tag{47}$$

where $\mathbf{H}_X$ is the Jacobian matrix (see Appendix E).

4 Adaptive On-Manifold Interpolation

Different sensors sample at different rates, such as a camera at 30 Hz and a LiDAR at 10 Hz, and their measurements are in general asynchronous if without (hardware) synchronization. This would incur difficulty to establish their measurement models as the corresponding states (of the IMU poses) are not available at the exact measurement times (see Eq. (1)). A naive solution is to add IMU poses into the state at every sampling time of each sensor, which would increase the size of the state and incur significant computational cost.

In contrast, we propose to represent the 6DOF poses with on-manifold interpolation and do not require any additional state augmentation. In our prior work (Lee et al. 2020b, 2021), we employed the 1st-order linear interpolation to handle asynchronous and delayed measurements:

$${}^{I_k}_{E}\mathbf{R} = \mathrm{Exp}(\lambda\mathrm{Log}({}^{I_b}_{E}\mathbf{R}\,{}^{E}_{I_a}\mathbf{R}))\,{}^{I_a}_{E}\mathbf{R} \tag{48}$$

$${}^{E}\mathbf{p}_{I_k} = (1-\lambda)\,{}^{E}\mathbf{p}_{I_a} + \lambda\,{}^{E}\mathbf{p}_{I_b} \tag{49}$$

$$\lambda = (t_k + {}^{X}t_I - t_a)/(t_b - t_a) \tag{50}$$

where the pose $({}^{I_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{I_k})$ at measurement time t_k was represented by two bounding IMU poses ($t_a \leq (t_k + {}^{X}t_I) \leq t_b$), ${}^{X}t_I$ is the sensor time offset, and $\mathrm{Exp}(\cdot)$ is the $SO(3)$ matrix exponential function (Chirikjian 2011). The Jacobians of the interpolated IMU pose with respect to each bounding pose can be found in Appendix F.

However, this linear interpolation relies on the assumption of constant linear and angular velocity, which may not hold in the case of highly-dynamic motion or for some sensitive sensors, causing inconsistency to the estimator due to interpolation errors, as illustrated in Fig. 3. As such, the proposed MINS adopts a high-order interpolation method to capture more complex motion profiles in practice.

Specifically, we represent the IMU pose $({}^{I_k}_{E}\mathbf{R}, {}^{E}\mathbf{p}_{I_k})$ at time t_k with a polynomial of degree n (Eckenhoff et al. 2021):

$${}^{I_k}_{E}\mathbf{R} = \mathrm{Exp}\Big(\sum_{i=1}^{n}\mathbf{a}_i\Delta t_k^i\Big)\,{}^{I_0}_{E}\mathbf{R} \tag{51}$$

$${}^{E}\mathbf{p}_{I_k} = {}^{E}\mathbf{p}_{I_0} + \sum_{i=1}^{n}\mathbf{b}_i\Delta t_k^i \tag{52}$$

where $\Delta t_k = t_k + {}^{X}t_I - t_0$, and $\mathbf{a}_i$ and $\mathbf{b}_i$ are the polynomial coefficients. To compute the coefficients, we fit the polynomial at a set of $n+1$ historical IMU poses ($\{I_j\}, j \in \{0, 1, \cdots, n\}$) that are closest to the measurement time t_k. Specifically, we compute the coefficients as follows:

$${}^{I_j}_{E}\mathbf{R} = \mathrm{Exp}\Big(\sum_{i=1}^{n}\mathbf{a}_i\delta t_j^i\Big)\,{}^{I_0}_{E}\mathbf{R}, \quad \forall j \in \{0, 1, \cdots, n\} \tag{53}$$

$$\Rightarrow \begin{bmatrix}\mathbf{a}_1 \\ \vdots \\ \mathbf{a}_n\end{bmatrix} = \begin{bmatrix}\delta t_1 & \dots & \delta t_1^n \\ \vdots & \ddots & \vdots \\ \delta t_n^n & \dots & \delta t_n^n\end{bmatrix}^{-1} \begin{bmatrix}\mathrm{Log}({}^{I_1}_{E}\mathbf{R}\,{}^{E}_{I_0}\mathbf{R}) \\ \vdots \\ \mathrm{Log}({}^{I_n}_{E}\mathbf{R}\,{}^{E}_{I_0}\mathbf{R})\end{bmatrix} \tag{54}$$

$${}^{E}\mathbf{p}_{I_j} = {}^{E}\mathbf{p}_{I_0} + \sum_{i=1}^{n}\mathbf{b}_i\delta t_j^i, \quad \forall j \in \{0, 1, \cdots, n\} \tag{55}$$

$$\Rightarrow \begin{bmatrix}\mathbf{b}_1 \\ \vdots \\ \mathbf{b}_n\end{bmatrix} = \begin{bmatrix}\delta t_1 & \dots & \delta t_1^n \\ \vdots & \ddots & \vdots \\ \delta t_n^n & \dots & \delta t_n^n\end{bmatrix}^{-1} \begin{bmatrix}{}^{E}\mathbf{p}_{I_1} - {}^{E}\mathbf{p}_{I_0} \\ \vdots \\ {}^{E}\mathbf{p}_{I_n} - {}^{E}\mathbf{p}_{I_0}\end{bmatrix} \tag{56}$$

where $\delta t_j = t_j - t_0$, The above equations show the relation between the interpolated pose and the $n+1$ poses in the state. Note that the interpolated pose is additionally a function of the unknown time offset, and thus we can further calibrate them in case needed. The Jacobians of the interpolated pose in respect to the $n+1$ poses and the time offset can be found in Appendix G.

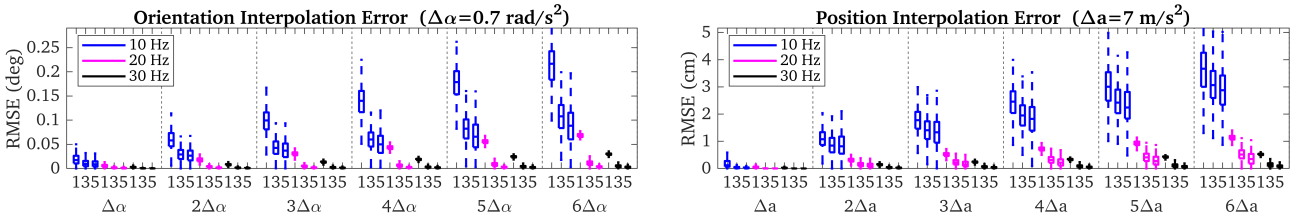


Figure 4. Interpolated pose errors depend on angular/linear accelerations (binned with $0.7\ rad/s^2$ and $7\ m/s^2$ resolution, respectively), interpolation order (1, 3, and 5), and temporal distance (10, 20, and 30 cloning Hz).

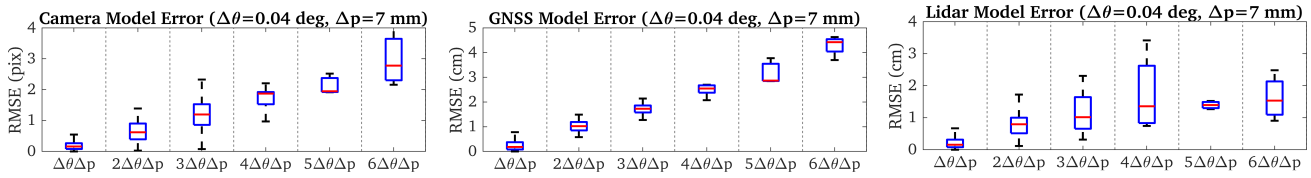


Figure 5. Sensor measurement model errors depend on interpolated pose errors (binned with $0.04\ deg$ and $7\ mm$ resolutions).

4.1 Numerical Analysis

The order of the interpolation polynomial is directly related to the accuracy and complexity of the interpolated pose (Cioffi et al. 2022). It is hard to analytically determine an optimal order of interpolation, as there are many factors affecting the interpolated pose, such as the interpolation distance or motion of the sensor platform (robot). To understand these factors, we numerically examine the error characteristics of the interpolated pose using realistic simulated motions and identify four key factors of interpolation errors including angular acceleration, linear acceleration, interpolation order, and temporal distance of interpolation. Fig. 4 depicts the box plots of the interpolation errors in different simulation setups. In particular, we found that the temporal distance of interpolation impacts the most on interpolation errors. The higher interpolation order has smaller interpolation errors, especially large accuracy gain between the first and third orders while having diminishing returns with higher orders. Interestingly, the interpolation error grows linearly with the increase of acceleration.

To understand the significance of the interpolation errors on different sensor measurement models, we use the interpolated poses, instead of the IMU poses, in the camera (9), GNSS (44), and LiDAR (34) measurement models in the numerical studies where the same sensor parameters as in the ensuing simulations are used as shown in Table 2. Note that the wheel measurement model is not considered as interpolation is not needed for the wheel-integrated measurements (see Section 3.3.1). Fig. 5 shows the errors of the sensor measurement models (see Eq. (20), Eq. (36), and Eq. (45)) with different levels of interpolation errors, which are binned with $0.04\ deg$ and $7\ mm$ resolution, respectively. It is clear from these results that the camera is sensitive to the interpolation error, as a small error can easily cause more than 1 pixel error which is generally the noise level of camera measurement (e.g. second column of the left figure of Fig. 5). Similarly, the LiDAR error easily exceeds 1 cm which is generally the noise level of LiDAR measurement, showing its sensitivity to the interpolation error. The interpolation errors do not impact much on the GNSS measurement model as single-point GPS measurement noise is around 1 m, which has orders of magnitude larger while it may become sensitive when using RTK-GPS with lower measurement noise. Sensor calibration can also affect these errors, although reasonable values are used in Table 2.

As evident, small interpolation errors can cause large errors for the measurement models and in turn may result in estimation failure. Based on the above extensive numerical studies, the proposed MINS by default chooses to employ the 3rd-order polynomials and 20 Hz cloning to suppress the interpolation error and balance the efficiency.

4.2 Incorporating Interpolation Error

While a higher-order setting (e.g. 3rd-order polynomials and 20 Hz cloning) may approximate the IMU pose more accurately, there always exists the interpolation error (see $\boldsymbol{\epsilon}$ in Fig. 3) that can hurt estimation performance. To address this issue, we explicitly incorporate the interpolation error into the interpolated pose in the measurement model:

$$\mathbf{z}_k = \mathbf{h}({}^{I_k}_{G}\mathbf{R}, {}^{G}\mathbf{p}_{I_k}) + \mathbf{n}_k \tag{57}$$

$$= \mathbf{h}(\mathbf{g}(\mathbf{x}) + \boldsymbol{\epsilon}) + \mathbf{n}_k \tag{58}$$

where $\mathbf{z}_k$ is the sensor measurement at t_k, $\mathbf{h}(\cdot)$ is the corresponding measurement model with respect to the IMU pose at t_k, $\mathbf{g}(\cdot)$ is the interpolation function, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_{\boldsymbol{\epsilon}})$ is the interpolation error modeled as zero-mean Gaussian, and $\mathbf{n}_k$ is the measurement noise. The linearized model in turn can be shown as:

$$\mathbf{r}_k := \mathbf{z}_k - \mathbf{h}(\mathbf{g}(\hat{\mathbf{x}})) = \frac{\partial \mathbf{h}}{\partial \mathbf{g}} \frac{\partial \mathbf{g}}{\partial \mathbf{x}} \tilde{\mathbf{x}} + \frac{\partial \mathbf{h}}{\partial \mathbf{g}} \boldsymbol{\epsilon} + \mathbf{n}_k \tag{59}$$

where the additional error term $\frac{\partial \mathbf{h}}{\partial \mathbf{g}} \boldsymbol{\epsilon}$ compensates for the interpolation error. To model the variance of the interpolation error, we leverage the preceding numerical findings. The orientation/position interpolation error is shown to have a linear relationship with angular/linear accelerations, while their slope coefficients depend on the cloning frequency and the interpolation order (see Fig. 4). We thus model the variance of the error as follows:

$$\mathbf{R}_{\boldsymbol{\epsilon}} = diag((\alpha \times s_o(\bar{c}, \bar{o}))^2 \mathbf{I}_3,\ (a \times s_p(\bar{c}, \bar{o}))^2 \mathbf{I}_3) \tag{60}$$

where α and a are the size of angular and linear accelerations, $s_o(\cdot)$ and $s_p(\cdot)$ are the slope coefficients of orientation and position given cloning frequency $\bar{c}$ and interpolation order $\bar{o}$. In practice, the proposed MINS has lookup tables ranging in cloning frequency between 4-30 Hz and interpolation order between 1-9 for both orientation and position to achieve the slopes directly.

4.3 Dynamic Cloning

As the interpolation error is shown to be proportional to the accelerations, we propose dynamic cloning to adaptively change the cloning frequency based on the robot's motion in order to achieve computational efficiency and maintain a comparable level of accuracy to the high fixed-rate cloning strategy. Specifically, we decrease the cloning frequency while the robot is in slow motion which will also reduce the total size of the state, and thus reduce the computation of the EKF update. On the other hand, when the robot undergoes dynamic motions, the low cloning frequency would cause large interpolation errors, thus we increase the cloning frequency to reduce them. To this end, we set thresholds for the variance model of the interpolation errors (see Eq. (60)) and find the lowest cloning frequency that passes the threshold:

$$\begin{aligned} \bar{c} &= \min \bar{C} \\ s.t.\ \ \alpha \times s_o(\bar{c}, \bar{o}) < \gamma_o,\ \ a \times s_p(\bar{c}, \bar{o}) < \gamma_p \end{aligned} \tag{61}$$

where $\bar{C}$ is the set of cloning frequencies, and γ_o and γ_p are the thresholds for orientation and position part, respectively. Note that the proposed MINS does not create a clone if there is no measurement between the last clone and the current desired clone time, i.e. clone up to 20 Hz if the maximum sensing rate is 20 Hz to avoid having redundant clones.

4.4 Online Calibration

One of the prerequisites to successfully run a multisensor fusion system is to have accurate sensor calibration as its failure will directly lead to low estimation performance or even divergence. Even if one precisely calibrates the sensor parameters offline, their values may change over time for environmental or mechanical reasons. Therefore, online calibration is necessary to achieve robustness to the poor calibration and improve localization performance.

To that end, we include the spatial extrinsics (see Eq. (16), (28), (35), (44)) and intrinsics (see Eq. (10), (26)) of each sensor when modeling its measurements. Also, the temporal extrinsics (time offset, see Eq. (50)) is naturally modeled while handling the asynchronicity of sensor measurement. To perform online calibration, we augment our state (see Eq. (1)) with these sensor calibration parameters and jointly estimate them. To be more specific, for the system that carries c-number of cameras, g-number of GNSS sensors, l-number of LiDAR sensors, and a pair of wheel encoders, we add sensor parameter state $\mathbf{x}_S$*:

$$\mathbf{x}_k = (\mathbf{x}_{I_k},\ \mathbf{x}_{H_k},\ \mathbf{x}_S) \tag{62}$$
$$\mathbf{x}_S = (\mathbf{x}_C,\ \mathbf{x}_G,\ \mathbf{x}_O,\ \mathbf{x}_L) \tag{63}$$
$$\mathbf{x}_C = (\mathbf{x}_{C_1},\ \cdots \mathbf{x}_{C_c}) \tag{64}$$
$$\mathbf{x}_{C_\tau} = (\mathbf{x}_{CE_\tau},\ \mathbf{x}_{CI_\tau})\ \forall \tau \in \{1, \cdots, c\} \tag{65}$$
$$\mathbf{x}_{CE_\tau} = ({}^{C_\tau}_{I}\mathbf{R},\ {}^{C_\tau}\mathbf{p}_I,\ {}^{C_\tau}t_I) \tag{66}$$
$$\mathbf{x}_{CI_\tau} = ({}^{\tau}f_x,\ {}^{\tau}f_y,\ {}^{\tau}c_x,\ {}^{\tau}c_y,\ {}^{\tau}k_1,\ {}^{\tau}k_2,\ {}^{\tau}p_1,\ {}^{\tau}p_2) \tag{67}$$
$$\mathbf{x}_G = (\mathbf{x}_{G_1},\ \cdots \mathbf{x}_{G_g}) \tag{68}$$
$$\mathbf{x}_{G_\tau} = ({}^{I}\mathbf{p}_{G_\tau},\ {}^{G_\tau}t_I)\ \forall \tau \in \{1, \cdots, g\} \tag{69}$$
$$\mathbf{x}_L = (\mathbf{x}_{L_1},\ \cdots,\ \mathbf{x}_{L_l}) \tag{70}$$
$$\mathbf{x}_{L_\tau} = ({}^{L_\tau}_{I}\mathbf{R},\ {}^{L_\tau}\mathbf{p}_I,\ {}^{L_\tau}t_I)\ \forall \tau \in \{1, \cdots, l\} \tag{71}$$
$$\mathbf{x}_O = (\mathbf{x}_{OE},\ \mathbf{x}_{OI}) \tag{72}$$
$$\mathbf{x}_{OE} = ({}^{O}_{I}\mathbf{R},\ {}^{O}\mathbf{p}_I,\ {}^{O}t_I) \tag{73}$$
$$\mathbf{x}_{OI} = (r_l,\ r_r,\ b) \tag{74}$$

where $\mathbf{x}_{CE}$ and $\mathbf{x}_{CI}$ are the camera extrinsics and intrinsics, $\mathbf{x}_G$ is the GNSS extrinsics, $\mathbf{x}_L$ is the LiDAR extrinsics, and $\mathbf{x}_{OE}$ and $\mathbf{x}_{OI}$ are the wheel extrinsics and intrinsics. Note that not all of these variables have to be included in the state, only those inaccurate ones are included and others can be treated as true in order to save computations if possible. Detailed derivations about the measurement Jacobians for the calibration parameters can be found in Appendix A.1, Appendix B.3, Appendix E, Appendix F, and Appendix G.

5 System Initialization

Initialization aims to compute state variables directly from available measurements to start estimation. An aided-INS estimator is typically required to initialize at least 11 DOF navigation states including roll, pitch, linear velocity, and the biases of the gyroscope and accelerometer. If the sensor platform (robot) starts from a stationary position, the initialization becomes relatively easy and can be solved by averaging the IMU measurements (Lin and Zhang 2022; Shan et al. 2021). However, in many scenarios, the estimator is required to perform dynamic initialization when the robot is in motion. This problem in VINS is often addressed by aligning visual structure-from-motion trajectory to the inertial preintegration measurements (Qin and Shen 2017; Qin et al. 2018), while Dong-Si and Mourikis (2011) formulated and solved a quadratic programming (QP) problem with quadratic constraints. In order to successfully initialize the estimator, these methods assume fully excited 3D motions which may not be possible for motion-constrained systems. For example, the VINS estimator may lose scale information when a ground vehicle moves straight with constant acceleration (Wu et al. 2017). While LiDAR-based methods are more robust to these problems as they can directly recover 3D pose information through 3D point-cloud matching (e.g. iterated closest points, or ICP (Yuan et al. 2022, 2023)), their performance heavily relies on the matching quality, which could be challenging when the robot undergoes dynamic motions or there are not many overlapping points between scans. On the other hand, for ground vehicles, wheel encoders can be leveraged to initialize the state, which appears to be straightforward but has not been sufficiently investigated in the literature. As the wheel-encoder can provide standalone odometry, most of the existing methods integrate wheel-encoder readings to achieve pose information and use it to initialize gyroscope

*We reused some of the notations introduced before to describe an arbitrary number set of the sensor calibration parameters.

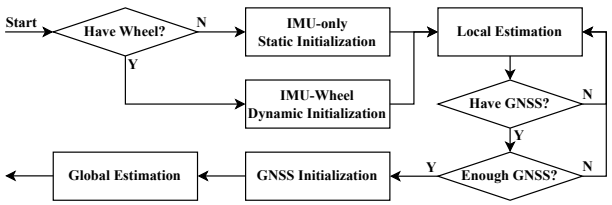


Figure 6. The proposed MINS initialization procedure.

bias (Liu et al. 2019; Gang et al. 2020), scale (Jiang et al. 2021; Feng 2021), or linear velocity (Gang et al. 2020; Hou et al. 2022). As these methods solve the initialization problem by setting relative-pose constraints which require double integration of the IMU and single integration of wheel-encoder readings, they may fail due to high nonlinearity.

To address these issues and enable robust estimation, the proposed MINS supports not only the IMU-only static initialization (Geneva et al. 2020a) but the IMU-wheel dynamic initialization which requires single IMU integration and no wheel integration. After the initialization, we may perform navigation in the local frame $\{W\}$ until GNSS measurements are available to initialize the system in the global frame $\{E\}$. Fig. 6 shows the overall procedure of the proposed MINS initialization.

5.1 IMU-Wheel Dynamic Initialization

To initialize, both IMU and wheel measurements are collected for a short period of time (e.g. less than 0.2 secs) and the angular and linear velocities of the wheel frame are pre-computed (see Eq. (24)). Then, $\mathbf{b}_g$ can be recovered as:

$$\mathbf{b}_g = {}^{I}\boldsymbol{\omega}_{avg} - {}^{I}_{O}\mathbf{R}\,{}^{O}\boldsymbol{\omega}_{avg} \tag{75}$$

where ${}^{O}\boldsymbol{\omega}_{avg}$ and ${}^{I}\boldsymbol{\omega}_{avg}$ are the average angular velocity of the wheel and IMU frames. Next, we can compute the linear velocity of the IMU frame ${}^{I_0}\mathbf{v}_{I_k}$ in the following two ways:

$$\begin{aligned} {}^{I_0}\mathbf{v}_{I_k} &= {}^{I_0}\mathbf{v} + \sum_{i=0}^{k-1}({}^{I_0}_{I_i}\mathbf{R}\,{}^{I_i}\mathbf{a} - {}^{I_0}_{I_i}\mathbf{R}\mathbf{b}_a - {}^{I_0}\mathbf{g})\Delta t_i \qquad (76)\\ &= {}^{I_0}_{I_k}\mathbf{R}\,{}^{I}_{O}\mathbf{R}({}^{O_k}\mathbf{v} + \lfloor {}^{O_k}\boldsymbol{\omega} \rfloor {}^{O}\mathbf{p}_I) \qquad (77)\end{aligned}$$

where $\{I_0\}$ is the first IMU frame among collected measurements and $\Delta t_i := t_i - t_{i-1}$ is the integration period. Eq. (76) computes ${}^{I_0}\mathbf{v}_{I_k}$ by first-order IMU integration and Eq. (77) computes the same property directly from the wheel velocities. Based on the above relations, the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is created to recover the gravity in the first IMU frame and the accelerometer bias:

$$\underbrace{\begin{bmatrix} -\sum_{i=0}^{k-1} {}^{I_0}_{I_i}\mathbf{R}\Delta t_i & -\sum_{i=0}^{k-1}\mathbf{I}_3\Delta t_i \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \mathbf{b}_a \\ {}^{I_0}\mathbf{g} \end{bmatrix}}_{\mathbf{x}} \tag{78}$$

$$= \underbrace{\begin{bmatrix} {}^{I_0}_{O_k}\mathbf{R}({}^{O_k}\mathbf{v} + \lfloor {}^{O_k}\boldsymbol{\omega} \rfloor {}^{O}\mathbf{p}_I) - {}^{I_0}\mathbf{v} - \sum_{i=0}^{k-1} {}^{I_0}_{I_i}\mathbf{R}\,{}^{I_i}\mathbf{a}\Delta t_i \end{bmatrix}}_{\mathbf{b}}$$

In analogy to (Dong-Si and Mourikis 2011), we can solve the above QP problem assuming the magnitude of the gravity is known. Finally, the initial rotation ${}^{I_0}_{W}\mathbf{R}$ is computed with the Gram-Schmidt process using ${}^{I_0}\mathbf{g}$. At this point, we have initialized the navigation state, including the orientation $({}^{I_0}_{W}\mathbf{R})$, the biases $(\mathbf{b}_g, \mathbf{b}_a)$, the linear velocity ${}^{I_0}\mathbf{v}$, and the zero position in the local frame $\{W\}$.

5.2 Local-Global Frame Transformation

The GNSS measurement model (44) assumes the IMU pose in the global frame $\{E\}$. However, before the GNSS initialization, the system performs navigation in the gravity-aligned local frame $\{W\}$. Thus, in order to process the GNSS measurements, the 4 DOF frame transformation $({}^{E}_{W}\mathbf{R}, {}^{E}\mathbf{p}_W)$ must be known (note that $\{E\}$ is also aligned with the gravity). To this end, we collect two sets of estimates of the GNSS receiver positions in $\{E\}$ and $\{W\}$ and formulate a non-linear optimization problem to align them. Note that if inaccurate GNSS measurements (e.g. single-point GPS) with large noise, this alignment based on a short trajectory may result in a poor transformation. However, the current sliding window of the MSCKF typically contains a most recent short trajectory and causes trouble in finding accurate local-global frame transformation. Therefore, we augment our state by selectively keeping the cloned poses (or keyframes) at the GNSS measurement times and performing initialization once we reach the desired trajectory length.

Specifically, given a set of the GNSS position measurements in the ENU frame $\{{}^{E}\mathbf{p}_{G_1}, \cdots, {}^{E}\mathbf{p}_{G_n}\}$ within the keyframe window and the corresponding interpolated positions in the local frame $\{{}^{W}\mathbf{p}_{G_1}, \cdots, {}^{W}\mathbf{p}_{G_n}\}$, we use the following geometric constraints to determine the frame transformation:

$$\begin{aligned} {}^{E}\mathbf{p}_{G_i} &= {}^{E}\mathbf{p}_W + {}^{E}_{W}\mathbf{R}\,{}^{W}\mathbf{p}_{G_i}, \forall i = 1\cdots n \qquad (79)\\ \Rightarrow {}^{E}\mathbf{p}_{G_j} - {}^{E}\mathbf{p}_{G_1} &= {}^{E}_{W}\mathbf{R}({}^{W}\mathbf{p}_{G_j} - {}^{W}\mathbf{p}_{G_1}), \forall j = 2\cdots n \qquad (80)\end{aligned}$$

Note that if we have more than one GNSS receivers, we also stack their measurements in Eq. (79) with the corresponding extrinsics to allow more robust initialization. Note also that due to the 4 DOF, instead of 6 DOF, transformation, we can use the rotation about the global yaw θ:

$${}^{E}_{W}\mathbf{R} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{81}$$

With this, Eq. (80) becomes:

$$\mathbf{A}_j \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} := \mathbf{A}_j\mathbf{w} = \mathbf{b}_j, \forall j = 2\cdots n \tag{82}$$

Stacking all these constraints yields the following linear least-squares with quadratic constraints:

$$\min_{\mathbf{w}} \ ||\mathbf{A}\mathbf{w} - \mathbf{b}||^2, \ \text{ s.t. } ||\mathbf{w}||^2 = 1 \tag{83}$$

We substitute the solution of Eq. (83) into Eq. (79) and solve for ${}^{E}\hat{\mathbf{p}}_W$:

$${}^{E}\hat{\mathbf{p}}_W = \frac{1}{n}\sum_{i=1}^{n}\left[{}^{E}\mathbf{p}_{G_i} - {}^{E}_{W}\hat{\mathbf{R}}\,{}^{W}\mathbf{p}_{G_i}\right] \tag{84}$$

This initial guess $({}^{E}_{W}\hat{\mathbf{R}}, {}^{E}\hat{\mathbf{p}}_W)$ is further refined with delayed initialization (Li 2014; DuToit et al. 2017). Specifically, by

augmenting the state vector with the transformation along with an *infinite* covariance prior to these new variables, we perform the EKF update with all the GNSS measurements:

$$\mathbf{z}_G = {}^E\mathbf{p}_W + {}^E_W\mathbf{R}({}^W\mathbf{p}_{I_k} + {}^W_{I_k}\mathbf{R}\,{}^I\mathbf{p}_G) + \mathbf{n}_G \tag{85}$$

After initialization, we marginalize all the keyframes to reduce the state to the original state size.

5.3 Global Navigation

As shown in our prior work (Lee et al. 2020a), the GNSS-aided INS in a local frame $\{W\}$ using the global-to-local transformation $({}^E_W\mathbf{R}, {}^E\mathbf{p}_W)$ to fuse GNSS measurements is not fully observable, although it gains global information.

Lemma 5.1. *If estimating states in the local frame, even with global GNSS measurements, the system remains unobservable and has four unobservable directions.*

Proof. See Appendix D.2.

While this result is somewhat counter-intuitive, the root cause of this unobservability is the gauge freedom of the 4DOF frame transformation between $\{W\}$ to $\{E\}$, which is inherited from INS (i.e. 4 unobservable directions (Hesch et al. 2012)). Even though we use the global measurements, the system has a non-trivial nullspace. As unobservable linearized systems may have erroneous nullspace due to improper linearization points, and the corresponding linearized estimator thus may gain information in spurious directions, causing inconsistency (Hesch et al. 2012; Huang et al. 2008; Chen et al. 2022a). To address this issue, we perform state estimation directly in the global frame once initialized, which is shown to make the system fully observable.

Lemma 5.2. *If estimating states in the global frame, the system is fully observable.*

Proof. See Appendix D.3.

After initialization, we hence transform the state from $\{W\}$ to $\{E\}$ as well as propagate the corresponding covariance. Specifically, we transform the state in local ${}^W\mathbf{x}_k$ to global ${}^E\mathbf{x}_k$ as:

$${}^E\mathbf{x}_k = \mathbf{g}({}^W\mathbf{x}_k, {}^E_W\mathbf{R}, {}^E\mathbf{p}_W) \tag{86}$$

where

$${}^{I_k}_E\mathbf{R} = {}^{I_k}_W\mathbf{R}\,{}^W_E\mathbf{R} \tag{87}$$

$${}^E\mathbf{p}_{I_k} = {}^E\mathbf{p}_W + {}^E_W\mathbf{R}\,{}^W\mathbf{p}_{I_k} \tag{88}$$

$${}^E\mathbf{v}_{I_k} = {}^E_W\mathbf{R}\,{}^W\mathbf{v}_{I_k} \tag{89}$$

Note that the above equations are used to warp all historical poses $\mathbf{x}_{H_k}$. Now we linearize Eq. (86) at the current estimate to get the Jacobian $\mathbf{\Psi}$ and propagate the covariance as follows:

$${}^E\tilde{\mathbf{x}}_k = \mathbf{\Psi}\,{}^W\tilde{\mathbf{x}}_k\,, \quad {}^E\mathbf{P}_k = \mathbf{\Psi}\,{}^W\mathbf{P}_k\mathbf{\Psi}^\top \tag{90}$$

where the detailed derivations of $\mathbf{\Psi}$ can be found in Appendix D.4. After transformation, we marginalize $({}^E_W\mathbf{R}, {}^E\mathbf{p}_W)$ from the state as they are no longer needed.

Table 2. Simulation setup parameters.

Parameter	Value	Parameter	Value
Clone Freq. (Hz)	20	Wheel Mode	*Wheel3DAng*
Window Size. (s)	1	Wheel Ang. Noise (rad/s)	1e-2
Intr. Order	3	LiDAR Freq. (Hz)	10
Dyn. Ori. Thresh (rad)	7e-3	# of LiDAR	2
Dyn. Pos. Thresh (m)	3e-3	LiDAR Noise (m)	1e-2
IMU Freq. (Hz)	200	${}^{C_1}_I\boldsymbol{\theta}$ (rad)	1.57, 0.00, 0.00
IMU Acc. Noise	2e-2	${}^{C_1}_I\mathbf{p}_I$ (m)	-0.01, 0.01, 0.01
IMU Acc. Bias	3e-2	${}^{C_2}_I\boldsymbol{\theta}$ (rad)	1.57, 0.00, 0.00
IMU Gyr. Noise	2e-3	${}^{C_2}_I\mathbf{p}_I$ (m)	0.01, 0.01, 0.01
IMU Gyr. Bias	2e-4	${}^I\mathbf{p}_{G_1}$ (m)	1.00, 1.00, 1.00
Cam Freq. (Hz)	30	${}^I\mathbf{p}_{G_2}$ (m)	-1.00, -1.00, -1.00
# of Cam	2	${}^O_I\boldsymbol{\theta}$ (rad)	0.00, 0.00, 0.00
Cam Noise (pix)	1	${}^O\mathbf{p}_I$ (m)	0.00, 0.00, 0.00
GNSS Freq. (Hz)	1	${}^{L_1}_I\boldsymbol{\theta}$ (rad)	0.26, 0.00, 0.00
# of GNSS	2	${}^{L_1}_I\mathbf{p}_I$ (m)	0.30, 0.30, 0.50
GNSS Noise (m)	0.1	${}^{L_2}_I\boldsymbol{\theta}$ (rad)	0.00, 1.57, 0.00
Wheel Freq. (Hz)	100	${}^{L_2}_I\mathbf{p}_I$ (m)	-0.01, -0.01, -0.01

* All sensor time offsets are set to 0.00 s.

Table 3. Orientation/position RMSE and NEES results of MINS with/without incorporating interpolation error model. The inconsistent results (NEES $>$ 4) are highlighted in red.

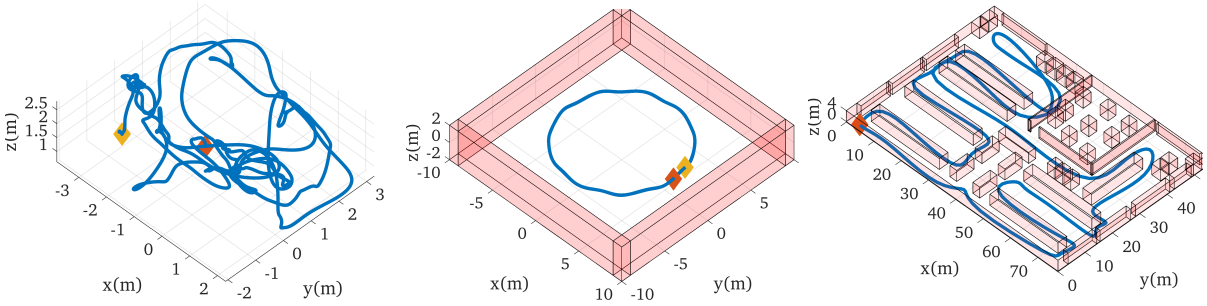
	Hz	RMSE (deg / m)	NEES
w Model	04	0.550 ± 0.084 / 0.061 ± 0.022	3.0 ± 1.1 / 2.1 ± 1.5
	06	0.459 ± 0.262 / 0.064 ± 0.014	3.9 ± 1.4 / 2.2 ± 1.0
	10	0.224 ± 0.032 / 0.036 ± 0.011	3.8 ± 0.8 / 1.4 ± 0.8
	20	0.172 ± 0.060 / 0.023 ± 0.012	3.9 ± 1.2 / 1.0 ± 0.9
	30	0.178 ± 0.110 / 0.019 ± 0.011	3.2 ± 1.3 / 0.8 ± 1.0
w/o Model	04	1.242 ± 1.084 / 1.479 ± 1.113	2e1 $\pm$ 2e1 / 3e3 $\pm$ 2e3
	06	1.982 ± 0.815 / 0.631 ± 0.632	3e1 $\pm$ 2e1 / 2e3 $\pm$ 2e3
	10	0.301 ± 0.106 / 0.033 ± 0.028	5.1 ± 4.0 / 2.0 ± 2.0
	20	0.200 ± 0.127 / 0.024 ± 0.018	4.5 ± 1.2 / 1.1 ± 1.0
	30	0.178 ± 0.110 / 0.019 ± 0.011	3.2 ± 1.3 / 0.8 ± 0.6

6 Simulation Results

We perform extensive Monte-Carlo simulation tests to validate the proposed MINS estimator. Built on top of the OpenVINS simulator (Geneva et al. 2019, 2018b), our MINS simulator generates synthetic measurements of IMU, camera, GNSS, wheel encoder and LiDAR based on the realistic trajectories as shown in Fig. 7. Table 2 summarizes the setup parameters used in the ensuing simulations.

6.1 Effects of Interpolation Errors

To verify our proposed interpolation error model, we tested the proposed MINS with and without incorporating the error model at different cloning frequencies (4, 6, 10, 20, and 30 Hz) on the simulated *EuRoC Vicon Room2 02* (Fig. 7a). We simulated a stereo camera and IMU for this test as the camera processes multiple measurements tracked over time which requires the interpolation of all over the MSCKF window. Table 3 shows the orientation and position root mean squared error (RMSE) and normalized estimation error-squared (NEES) results of each cloning frequency averaged over 10 Monte-Carlo runs. Note that the 30 Hz cloning frequency results of both with and without the error model are the same because no interpolation was performed in both cases as the clone frequency and camera measurement rate are the same.



(a) *EuRoc Vicon Room2 02* **(b)** *UD Small* **(c)** *UD Warehouse*

Figure 7. Simulated trajectories for Monte-Carlo simulations. (a): *EuRoc Vicon Room2 02* (115s, Burri et al. (2016)) with fully excited 3D motion; (b) & (c): *UD Small* (60s) & *UD Warehouse* (52s) with nonholonomic constrained 3D motion along with simulated walls, floor, and ceiling (floor and ceiling are not shown for better visibility). The red and yellow diamonds denote the beginning and end of these trajectories, respectively.

Clearly, the MINS estimators without the error model are shown to be inconsistent and overconfident for all the cases except 30 Hz as their NEESs are above 4. Note that ideal NEES of orientation and position should be around 3 if the estimator is consistent as their dimensions are 3. The estimators tend to show higher NEES and larger RMSE with lower cloning frequencies which indicates the unmodelled interpolation error has a larger impact on the lower setup. On the other hand, all the estimators with the error model were shown to be consistent with NEES below 4. The RMSE results are also lower than those without the error model showing that improved estimator consistency can benefit the localization performance.

6.2 Dynamic Cloning

Using the same setup, we now investigate how the proposed dynamic cloning balances localization performance and computational burden. We control the thresholds of dynamic cloning by multiplying different coefficients (0.01 - 100) to the default values (see Table 2). Fig. 8 shows the pose RMSE and total computation time results of dynamic cloning (red circles) which are averaged over 10 runs. The results of the estimator without dynamic cloning are also plotted with blue dots (fixed-rate cloning).

It is clear from these results that if we set the threshold very low (e.g. 0.01 in the figure) the performance of dynamic cloning is almost the same as the highest fixed-rate cloning frequency (30 Hz). This is because the dynamic cloning will enforce the MINS to maintain the interpolation error small which will end up setting the cloning frequency the highest most of the time. On the other hand, in the case we set the threshold very high (e.g. 100 in the figure), the performance of dynamic cloning is close to the lowest fixed-rate cloning frequency (4 Hz). The impact of dynamic cloning on how it balances accuracy and efficiency is more clear within the range of 0.1 - 10. In the case of 0.1 and 1, the accuracies of dynamic cloning were comparable to the 30 Hz fixed-rate cloning, while the computation times were only 75.9% and 61.9%, respectively. Also, in the case of 10, dynamic cloning was able to show computation time comparable to 4 Hz fixed-rate cloning, while the accuracy was better than 6 Hz fixed-rate cloning. Table 4 summarizes the pose RMSE and computation times of dynamic cloning with each threshold coefficient compared to 30 Hz fixed-rate cloning.

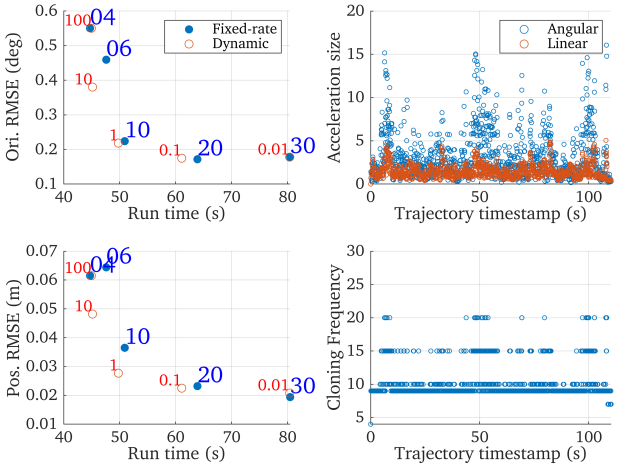


Figure 8. *Left*: Pose RMSE-total run time comparison between fixed-rate cloning and dynamic cloning. The fixed-rate clonings are plotted with blue dots along with frequencies, and dynamic clonings are shown with red circles with threshold coefficients. *Right*: Accelerations and cloning frequencies chosen by dynamic cloning with coefficient 1.

Table 4. Pose RMSE (deg/m) and total computation time (s) of fixed-rate and dynamic cloning with different coefficients.

	Fixed	**Dynamic cloning threshold coefficients**				
	30 Hz	**0.01**	**0.1**	**1**	**10**	**100**
Ori.	0.178	0.178	0.175	0.218	0.380	0.550
Pos.	0.019	0.021	0.023	0.028	0.048	0.062
Time	80.5 (100%)	80.4 (99.9%)	61.1 (75.9%)	49.8 (61.9%)	45.1 (56.1%)	44.9 (55.9%)

The right column of Fig. 8 shows an exemplary case of how the MINS changed its cloning frequency with given angular/linear accelerations and coefficient 1.

6.3 Fusing Different Sensors

Based on the interpolation strategy, we validate the proposed MINS with different combinations of available sensors. We test it on the *UD Small* trajectory with non-holonomic constraints (see Fig. 7b), as two differential wheeled vehicles cannot perform the holonomic motion. Note that the GNSS measurement noise level is set to 0.1 m (see Table 2), differential GPS quality to get comparable results with other sensor suits. Table 5 shows the orientation and position

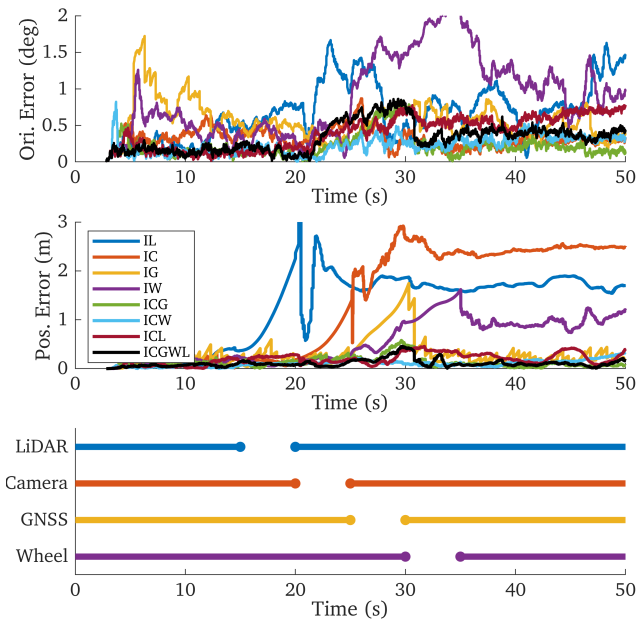


Figure 9. The performance of each algorithm under sensor failure scenarios. Pose estimation errors over time are displayed in the top two figures, and the times each sensor measurement was available are shown in the bottom figure (I: IMU, C: camera, G: GNSS, W: Wheel, L: LiDAR).

Table 5. Orientation/position RMSE and NEES, and run time results of MINS with different combinations of sensors. The best RMSE results are highlighted in bold text.

I	C	G	W	L	RMSE (deg / m)	NEES	Time (s)
●	●				0.505 ± 0.226 / 0.139 ± 0.054	3.2 ± 1.4 / 2.4 ± 1.4	23.6 ± 0.3
●		●			1.244 ± 0.250 / 0.191 ± 0.015	3.2 ± 1.3 / 3.6 ± 0.7	0.5 ± 0.0
●			●		3.053 ± 1.603 / 0.636 ± 0.159	2.4 ± 1.0 / 0.9 ± 0.3	1.0 ± 0.0
●				●	0.474 ± 0.106 / 0.098 ± 0.019	1.9 ± 0.6 / 1.1 ± 0.3	22.3 ± 0.6
●	●	●			0.318 ± 0.042 / 0.057 ± 0.008	3.8 ± 1.4 / 3.6 ± 1.2	23.6 ± 0.2
●	●		●		0.505 ± 0.231 / 0.102 ± 0.029	3.2 ± 1.4 / 2.2 ± 0.8	24.0 ± 0.3
●	●			●	0.414 ± 0.174 / 0.084 ± 0.038	2.6 ± 1.2 / 2.3 ± 1.4	49.6 ± 0.4
●	●	●	●	●	**0.261** ± 0.054 / **0.050** ± 0.005	3.0 ± 1.3 / 3.9 ± 1.1	49.9 ± 0.2

* I: IMU, C: Camera, G: GNSS, W: Wheel, L: LiDAR

RMSE and NEES, and the run time results of each sensor combination averaging 10 runs. All results are collected with an Intel i7 CPU single-threaded. Overall, the proposed MINS with all sensor combinations show consistent performance with NEES under 4. The IMU-camera and IMU-LiDAR pairs show comparable accuracies to each other and both ran 2.6 times faster than real time, while IMU-GNSS and IMU-wheel have lower accuracies with extremely small computation. The accuracies are shown to be improved with more sensors used in this case and the best accuracy was achieved when fusing all the sensors, confirming the benefits of multi-sensor fusion.

6.4 Robust to Sensor Failure

To show the robustness of the proposed MINS to sensor failures, we simulate the scenarios that measurements of each sensor are dropped for 5 seconds at different times. We use *UD Warehouse* (Fig. 7c) for this test, to be more realistic and ensure the LiDAR is not necessarily getting the map matching after the sensor failure (no loop-closure to the map built before sensor drop). The pose errors of different combinations of sensors and the period of time when the sensor measurements are available are shown in Fig. 9. Clearly, two sensor-paired systems are shown to have large error increases during sensor failure and maintained the error even after the sensors are recovered (except IG which has GNSS). On the other hand, those systems combining more than 2 sensors were able to bound the error growth by leveraging the auxiliary sensors, showing the robustness to the sensor failure and providing more accurate estimation.

6.5 Robust to Poor Calibration

To evaluate the online calibration of the proposed MINS, we test each calibration giving 4 different initial errors. Fig. 10 shows the calibration errors of spatiotemporal extrinsic and intrinsic of each sensor over time. It is clear that each error converges quickly to near zero after the system is initialized and remains bounded by 3σ envelopes showing the consistency of the online calibration. Note we showed 1-dimensional errors by summing the error of each axis of the calibration parameter, e.g. the position error is the summation of x, y, z errors, for concise visualization while the results of the individual axis also shows the same consistent results.

We further validate the necessity of online calibration for accurate and consistent localization by enabling/disabling the calibration at different levels of initial calibration errors. To be more specific, we investigated how robust the system is to the initial perturbations and whether the use of online sensor calibration enables improvements in accuracy and consistency. As shown in Fig. 11 and Fig. 12, for each of the different calibration parameters we perturb it with different levels of noise (note that we also change the initial covariance of the variable corresponds to the noise level used for perturbation). We can see that the proposed estimator is relatively invariant to the initial inaccuracies of the parameters and is able to output a near-constant trajectory error and consistency. An estimator, which does not perform this online estimation, has its trajectory estimation error and NEES values quickly increase to non-usable levels which validate the use of online calibration.

7 Experimental Results

We further evaluated the proposed MINS on both the public KAIST Urban Dataset (Jeong et al. 2019) and our own collected UD Husky Dataset, whose details are the following:

- **KAIST Urban Dataset**: collected on ground vehicles in challenging scenarios – highways (18-25, 35-37) and cities (26-34, 38-39) – and have a 100 Hz Xsens MTi-300 IMU, a 10 Hz Pointgrey Flea3 stereo camera, two 10 Hz Velodyne 16 channel LiDARs, a 10Hz u-blox EVK-7P GNSS, and a 100 Hz RLS LM13 wheel encoders.
- **UD Husky Dataset**: collected on a Clearpath Husky robot in structured environments (indoors and outdoors) and unstructured environments (trails). The dataset contains a T265 stereo camera (two 30 Hz cameras and a 200 Hz IMU), a 1 Hz Garmin GPS 18x, a 5 Hz Emlid reach m+ GPS (RTK), a 10 Hz Ouster

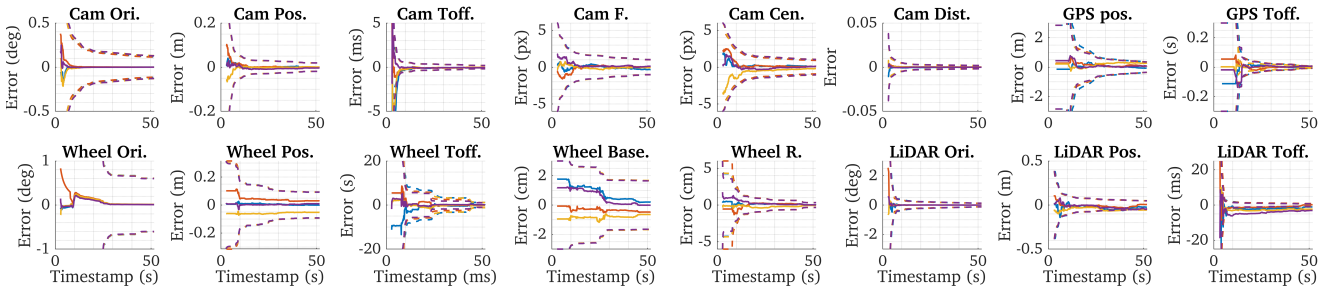


Figure 10. Calibration error of 4 runs with different initial guesses (different colors). The errors and their 3σ envelopes are shown with solid lines and dotted lines, respectively.

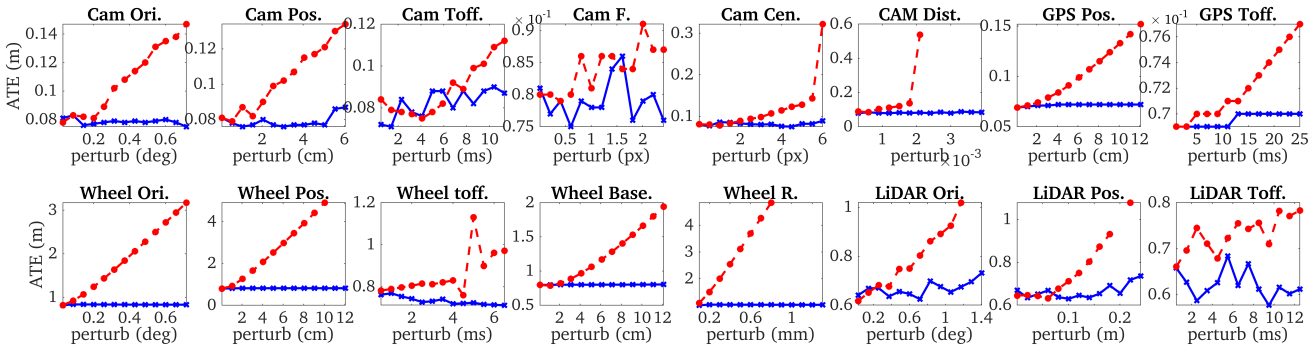


Figure 11. Position ATE under different levels of sensor calibration parameter perturbations. Results of with online calibration and without calibration are shown with red and blue lines.

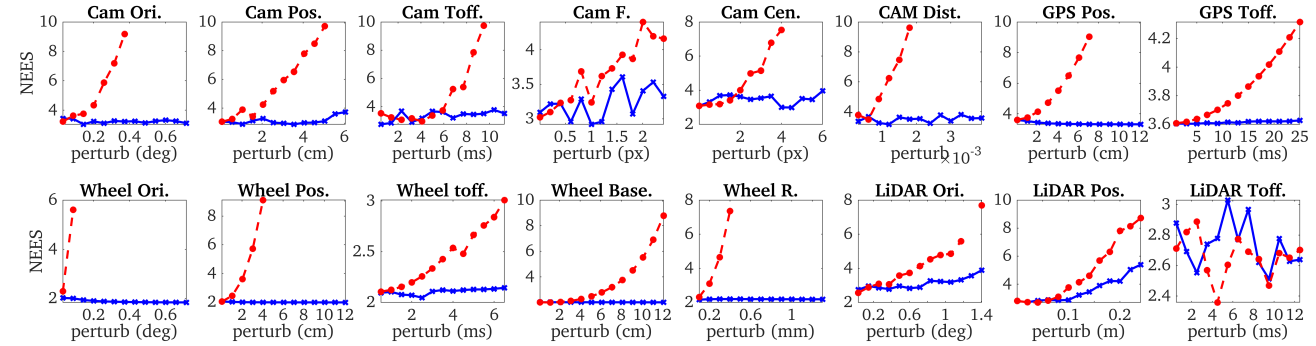


Figure 12. Position NEES under different levels of sensor calibration parameter perturbations. Results of with online calibration and without calibration are shown with red and blue lines.

64 channel LiDAR, and a 10 Hz wheel encoders. We considered the OptiTrack information (indoor) or RTK-GPS measurements (outdoor, trail) as ground truth for evaluation (see Fig. 15).

We compared our MINS to the following state-of-the-art (SOTA) methods as benchmarks:

- **VINS-Fusion** (Qin et al. 2019): Camera and IMU are fused based on the sliding window (VIO) and loop closure or GNSS information can be loosely coupled within pose graphs. The system supports dynamic initialization. We evaluated the VIO part VINS-Fusion(V), with loop closure VINS-Fusion(L), and with GPS VINS-Fusion(G). Note the algorithm assumes the GNSS and VIO are synchronized which is not the case in the dataset. We, therefore, allowed a maximum of 50 ms mismatch between VIO and GNSS.
- **ORB-SLAM3** (Campos et al. 2021): Camera and IMU are fused within a factor graph performing multimap-based local BA and loop closure. The system supports dynamic initialization.
- **FAST-LIO2** (Xu et al. 2022a): LiDAR and IMU are fused based on iterated Kalman filter. The system models LiDAR measurement as point-on-plane constraints building a global map using ikd-tree (Cai et al. 2021). The system supports static initialization.
- **LIW-OAM** (Yuan et al. 2023): LiDAR, IMU, and wheel are fused in local BA fashion using point-on-plane constraints as LiDAR measurement model. The system builds a global voxel map and supports dynamic initialization using LiDAR and IMU.
- **Lvio-Fusion** (Jia et al. 2021): Camera, LiDAR, and IMU are tightly coupled within the factor graph while GNSS is coupled loosely within the pose graph. The

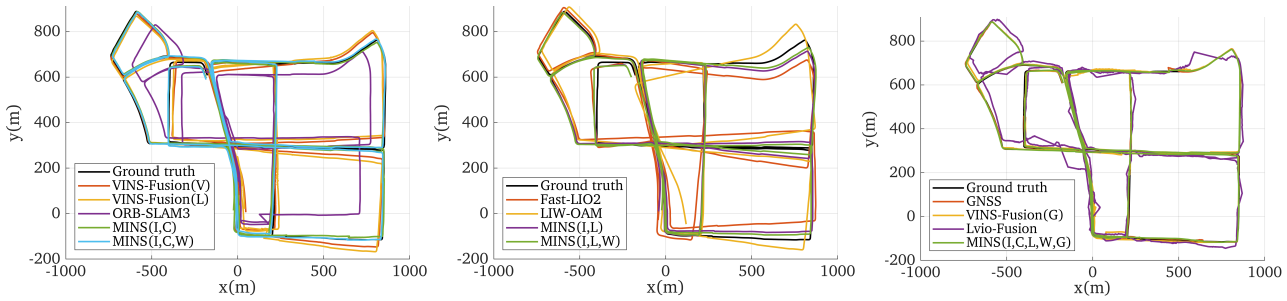


Figure 13. Trajectories of each algorithm on the KAIST Urban 38 dataset. *Left*: Camera-based algorithms; *Middle*: LiDAR-based algorithms; *Right*: GNSS-based algorithms.

Table 6. Average (5 runs) position ATE (m) of each algorithm per 1 km on the KAIST Urban dataset (sequence 18 - 39). Above 30 m errors are not reported. Shaded sequences are highway scenarios and the others are city scenarios. The best results among the same sensor group are highlighted with bold text.

	Algorithms	18 4km	19 3km	20 3km	21 4km	22 3km	23 3km	24 4km	25 3km	26 4km	27 5km	28 11km	29 4km	30 6km	31 11km	32 7km	33 8km	34 8km	35 3km	36 9km	37 12km	38 11km	39 11km
Camera	VINS-Fusion(V)	-	-	-	-	-	-	-	-	8.05	10.34	1.54	10.03	24.83	-	-	11.81	-	-	-	-	2.23	2.11
	VINS-Fusion(L)	-	-	-	-	-	-	-	-	7.82	11.97	2.54	10.03	25.21	-	-	11.81	-	-	-	-	2.86	1.68
	ORB-SLAM3	10.93	-	12.35	-	-	10.00	-	-	5.65	15.60	2.34	6.81	2.58	**2.03**	**3.14**	6.67	-	-	-	29.70	10.34	15.32
	MINS(I,C)	-	-	-	-	-	-	-	-	2.04	**2.86**	1.45	3.44	3.50	5.39	4.69	1.97	**1.47**	-	-	-	1.13	**1.17**
	MINS(I,C,W)	**7.13**	**8.74**	**8.25**	**8.48**	**9.16**	**7.10**	**8.17**	**10.74**	**1.57**	3.85	**1.09**	**3.16**	**2.03**	29.80	4.15	2.46	2.44	**8.15**	**16.48**	**18.50**	**0.94**	1.23
LiDAR	FAST-LIO2	-	-	-	-	-	-	-	-	9.75	7.59	-	**3.01**	**4.88**	-	11.14	9.90	-	-	-	-	4.74	**1.64**
	LIW-OAM	-	13.47	24.60	-	-	-	-	-	3.59	-	6.42	5.84	20.49	15.60	-	20.39	-	-	18.70	-	6.86	25.99
	MINS(I,L)	-	-	-	-	-	-	-	-	2.69	**2.97**	**2.53**	3.97	6.21	5.11	5.68	9.17	**5.30**	-	-	-	3.50	2.17
	MINS(I,L,W)	**6.96**	**8.11**	**4.21**	**2.99**	**6.91**	**4.25**	**6.85**	**3.76**	**2.24**	4.15	2.71	4.40	5.47	**5.01**	**4.72**	**7.92**	7.86	**6.13**	**12.01**	**7.96**	**3.24**	2.50
GNSS	GNSS	2.16	4.13	1.53	**0.64**	**0.53**	2.61	2.61	4.10	1.74	1.90	0.87	3.07	2.59	0.70	**0.69**	1.21	**0.47**	**2.46**	2.98	0.67	0.68	0.67
	VINS-Fusion(G)	-	-	-	-	-	-	-	-	4.02	-	1.60	4.77	9.35	8.02	-	5.80	-	-	-	-	1.07	2.26
	Lvio-Fusion	-	-	27.43	20.77	-	24.93	-	-	2.70	5.20	1.32	13.27	6.64	-	2.55	2.37	-	-	-	-	5.57	5.70
	MINS(I,C,L,W,G)	**1.52**	**3.46**	**0.85**	0.79	1.50	**1.38**	**1.86**	**3.44**	**1.06**	**1.72**	**0.79**	**2.58**	**1.83**	**0.70**	0.77	**1.16**	0.94	5.47	**2.38**	**0.60**	**0.64**	**0.56**

Figure 14. Representative images from the datasets. The Left shows the beginning of sequence 18 which is on the highway and the right shows the city driving scene of sequence 38 where the cars stopped at the traffic light which will become dynamic objects of the camera and LiDAR when the light turns green.

initialization is conducted using the camera and IMU supporting dynamic initialization.

Note that all the above LiDAR-based methods can fuse only one LiDAR sensor which showed poor performance on the KAIST dataset due to its limited overlapping points between the LiDAR scans. To make fair comparisons, we modified the KAIST dataset by transforming all the second LiDAR pointcloud to the first LiDAR's frame and created one synthetic 20 Hz LiDAR measurement while MINS used the original data.

7.1 Localization Accuracy

7.1.1 KAIST Urban Dataset Table 6 shows the average position ATE of 5 runs of each algorithm on each sequence (above 30 m error not reported) and Fig. 13 shows exemplary trajectories of all algorithms tested on the KAIST urban 38. Note that we show the ATE values per 1 km to match the scale of different trajectory lengths. The shaded sequences represent the datasets collected on highways while the vehicle was running at high speed and the other sequences are collected within the cities.

For the camera-based methods, the subsets of MINS using IMU and camera MINS(I,C) and IMU, camera, and wheel MINS(I,C,W) are evaluated along with SOTA. It is clear that all the methods that leverage only the camera and IMU performed poorly on highway datasets due to initialization failure (see Fig. 14 left). As the datasets start with high speed and mostly straight-line motion, none of those methods was successfully initialized except ORB-SLAM3 on a few sequences with sill large errors. On the other hand, MINS(I,C,W) leveraged wheel measurements to successfully initialize and perform accurate localization without failure. For the city sequences, all VINS methods initialized successfully, but their performances were largely affected by the dynamic objects (see an exemplary case in Fig. 14 right). MINS handled the issue by performing all camera update with the MSCKF technique (see Eq. (22)) which minimizes the effect of the dynamic objects due to shorter tracking period. To this end, the camera-based MINS modules are shown to outperform the other methods and are able to robustly handle hard scenarios.

For the LiDAR-based methods, the subsets of MINS using IMU and LiDAR MINS(I,L) and IMU, LiDAR, and wheel MINS(I,L,W) are evaluated along with SOTA. Similar to the camera-based methods, the initialization was the biggest issue on the highway sequences. FAST-LIO2, MINS(I,L), and LIW-OAM mostly failed in initialization and quickly diverged while MINS(I,L,W) was able to initialize with wheel measurements and perform estimation consistently. On city sequences, relatively large orientation drifts of both FAST-LIO2 and LIW-OAM (see Fig. 13) are observed when

Figure 15. Husky robot and exemplary scenes of UD Husky Dataset. *Top-Left*: Husky robot with multiple sensors mounted. *Top-Right*: Indoor dataset. *Bottom-Left*: Outdoor dataset. *Bottom-Right*: Trail dataset.

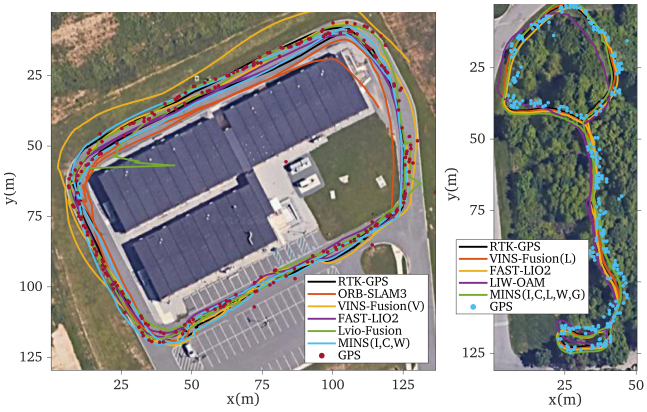


Figure 16. Trajectories of each algorithm on UD Husky Dataset. *left*: Outdoor 1; *right*: Trail 3.

Table 7. Average (5 runs) position ATE (m) of each algorithm on the UD Husky dataset. Above 5 m errors are not reported (except GNSS for reference).

	Algorithms	Structured Env.				Unstructured Env.			
		I1	I2	O1	O2	T1	T2	T3	T4
Camera	VINS-Fusion(V)	0.63	0.71	-	4.39	4.82	4.92	1.60	-
	VINS-Fusion(L)	0.55	0.49	-	4.07	3.36	4.31	**0.92**	3.11
	ORB-SLAM3	0.50	**0.48**	4.51	-	2.50	-	1.69	**2.74**
	MINS(I,C)	0.33	0.73	2.93	3.92	3.16	3.19	2.64	4.66
	MINS(I,C,W)	**0.22**	0.67	**1.39**	**2.57**	**2.40**	**1.95**	2.16	3.83
LiDAR	FAST-LIO2	0.18	0.16	2.34	2.74	**0.95**	2.82	1.37	2.02
	LIW-OAM	0.12	0.16	**1.01**	2.92	2.36	3.61	2.37	3.04
	MINS(I,L)	**0.07**	**0.11**	1.73	2.05	1.64	1.28	1.93	2.13
	MINS(I,L,W)	0.08	0.16	1.35	**1.99**	1.35	**1.05**	**1.25**	**1.62**
GNSS	GNSS			3.17	2.17	15.71	10.34	5.93	7.02
	VINS-Fusion(G)			-	3.25	-	-	-	-
	Lvio-Fusion			3.14	2.63	-	-	-	4.02
	MINS(I,C,L,W,G)			**0.96**	**1.07**	**3.03**	**2.89**	**1.33**	**1.30**

* I: Indoor dataset, O: Outdoor dataset, T: Trail dataset

the vehicle is making turns. The map density (0.3, 0.3, and 1.5 unit voxel size for FAST-LIO2, MINS, and LIW-OAM, respectively) and the lack of overlapping point clouds between the map and the new scan which creates an ill-constrained ICP problem are presumed to be the key reasons, while MINS was more robust to the scenarios even though all methods adopt the same point-on-plane measurement model.

For the GNSS-based methods, MINS using all the sensors MINS(I,C,L,W,G) is evaluated along with SOTA. Note that we recorded the estimated poses of SOTA in real-time, not the final optimized trajectory at the end of the run. We also showed the raw GNSS measurements for reference. Overall, VINS-Fusion(G) and Lvio-Fusion were not able to show consistent estimation on both highway and city datasets even with GNSS global information due to their loosely coupling manner. This is because their sub-systems, VIO of VINS-Fusion and LIO of Lvio-Fusion suffered from initialization and dynamic objects that they kept injecting inconsistent odometry information to the pose graph making global estimation highly inconsistent and resulting in even worse estimation performance than just GNSS record. On the other hand, MINS tightly coupled GNSS information with other sensors so that the state estimation was able to properly constrain its drift showing globally accurate and locally precise estimation performance.

7.1.2 UD Husky Dataset Table 7 shows the average position ATE of 5 runs of the same set of algorithms on each UD Husky Dataset (above 5 m error not reported). Note that all the sequences of the dataset start with the robot standing still enabling all the methods to successfully initialize.

For the camera-based methods, overall the algorithms that leverage only the camera and IMU reported poor localization performance showing large trajectory scale error (see Fig. 16). This is because the ground robot mainly undergoes the degenerate motion (Wu et al. 2017) which makes the scale of VINS unobservable. On the other hand, MINS(I,C,W) was able to constrain the scale from wheel information and showed higher accuracy.

The LiDAR-based methods were able to show smaller ATE under the same motion profiles as they directly gained the scale information from the 3D pointcloud, and MINS was able to outperform the others in most of the sequences. An observation is that the wheel information did not improve the performance in the indoor dataset. The dataset contains many sharp-turning motions of the robot which our wheel model (see Eq. (27)) may not be able to accurately represent the actual robot motion, thus wheel information may harm the estimation performance especially when it is centimeter-level accuracy.

Compared to the GPS of the KAIST dataset, our GPS measurements were more intermittent and noisy as shown in the table which imposes a larger challenge in fusing the information. The loosely-coupled methods, VINS-Fusion(G) and Lvio-Fusion, failed most of the sequences, due to their VIO drift or large GPS noise. On the other hand, MINS was able to show consistent estimation by tightly fusing all sensor information. However, it is also shown that fusing all the sensors did not necessarily return the best localization performance. For example, in Trail 1 - 3 sequences, MINS(I,C,L,W,G) performed worse than both MINS(I,L,W) and MINS(I,C,W) which indicates a proper choice of sensors can lead to a better performance in certain scenarios.

7.2 Computation Efficiency

Table 8 shows the timing breakdown (frontend, backend, and total) of each algorithm collected on ThinkPad P17 which has an Intel i7 CPU and 32 GB RAM. The timings are reported as the average time taken per the function call on the KAIST Urban 38 dataset.

The frontend includes all the processes required before optimization or EKF update, such as IMU preintegration, image processing, or LiDAR pointcloud matching. The camera is reported to be the one that took the most time to process. Two subsets of MINS, MINS(I,C) and MINS(I,C,L,W,G), reported different camera timing because we lowered the number of features extracted from images to balance the computation with LiDAR. Other frontend timings including IMU, GNSS, wheel, and LiDAR are shown to be very low for all the algorithms.

The backend includes the map management and the optimization (or EKF update) which shows large differences among the algorithms. Those algorithms using cameras mostly use the map to find loop closures while those using LiDARs build the map to perform scan matching. VINS-Fusion and ORB-SLAM3 perform the loop closure detection taking 70 ms and 10 ms on average, while MINS does not maintain a map of the camera for computational efficiency. Both FAST-LIO2 and subsets of MINS utilized ikd-tree (Cai et al. 2021) to efficiently manage the LiDAR map points, but MINS took more time because it keeps more pointclouds and transforms all the map points to a new anchor frame occasionally (see Sec. 3.4.1). Other LiDAR methods recorded similar or larger map-management time.

The optimization (or EKF update) time also shows large variations among the algorithms. MINS and FAST-LIO2, which are the filters, showed the shortest time due to their small state size and the number of iterations. Especially, MINS recorded 1 ms for the EKF update regardless of the number of sensors being used, due to the measurement compression and the dynamic cloning which reduced the state size and the computation. Other graph-based methods report one or two orders of magnitude larger optimization time due to their large state size (number of nodes). Note we reported the summation of the optimization time if an algorithm runs multiple threads that do optimization, e.g. VINS-Fusion(G) optimization time is computed by summing VINS-Fusion(V) optimization time and global graph optimization time.

Lastly, Table 8 also shows the total time which can represent the amount of time expected if all processes run in serial. Clearly, MINS and FAST-LIO2 reported the smallest computation time due to the small optimization time. While FAST-LIO2 was the fastest, MINS recorded the second fastest results and showed almost invariant computation time to the number of sensors being used. Other graph-based methods show larger time records mostly determined by the optimization time. Note that a large total time does not mean the algorithm cannot run in real-time (actually all the algorithms ran in real-time on the laptop) as the processes can be multi-threaded. However, their performance can be largely affected on embedded systems as their computation power is relatively limited.

Table 8. Timing records on the KAIST Urban 38 (*ms*).

	Algorithms	Frontend					Backend		Total
		I	C	G	W	L	Map	Opt.	
Camera	VINS-Fusion(V)	0	27	-	-	-	-	68	95
	VINS-Fusion(L)	0	27	-	-	-	70	125	222
	ORB-SLAM3	0	28	-	-	-	10	290	328
	MINS(I,C)	0	53	-	-	-	-	1	54
	MINS(I,C,W)	0	55	-	0	-	-	1	56
LiDAR	FAST-LIO2	0	-	-	-	1	5	14	20
	LIW-OAM	0	-	-	0	7	40	48	95
	MINS(I,L)	0	-	-	-	6	22	1	28
	MINS(I,L,W)	0	-	-	0	7	23	1	31
GNSS	VINS-Fusion(G)	0	27	-	-	-	-	457	484
	Lvio-Fusion	0	18	0	-	2	20	106	146
	MINS(I,C,L,W,G)	0	34	0	0	2	10	1	47

8 Conclusions and Future Work

In this paper, we have developed a robust Multi-sensor-aided Inertial Navigation System (MINS) that integrates an IMU, a pair of wheel encoders, and arbitrary numbers of cameras, LiDARs, and GNSS for robust and accurate state estimation. The proposed MINS addresses some of the key challenges of multisensor fusion by introducing consistent high-order on manifold state interpolation, dynamic cloning for managing state size and computation, online calibration of sensor parameters, and IMU-wheel combined initialization. The proposed approach has been validated through extensive evaluations conducted in realistic simulations and challenging real-world datasets, showing superior accuracy and consistency while maintaining lower computational complexity compared to the SOTA methods. Our future work will explore the optimal weighting of multi-modal measurements, which plays a crucial role in multisensor fusion and even can harm the estimator performance if not set properly as shown in our experimental results.

Acknowledgements

This work was partially supported by the University of Delaware (UD) College of Engineering, the NSF (IIS-1924897), and the ARL (W911NF-19-2-0226).

References

Anderson S, Barfoot TD, Tong CH and Särkkä S (2015) Batch nonlinear continuous-time trajectory estimation as exactly sparse gaussian process regression. *Autonomous Robots* 39(3): 221–238.

Anderson S, Dellaert F and Barfoot TD (2014) A hierarchical wavelet decomposition for continuous-time slam. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 373–380.

Barfoot TD (2017) *State estimation for robotics*. Cambridge University Press.

Brommer C, Jung R, Steinbrener J and Weiss S (2020) Mars: A modular and robust sensor-fusion framework. *IEEE Robotics and Automation Letters* 6(2): 359–366.

Burri M, Nikolic J, Gohl P, Schneider T, Rehder J, Omari S, Achtelik MW and Siegwart R (2016) The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research* DOI:10.1177/0278364915620033.

URL http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract.

Cai Y, Xu W and Zhang F (2021) ikd-tree: An incremental kd tree for robotic applications. *arXiv preprint arXiv:2102.10808* .

Campos C, Elvira R, Rodríguez JJG, Montiel JM and Tardós JD (2021) Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics* 37(6): 1874–1890.

Camurri M, Ramezani M, Nobili S and Fallon M (2020) Pronto: A multi-sensor state estimator for legged robots in real-world scenarios. *Frontiers in Robotics and AI* 7: 68.

Cao S, Lu X and Shen S (2022) Gvins: Tightly coupled gnss–visual–inertial fusion for smooth and consistent state estimation. *IEEE Transactions on Robotics* .

Chambers A, Scherer S, Yoder L, Jain S, Nuske S and Singh S (2014) Robust multi-sensor fusion for micro aerial vehicle navigation in gps-degraded/denied environments. In: *2014 American Control Conference*. IEEE, pp. 1892–1899.

Chen C, Yang Y, Geneva P and Huang G (2022a) FEJ2: A consistent visual-inertial state estimator design. In: *International Conference on Robotics and Automation (ICRA)*. Philadelphia, USA.

Chen W, Zhou C, Shang G, Wang X, Li Z, Xu C and Hu K (2022b) Slam overview: From single sensor to heterogeneous fusion. *Remote Sensing* 14(23): 6033.

Chirikjian G (2011) *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*, volume 2. Springer Science & Business Media.

Chiu HP, Zhou XS, Carlone L, Dellaert F, Samarasekera S and Kumar R (2014) Constrained optimal selection for multi-sensor robot navigation using plug-and-play factor graphs. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 663–670.

Cioffi G, Cieslewski T and Scaramuzza D (2022) Continuous-time vs. discrete-time vision-based slam: A comparative study. *IEEE Robotics and Automation Letters* 7(2): 2399–2406.

Cioffi G and Scaramuzza D (2020) Tightly-coupled fusion of global positional measurements in optimization-based visual-inertial odometry. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5089–5095.

Dang Z, Wang T and Pang F (2018) Tightly-coupled data fusion of vins and odometer based on wheel slip estimation. In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 1613–1619.

Dong B and Zhang K (2022) A tightly coupled visual-inertial gnss state estimator based on point-line feature. *Sensors* 22(9): 3391.

Dong-Si TC and Mourikis AI (2011) Closed-form solutions for vision-aided inertial navigation. Technical report, Tech. rep. Dept. of Electrical Engineering, University of California ....

Droeschel D and Behnke S (2018) Efficient continuous-time slam for 3d lidar-based online mapping. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5000–5007.

DuToit RC, Hesch JA, Nerurkar ED and Roumeliotis SI (2017) Consistent map-based 3d localization on mobile devices. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6253–6260.

Eckenhoff K, Geneva P and Huang G (2021) MIMC-VINS: A versatile and resilient multi-imu multi-camera visual-inertial navigation system. *IEEE Transactions on Robotics* 37(5): 1360–1380. DOI:10.1109/TRO.2021.3049445.

Eidson JC, Fischer M and White J (2002) Ieee-1588™ standard for a precision clock synchronization protocol for networked measurement and control systems. In: *Proceedings of the 34th Annual Precise Time and Time Interval Systems and Applications Meeting*. pp. 243–254.

Feng L (2021) Initialization improvement and map reuse based on orbslam3. In: *2021 2nd International Conference on Artificial Intelligence and Information Systems*. pp. 1–7.

Furgale P, Barfoot TD and Sibley G (2012) Continuous-time batch estimation using temporal basis functions. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE, pp. 2088–2095.

Furgale P, Rehder J and Siegwart R (2013) Unified temporal and spatial calibration for multi-sensor systems. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 1280–1286.

Gang P, Zezao L, Bocheng C, Shanliang C and Dingxin H (2020) Robust tightly coupled pose estimation based on monocular vision, inertia, and wheel speed. *arXiv preprint arXiv:2003.01496* .

Geneva P, Eckenhoff K and Huang G (2018a) Asynchronous multi-sensor fusion for 3d mapping and localization. In: *Proc. of the IEEE International Conference on Robotics and Automation*. Brisbane, Australia, pp. 5994–5999.

Geneva P, Eckenhoff K and Huang G (2019) A linear-complexity EKF for visual-inertial navigation with loop closures. In: *Proc. International Conference on Robotics and Automation*. Montreal, Canada, pp. 3535–3541.

Geneva P, Eckenhoff K, Lee W, Yang Y and Huang G (2020a) OpenVINS: a research platform for visual-inertial estimation. In: *Proc. of the IEEE International Conference on Robotics and Automation*. Paris, France, pp. 4666–4672. URL https://github.com/rpng/open_vins.

Geneva P, Eckenhoff K, Yang Y and Huang G (2018b) LIPS: Lidar-inertial 3d plane slam. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Madrid, Spain.

Geneva P, Merrill N, Yang Y, Chen C, Lee W and Huang G (2020b) Versatile 3d multi-sensor fusion for lightweight 2d localization. In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Las Vegas, NV.

Girrbach F, Zandbergen R, Kok M, Hageman T, Bellusci G and Diehl M (2019) Towards in-field and online calibration of inertial navigation systems using moving horizon estimation. In: *2019 18th European Control Conference (ECC)*. IEEE, pp. 4338–4343.

Golub GH and Van Loan CF (2013) *Matrix computations*. JHU press.

Hackett JK and Shah M (1990) Multi-sensor fusion: a perspective. In: *Proceedings., IEEE International Conference on Robotics and Automation*. IEEE, pp. 1324–1330.

Han S, Deng F, Li T and Pei H (2022) Tightly coupled optimization-based gps-visual-inertial odometry with online calibration and initialization. *arXiv preprint arXiv:2203.02677* .

Hausman K, Weiss S, Brockers R, Matthies L and Sukhatme GS (2016) Self-calibrating multi-sensor fusion with probabilistic measurement validation for seamless sensor switching on a

uav. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4289–4296.

Heng L, Li B and Pollefeys M (2013) Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. pp. 1793–1800. DOI:10.1109/IROS.2013.6696592.

Hertzberg C, Wagner R, Frese U and Schröder L (2013) Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds. *Information Fusion* 14(1): 57–77.

Hesch JA, Kottas DG, Bowman SL and Roumeliotis SI (2012) Observability-constrained vision-aided inertial navigation. *University of Minnesota, Dept. of Comp. Sci. & Eng., MARS Lab, Tech. Rep* 1: 6.

Hou JJ, Wu X, Shan J, Li D and Wang H (2022) Robust optimization-based fusion of gnss and visual-inertial-wheel odometry. In: *2022 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, pp. 1–6.

Huang G (2019) Visual-inertial navigation: A concise review. In: *Proc. International Conference on Robotics and Automation*. Montreal, Canada, pp. 9572–9582.

Huang G, Mourikis AI and Roumeliotis SI (2008) A first-estimates Jacobian EKF for improving SLAM consistency. In: *Proc. of the 11th International Symposium on Experimental Robotics*. Athens, Greece, pp. 373–382.

Jeong J, Cho Y, Shin YS, Roh H and Kim A (2019) Complex urban dataset with multi-level sensors from highly diverse urban environments. *The International Journal of Robotics Research* 38(6): 642–657.

Jia Y, Luo H, Zhao F, Jiang G, Li Y, Yan J, Jiang Z and Wang Z (2021) Lvio-fusion: A self-adaptive multi-sensor fusion slam framework using actor-critic method. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 286–293.

Jiang F, Chen J and Ji S (2021) Panoramic visual-inertial slam tightly coupled with a wheel encoder. *ISPRS Journal of Photogrammetry and Remote Sensing* 182: 96–111.

Jung JH, Cha J, Chung JY, Kim TI, Seo MH, Park SY, Yeo JY and Park CG (2020) Monocular visual-inertial-wheel odometry using low-grade imu in urban areas. *IEEE Transactions on Intelligent Transportation Systems* .

Kang R, Xiong L, Xu M, Zhao J and Zhang P (2019) Vins-vehicle: A tightly-coupled vehicle dynamics extension to visual-inertial state estimator. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, pp. 3593–3600.

Kannala J and Brandt SS (2006) A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE transactions on pattern analysis and machine intelligence* 28(8): 1335–1340.

Kelly J and Sukhatme GS (2011) Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration. *The International Journal of Robotics Research* 30(1): 56–79.

Khattak S, Nguyen H, Mascarich F, Dang T and Alexis K (2020) Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments. In: *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, pp. 1024–1029.

Kubelka V, Oswald L, Pomerleau F, Colas F, Svoboda T and Reinstein M (2015) Robust data fusion of multimodal sensory information for mobile robots. *Journal of Field Robotics* 32(4): 447–473.

Lang X, Lv J, Huang J, Ma Y, Liu Y and Zuo X (2022) Ctrl-vio: Continuous-time visual-inertial odometry for rolling shutter cameras. *IEEE Robotics and Automation Letters* 7(4): 11537–11544.

Lee W, Eckenhoff K, Geneva P and Huang G (2019) Gps-aided visual-inertial navigation in large-scale environments. Technical report, Robot Perception and Navigation Group (RPNG), University of Delaware. URL https://copland.udel.edu/~ghuang/papers/tr_gps-vio.pdf.

Lee W, Eckenhoff K, Geneva P and Huang G (2020a) Intermittent gps-aided vio: Online initialization and calibration. In: *Proc. of the IEEE International Conference on Robotics and Automation*. Paris, France, pp. 5724–5731.

Lee W, Eckenhoff K, Yang Y, Geneva P and Huang G (2020b) Visual-inertial-wheel odometry with online calibration. In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Las Vegas, NV, pp. 4559–4566.

Lee W, Geneva P, Yang Y and Huang G (2022) Tightly-coupled gnss-aided visual-inertial localization. In: *International Conference on Robotics and Automation (ICRA)*. Philadelphia, USA, pp. 9484–9491.

Lee W and Huang G (2020) Visual inertial wheel odometry: Online calibration technical report. Technical report, Robot Perception and Navigation Group (RPNG), University of Delaware. URL https://copland.udel.edu/~ghuang/papers/tr_wheel-vio.pdf.

Lee W, Yang Y and Huang G (2021) Efficient multi-sensor aided inertial navigation with online calibration. In: *Proc. of the IEEE International Conference on Robotics and Automation*. Xi'an, China, pp. 5706–5712.

Lee Y, Yoon J, Yang H, Kim C and Lee D (2016) Camera-gps-imu sensor fusion for autonomous flying. In: *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, pp. 85–88.

Li M (2014) *Visual-inertial odometry on resource-constrained systems*. University of California, Riverside.

Li M and Mourikis AI (2014a) Online temporal calibration for camera–imu systems: Theory and algorithms. *The International Journal of Robotics Research* 33(7): 947–964.

Li M and Mourikis AI (2014b) Vision-aided inertial navigation with rolling-shutter cameras. *The International Journal of Robotics Research* 33(11): 1490–1507.

Lin J and Zhang F (2022) R^3live: A robust, real-time, rgb-colored, lidar-inertial-visual tightly-coupled state estimation and mapping package. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 10672–10678.

Lin J, Zheng C, Xu W and Zhang F (2021) R^2live: A robust, real-time, lidar-inertial-visual tightly-coupled state estimator and mapping. *IEEE Robotics and Automation Letters* 6(4): 7469–7476.

Liu J, Gao W and Hu Z (2019) Visual-inertial odometry tightly coupled with wheel encoder adopting robust initialization and online extrinsic calibration. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5391–5397.

Liu J, Gao W and Hu Z (2021a) Bidirectional trajectory computation for odometer-aided visual-inertial slam. *IEEE Robotics and Automation Letters* 6(2): 1670–1677.

Liu J, Gao W and Hu Z (2021b) Optimization-based visual-inertial slam tightly coupled with raw gnss measurements. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 11612–11618.

Lv J, Lang X, Xu J, Wang M, Liu Y and Zuo X (2023) Continuous-time fixed-lag smoothing for lidar-inertial-camera slam. *IEEE/ASME Transactions on Mechatronics* .

Lv J, Zuo X, Hu K, Xu J, Huang G and Liu Y (2022) Observability-aware intrinsic and extrinsic calibration of lidar-imu systems. *IEEE Transactions on Robotics* .

Lynen S, Achtelik MW, Weiss S, Chli M and Siegwart R (2013) A robust and modular multi-sensor fusion approach applied to mav navigation. In: *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, pp. 3923–3929.

Ma F, Shi J, Wu L, Dai K and Zhong S (2020) Consistent monocular ackermann visual–inertial odometry for intelligent and connected vehicle localization. *Sensors* 20(20): 5757.

Ma F, Shi J, Yang Y, Li J and Dai K (2019) Ack-msckf: Tightly-coupled ackermann multi-state constraint kalman filter for autonomous vehicle localization. *Sensors* 19(21): 4816.

Mascaro R, Teixeira L, Hinzmann T, Siegwart R and Chli M (2018) Gomsf: Graph-optimization based multi-sensor fusion for robust uav pose estimation. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1421–1428.

Meng X, Wang H and Liu B (2017) A robust vehicle localization approach based on gnss/imu/dmi/lidar sensor fusion for autonomous vehicles. *Sensors* 17(9): 2140.

Merfels C and Stachniss C (2016) Pose fusion with chain pose graphs for automated driving. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 3116–3123.

Merfels C and Stachniss C (2017) Sensor fusion for self-localisation of automated vehicles. *PFG–Journal of Photogrammetry, Remote Sensing and Geoinformation Science* 85(2): 113–126.

Mourikis AI and Roumeliotis SI (2007) A multi-state constraint Kalman filter for vision-aided inertial navigation. In: *Proceedings of the IEEE International Conference on Robotics and Automation*. Rome, Italy, pp. 3565–3572.

Mueggler E, Gallego G, Rebecq H and Scaramuzza D (2018) Continuous-time visual-inertial odometry for event cameras. *IEEE Transactions on Robotics* 34(6): 1425–1440.

Nguyen TM, Cao M, Yuan S, Lyu Y, Nguyen TH and Xie L (2021a) Viral-fusion: A visual-inertial-ranging-lidar sensor fusion approach. *IEEE Transactions on Robotics* 38(2): 958–977.

Nguyen TM, Yuan S, Cao M, Nguyen TH and Xie L (2021b) Viral slam: Tightly coupled camera-imu-uwb-lidar slam. *arXiv preprint arXiv:2105.03296* .

Oliveira M, Pedrosa E, de Aguiar AP, Rato DFPD, dos Santos FN, Dias P and Santos V (2022) Atom: A general calibration framework for multi-modal, multi-sensor systems. *Expert Systems with Applications* 207: 118000.

Owens JL, Osteen PR and Daniilidis K (2015) Msg-cal: Multi-sensor graph-based calibration. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 3660–3667.

Paul MK and Roumeliotis SI (2018) Alternating-stereo vins: Observability analysis and performance evaluation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 4729–4737.

Qin T, Cao S, Pan J and Shen S (2019) A general optimization-based framework for global pose estimation with multiple sensors. *arXiv preprint arXiv:1901.03642* .

Qin T, Li P and Shen S (2018) Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics* 34(4): 1004–1020.

Qin T and Shen S (2017) Robust initialization of monocular visual-inertial estimation on aerial robots. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 4225–4232.

Quan M, Piao S, Tan M and Huang SS (2019) Tightly-coupled monocular visual-odometric slam using wheels and a mems gyroscope. *IEEE Access* 7: 97374–97389.

Quenzel J and Behnke S (2021) Real-time multi-adaptive-resolution-surfel 6d lidar odometry using continuous-time trajectory optimization. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5499–5506.

Rahman S, Quattrini Li A and Rekleitis I (2022) Svin2: A multi-sensor fusion-based underwater slam system. *The International Journal of Robotics Research* : 02783649221110259.

Ramanandan A, Chari M and Joshi A (2019) Systems and methods for using a global positioning system velocity in visual-inertial odometry. US Patent 10,371,530.

Rehder J, Gupta K, Nuske S and Singh S (2012) Global pose estimation with limited gps and long range visual odometry. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE, pp. 627–633.

Rehder J, Nikolic J, Schneider T, Hinzmann T and Siegwart R (2016a) Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4304–4311.

Rehder J, Siegwart R and Furgale P (2016b) A general approach to spatiotemporal calibration in multisensor systems. *IEEE Transactions on Robotics* 32(2): 383–398.

Rodrigues RT, Tsiogkas N, Pascoal A and Aguiar AP (2021) Online range-based slam using b-spline surfaces. *IEEE Robotics and Automation Letters* 6(2): 1958–1965.

Serov A, Clemens J and Schill K (2021) Visual-multi-sensor odometry with application in autonomous driving. In: *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*. IEEE, pp. 1–7.

Shan T, Englot B, Ratti C and Rus D (2021) Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In: *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 5692–5698.

Shao W, Vijayarangan S, Li C and Kantor G (2019) Stereo visual inertial lidar simultaneous localization and mapping. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 370–377.

Shen S, Mulgaonkar Y, Michael N and Kumar V (2014) Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft mav. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4974–4981.

Siegwart R, Nourbakhsh IR and Scaramuzza D (2011) *Introduction to autonomous mobile robots*. MIT press.

Simanek J, Kubelka V and Reinstein M (2015) Improving multi-modal data fusion by anomaly detection. *Autonomous Robots* 39(2): 139–154.

Su Y, Wang T, Shao S, Yao C and Wang Z (2021) Gr-loam: Lidar-based sensor fusion slam for ground robots on complex terrain. *Robotics and Autonomous Systems* 140: 103759.

Suhr JK, Jang J, Min D and Jung HG (2016) Sensor fusion-based low-cost vehicle localization system for complex urban environments. *IEEE Transactions on Intelligent Transportation Systems* 18(5): 1078–1086.

Sun H, Jin Y, Fu M, He J, Liu H and Zhang WA (2022) A multisensor-based tightly coupled integrated navigation system. In: *2022 5th International Symposium on Autonomous Systems (ISAS)*. IEEE, pp. 1–6.

Tessier C, Cariou C, Debain C, Chausse F, Chapuis R and Rousset C (2006) A real-time, multi-sensor architecture for fusion of delayed observations: application to vehicle localization. In: *2006 IEEE Intelligent Transportation Systems Conference*. IEEE, pp. 1316–1321.

Trawny N and Roumeliotis SI (2005) Indirect Kalman filter for 3D attitude estimation. Technical report, University of Minnesota, Dept. of Comp. Sci. & Eng.

Wang T, Su Y, Shao S, Yao C and Wang Z (2021a) Gr-fusion: Multi-sensor fusion slam for ground robots with high robustness and low drift. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5440–5447.

Wang Z, Li M, Zhou D and Zheng Z (2021b) Direct sparse stereo visual-inertial global odometry. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 14403–14409.

Wang Z, Zhang J, Chen S, Yuan C, Zhang J and Zhang J (2019) Robust high accuracy visual-inertial-laser slam system. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 6636–6641.

Wisth D, Camurri M and Fallon M (2022) Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots. *IEEE Transactions on Robotics* .

Wu D, Zhong X, Peng X, Hu H and Liu Q (2022) Multi-modal information fusion for high-robustness and low-drift state estimation of ugvs in diverse scenes. *IEEE Transactions on Instrumentation and Measurement* .

Wu KJ, Guo CX, Georgiou G and Roumeliotis SI (2017) Vins on wheels. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5155–5162.

Xu W, Cai Y, He D, Lin J and Zhang F (2022a) Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics* .

Xu X, Zhang L, Yang J, Cao C, Wang W, Ran Y, Tan Z and Luo M (2022b) A review of multi-sensor fusion slam systems based on 3d lidar. *Remote Sensing* 14(12): 2835.

Yang Y, Geneva P, Eckenhoff K and Huang G (2019a) Degenerate motion analysis for aided INS with online spatial and temporal calibration. *IEEE Robotics and Automation Letters (RA-L)* 4(2): 2070–2077.

Yang Y, Geneva P, Eckenhoff K and Huang G (2019b) Visual-inertial navigation with point and line features. In: *Proc. International Conference on Intelligent Robots and Systems (IROS)*. Macau, China, p. 3.

Yang Y, Geneva P and Huang G (2023) Multi-visual-inertial system: Analysis, calibration and estimation. *arXiv preprint arXiv:2308.05303* .

Yang Y, Lee W, Osteen P, Geneva P, Zuo X and Huang G (2021) iCalib: Inertial aided multi-sensor calibration. In: *Workshop on Visual-Inertial Navigation Systems*.

Yang Y, Maley J and Huang G (2017) Null-space-based marginalization: Analysis and algorithm. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vancouver, Canada, pp. 6749–6755.

Ye C, Pan H and Gao H (2021) Keypoint-based lidar-camera online calibration with robust geometric network. *IEEE Transactions on Instrumentation and Measurement* 71: 1–11.

Yuan Z, Lang F, Xu T and Yang X (2023) Liw-oam: Lidar-inertial-wheel odometry and mapping. *arXiv preprint arXiv:2302.14298* .

Yuan Z, Lang F and Yang X (2022) Sr-lio: Lidar-inertial odometry with sweep reconstruction. *arXiv preprint arXiv:2210.10424* .

Zhang J and Singh S (2018) Laser–visual–inertial odometry and mapping with high robustness and low drift. *Journal of field robotics* 35(8): 1242–1264.

Zhang M, Zuo X, Chen Y, Liu Y and Li M (2021) Pose estimation for ground robots: On manifold representation, integration, reparameterization, and optimization. *IEEE Transactions on Robotics* 37(4): 1081–1099.

Zhao S, Zhang H, Wang P, Nogueira L and Scherer S (2021) Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 8729–8736.

Zheng C, Zhu Q, Xu W, Liu X, Guo Q and Zhang F (2022) Fast-livo: Fast and tightly-coupled sparse-direct lidar-inertial-visual odometry. *arXiv preprint arXiv:2203.00893* .

Zheng F and Liu YH (2019) Visual-odometric localization and mapping for ground vehicles using se (2)-xyz constraints. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3556–3562.

Zheng X, Li M and Mourikis AI (2015) Decoupled representation of the error and trajectory estimates for ef_cient pose estimation. In: *Robotics: Science and Systems*.

Zuo X, Geneva P, Lee W, Liu Y and Huang G (2019a) LIC-Fusion: Lidar-inertial-camera odometry. In: *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*. Macau, China, pp. 5848–5854.

Zuo X, Yang Y, Geneva P, Lv J, Liu Y, Huang G and Pollefeys M (2020) Lic-fusion 2.0: Lidar-inertial-camera odometry with sliding-window plane-feature tracking. In: *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*. Las Vegas, NV, pp. 5112–5119.

Zuo X, Zhang M, Chen Y, liu Y, Huang G and Li M (2019b) Visual-inertial localization for skid-steering robots with kinematic constraints. In: *Proc. of the International Symposium on Robotics Research (ISRR)*. Hanoi, Vietnam, pp. 741–756.

A Jacobians of Camera Measurements

Given the nested functions (see, Eq. (9)), we can leverage the chain rule to find the full Jacobian matrix of the camera measurement model in respect to the state:

$$\mathbf{H}_C := \frac{\partial \mathbf{z}_C}{\partial \mathbf{x}_k} = \underbrace{\frac{\partial \mathbf{h}_d}{\partial \mathbf{z}_n}}_{\mathbf{H}_{d1}} \underbrace{\frac{\partial \mathbf{h}_\rho}{\partial {}^{C_k}\mathbf{p}_F}}_{\mathbf{H}_\rho} \underbrace{\frac{\partial \mathbf{h}_t}{\partial \mathbf{x}_k}}_{\mathbf{H}_t} + \underbrace{\frac{\partial \mathbf{h}_d}{\partial \mathbf{x}_{CI}}}_{\mathbf{H}_{d2}}$$

Note the second term of the above equation actually should be $\frac{\partial \mathbf{h}_d}{\partial \mathbf{x}_{CI}} \frac{\partial \mathbf{x}_{CI}}{\partial \mathbf{x}_k}$ which the additional term $\frac{\partial \mathbf{x}_{CI}}{\partial \mathbf{x}_k}$ maps $\frac{\partial \mathbf{h}_d}{\partial \mathbf{x}_{CI}}$ into a proper position in the full Jacobian. However, we will omit the trivial mapping terms for brevity.

A.1 Distortion

The Jacobian with respect to the normalized coordinates can be obtained as:

$$\mathbf{H}_{d1} = \begin{bmatrix} h_{d11} & h_{d12} \\ h_{d21} & h_{d22} \end{bmatrix}$$

$$\begin{aligned} h_{d11} =& f_x((1 + k_1 r^2 + k_2 r^4) + (2k_1 x_n^2 + 4k_2 x_n^2 (x_n^2 + y_n^2)) \\ &+ 2p_1 y_n + (2p_2 x_n + 4p_2 x_n)) \\ h_{d12} =& f_x(2k_1 x_n y_n + 4k_2 x_n y_n (x_n^2 + y_n^2) + 2p_1 x_n + 2p_2 y_n) \\ h_{d21} =& f_y(2k_1 x_n y_n + 4k_2 x_n y_n (x_n^2 + y_n^2) + 2p_1 x_n + 2p_2 y_n) \\ h_{d22} =& f_y((1 + k_1 r^2 + k_2 r^4) + (2k_1 y_n^2 + 4k_2 y_n^2 (x_n^2 + y_n^2)) \\ &+ (2p_1 y_n + 4p_1 y_n) + 2p_2 x_n) \end{aligned}$$

and the Jacobian in respect to the intrinsic parameters:

$$\mathbf{H}_{d2} = \begin{bmatrix} x\ 0\ 1\ 0\ f_x x_n r^2\ f_x x_n r^4\ \ 2f_x x_n y_n) & f_x(r^2 + 2x_n^2) \\ 0\ y\ 0\ 1\ f_y y_n r^2\ f_y y_n r^4\ f_y(r^2 + 2y_n^2) & 2f_y x_n y_n \end{bmatrix}$$

where definition of each variable x, y, x_n, y_n, f_x, f_y, c_x, c_y, k_1, k_2, p_1, p_2, and r can be found in Sec. 3.2.1.

A.2 Perspective Projection

The Jacobian matrix is defined as follows:

$$\mathbf{H}_\rho = \begin{bmatrix} \frac{1}{{}^{C_k}z} & 0 & \frac{-{}^{C_k}x}{({}^{C_k}z)^2} \\ 0 & \frac{1}{{}^{C_k}z} & \frac{-{}^{C_k}y}{({}^{C_k}z)^2} \end{bmatrix}$$

where ${}^{C_k}\mathbf{p}_F = [{}^{C_k}x\ {}^{C_k}y\ {}^{C_k}z]^\top$.

A.3 Euclidean Transformation

The Jacobian matrix of Euclidean transformation can be represented using the chain rule to the camera pose (interpolated) and the camera feature:

$$\mathbf{H}_t = \begin{bmatrix} \frac{\partial \mathbf{h}_t}{\partial {}^{C_k}_E\boldsymbol{\theta}} & \frac{\partial \mathbf{h}_t}{\partial {}^E\mathbf{p}_{C_k}} \end{bmatrix} \mathbf{H}^A_{C,k} + \frac{\partial \mathbf{h}_t}{\partial {}^E\mathbf{p}_F} := \mathbf{H}_e \mathbf{H}^A_{C,k} + \mathbf{H}_F$$

$$\frac{\partial \mathbf{h}_t}{\partial {}^{C_k}_E\boldsymbol{\theta}} = \lfloor {}^{C_k}\mathbf{p}_F \rfloor, \quad \frac{\partial \mathbf{h}_t}{\partial {}^E\mathbf{p}_{C_k}} = -{}^{C_k}_E\mathbf{R}, \quad \frac{\partial \mathbf{h}_t}{\partial {}^E\mathbf{p}_F} = {}^{C_k}_E\mathbf{R}$$

where the definition of remaining chain $\mathbf{H}^A_{C,k}$, the camera pose to the interpolating IMU poses, and the spatiotemporal extrinsic calibration parameters, can be found in Appendix F (linear interpolation) or Appendix G (high-order interpolation). The final form of the camera Jacobian matrix is:

$$\mathbf{H}_C = \mathbf{H}_{d1} \mathbf{H}_\rho \mathbf{H}_e \mathbf{H}^A_{C,k} + \mathbf{H}_{d1} \mathbf{H}_\rho \mathbf{H}_F + \mathbf{H}_{d2}$$

B Wheel Odometry Measurements

Here, we introduce the essential derivations of how the wheel measurement model (and its linearization) is formulated. For the full derivations, please refer to our previous tech report (Lee and Huang 2020). The overall process is outlined in Algorithm 1.

Algorithm 1 Wheel Odometry Measurement Update

procedure WHEEL_UPDATE($\mathbf{x}_{k+1}$, $\{\omega_{ml}, \omega_{mr}\}_{k:k+1}$)
  // Preintegrate measurement, Jacobian, and noise
  $\mathbf{g}_O = \mathbf{0}_{3\times 1}$, $\mathbf{G}_O = \mathbf{0}_3$, $\mathbf{R}_O = \mathbf{0}_3$
  for $\omega_{ml,\tau}, \omega_{mr,\tau} \in \{\omega_{ml}, \omega_{mr}\}_{k:k+1}$ **do**
    $\mathbf{g}_O \leftarrow \mathbf{g}_O + \Delta\mathbf{g}$
    $\mathbf{R}_O \leftarrow \boldsymbol{\Phi}_{tr,\tau} \mathbf{R}_O \boldsymbol{\Phi}^\top_{tr,\tau} + \boldsymbol{\Phi}_{n,\tau} \mathbf{Q}_\tau \boldsymbol{\Phi}^\top_{n,\tau}$
    $\mathbf{G}_O \leftarrow \boldsymbol{\Phi}_{tr,\tau} \mathbf{G}_O + \boldsymbol{\Phi}_{OI,\tau}$
  end for
  // Compute residual and Jacobian
  $\tilde{\boldsymbol{z}}_O = \mathbf{g}_O(\hat{\mathbf{x}}_{OI}) - \mathbf{h}_O(\hat{\mathbf{x}}_k)$
  $\mathbf{H}_O = \frac{\partial \mathbf{h}_O}{\partial \tilde{\mathbf{x}}_k} - \mathbf{G}_O \frac{\partial \tilde{\mathbf{x}}_{OI}}{\partial \tilde{\mathbf{x}}_k}$
  // Perform χ^2 test & update
  if $\chi^2(\tilde{\boldsymbol{z}}, \mathbf{H}_O, \mathbf{R}_O) == Pass$ **then**
    EKF_Update($\hat{\mathbf{x}}_{k+1}, \tilde{\boldsymbol{z}}_O, \mathbf{H}_O, \mathbf{R}_O$)
  end if
end procedure

B.1 Wheel Odometry Preintegration

Assume we are in the process of wheel odometry preintegration (see Eq. (26)) and it is the turn to integrate measurements at t_τ. Based on the kinematic model (see Eq. (25)) we perform following integration of measurements:

$${}^{O_{\tau+1}}_{O_k}\theta \approx {}^{O_\tau}_{O_k}\theta - {}^{O_\tau}\omega \Delta t \tag{91}$$

$${}^{O_k}x_{O_{\tau+1}} \approx {}^{O_k}x_{O_\tau} - {}^{O_\tau}v(\sin({}^{O_{\tau+1}}_{O_k}\theta) - \sin({}^{O_\tau}_{O_k}\theta))/{}^{O_\tau}\omega \tag{92}$$

$${}^{O_k}y_{O_{\tau+1}} \approx {}^{O_k}y_{O_\tau} - {}^{O_\tau}v(\cos({}^{O_{\tau+1}}_{O_k}\theta) - \cos({}^{O_\tau}_{O_k}\theta))/{}^{O_\tau}\omega \tag{93}$$

where $\Delta t = t_{\tau+1} - t_\tau$. Note that we assume constant ${}^{O_\tau}\omega$ and ${}^{O_\tau}v$ (discrete sensor model) but considered the change of heading angle between t_τ and $t_{\tau+1}$ so that we have a more accurate model than assuming it constant. Repeating the process until t_{k+1} finishes wheel odometry preintegration.

B.2 Jacobian of Wheel-Encoder Intrinsics

As evident from Eq. (27), the wheel odometry integration entangles the intrinsic $\mathbf{x}_{OI}$, therefore the linearization would yield the following form:

$$\mathbf{z}_O \simeq \mathbf{g}_O(\{\omega_{ml}, \omega_{mr}\}_{k:k+1}, \hat{\mathbf{x}}_{OI}) + \underbrace{\frac{\partial \mathbf{g}_O}{\partial \tilde{\mathbf{x}}_{OI}}}_{\mathbf{G}_O} \tilde{\mathbf{x}}_{OI} + \frac{\partial \mathbf{g}_O}{\partial \mathbf{n}_w} \mathbf{n}_w$$

where $\mathbf{n}_\omega$ is the stacked noise vector whose τ-th block is corresponding to the encoder measurement noise at $t_\tau \in [t_k, t_{k+1}]$ (i.e., $[n_{\omega_{l,\tau}}\ n_{\omega_{r,\tau}}]^\top$, see (23)).

Clearly, performing EKF update with this measurement requires the Jacobians with respect to both the intrinsics and the noise. It is important to note that as the preintegration of $\mathbf{g}_O(\cdot)$ is computed incrementally using the encoders' measurements in the interval $[t_k, t_{k+1}]$, we accordingly calculate the measurement Jacobians incrementally one step at a time. Note also that since the noise Jacobian and $\mathbf{n}_\omega$ are often of high dimensions and may be computationally expensive when computing the stacked noise covariance during the update, we instead compute the noise covariance $\mathbf{R}_O$ by performing small matrix operations at each step.

Now we get the Jacobian of t_τ step integration from Eq. (91), (92), and (93) (again, the full derivations, which this margin is too narrow to contain, can be found in our previous tech report (Lee and Huang 2020), thus here we only show

the structure of them):

$$
\begin{aligned}
{}^{O_{\tau+1}}_{O_k}\tilde{\theta} &= {}^{O_\tau}_{O_k}\tilde{\theta} + \mathbf{\Phi}_1\tilde{\mathbf{x}}_{OI} + \mathbf{\Phi}_2\mathbf{n}_{\omega,\tau} \\
{}^{O_k}\tilde{x}_{O_{\tau+1}} &= {}^{O_k}\tilde{x}_{O_\tau} + \mathbf{\Phi}_3{}^{O_\tau}_{O_k}\tilde{\theta} + \mathbf{\Phi}_4\tilde{\mathbf{x}}_{OI} + \mathbf{\Phi}_5\mathbf{n}_{\omega,\tau} \\
{}^{O_k}\tilde{y}_{O_{\tau+1}} &= {}^{O_k}\tilde{y}_{O_\tau} + \mathbf{\Phi}_6{}^{O_\tau}_{O_k}\tilde{\theta} + \mathbf{\Phi}_7\tilde{\mathbf{x}}_{OI} + \mathbf{\Phi}_8\mathbf{n}_{\omega,\tau}
\end{aligned}
$$

It can be found that the error of $\tau+1$ step preintegration is the linear combination of τ step preintegration and measurement errors. With the above equations, we can recursively compute the noise covariance $\mathbf{R}_O$ and the Jacobian $\mathbf{G}_O$ as follows:

$$
\begin{aligned}
&\mathbf{\Phi}_{tr,\tau} = \begin{bmatrix} 1 & 0 & 0 \\ \mathbf{\Phi}_3 & 1 & 0 \\ \mathbf{\Phi}_6 & 0 & 1 \end{bmatrix}, \ \mathbf{\Phi}_{OI,\tau} = \begin{bmatrix} \mathbf{\Phi}_1 \\ \mathbf{\Phi}_4 \\ \mathbf{\Phi}_7 \end{bmatrix}, \ \mathbf{\Phi}_{n,\tau} = \begin{bmatrix} \mathbf{\Phi}_2 \\ \mathbf{\Phi}_5 \\ \mathbf{\Phi}_8 \end{bmatrix} \\
&\mathbf{R}_O \leftarrow \mathbf{\Phi}_{tr,\tau}\mathbf{R}_O\mathbf{\Phi}_{tr,\tau}^\top + \mathbf{\Phi}_{n,\tau}\mathbf{Q}_\tau\mathbf{\Phi}_{n,\tau}^\top \\
&\mathbf{G}_O \leftarrow \mathbf{\Phi}_{tr,\tau}\mathbf{G}_O + \mathbf{\Phi}_{OI,\tau}
\end{aligned}
$$

where $\mathbf{Q}_\tau$ is the noise covariance of wheel encoder measurement at t_τ. We can recursively compute the measurement noise covariance $\mathbf{R}_O$ and the Jacobian matrix $\frac{\partial \mathbf{g}_O}{\partial \tilde{\mathbf{x}}_{OI}}$ at the end of preintegration t_{k+1}, based on the zero initial condition (i.e., $\mathbf{R}_O = \mathbf{G}_O = \mathbf{0}_3$). The final structure of $\mathbf{G}_O$ can be shown as:

$$
\mathbf{G}_O = \begin{bmatrix} \mathbf{\Gamma}_{\boldsymbol{\theta}1} & \mathbf{\Gamma}_{\boldsymbol{\theta}2} & \mathbf{\Gamma}_{\boldsymbol{\theta}3} \\ \mathbf{\Gamma}_{\mathbf{x}1} & \mathbf{\Gamma}_{\mathbf{x}2} & \mathbf{\Gamma}_{\mathbf{x}3} \\ \mathbf{\Gamma}_{\mathbf{y}1} & \mathbf{\Gamma}_{\mathbf{y}2} & \mathbf{\Gamma}_{\mathbf{y}3} \end{bmatrix}
$$

Assuming n number of measurements are integrated, the structure of the element can be shown as:

$$
\begin{aligned}
\mathbf{\Gamma}_{\boldsymbol{\theta}1} &= \sum_{i=1}^{n} -\Delta t_i \frac{\omega_{l,i}}{b} \\
\mathbf{\Gamma}_{\boldsymbol{\theta}2} &= \sum_{i=1}^{n} \Delta t_i \frac{\omega_{r,i}}{b} \\
\mathbf{\Gamma}_{\boldsymbol{\theta}3} &= \sum_{i=1}^{n} -\Delta t_i \frac{{}^{O_i}\omega}{b} \\
\mathbf{\Gamma}_{\mathbf{x}1} &= \sum_{i=1}^{n} \left\{ \omega_{l,i} \left(\frac{h_{xv,i}}{2} - \frac{h_{x\omega,i}}{b} \right) - h_{x\theta,i} \sum_{j=1}^{i-1} \Delta t_j \frac{\omega_{l,j}}{b} \right\} \\
\mathbf{\Gamma}_{\mathbf{x}2} &= \sum_{i=1}^{n} \left\{ \omega_{r,i} \left(\frac{h_{xv,i}}{2} + \frac{h_{x\omega,i}}{b} \right) + h_{x\theta,i} \sum_{j=1}^{i-1} \Delta t_j \frac{\omega_{r,j}}{b} \right\} \\
\mathbf{\Gamma}_{\mathbf{x}3} &= \sum_{i=1}^{n} \left\{ -h_{x\omega,i} \frac{{}^{O_i}\omega}{b} - h_{x\theta,i} \sum_{j=1}^{i-1} \Delta t_j \frac{{}^{O_j}\omega}{b} \right\} \\
\mathbf{\Gamma}_{\mathbf{y}1} &= \sum_{i=1}^{n} \left\{ \omega_{l,i} \left(\frac{h_{yv,i}}{2} - \frac{h_{y\omega,i}}{b} \right) - h_{y\theta,i} \sum_{j=1}^{i-1} \Delta t_j \frac{\omega_{l,j}}{b} \right\} \\
\mathbf{\Gamma}_{\mathbf{y}2} &= \sum_{i=1}^{n} \left\{ \omega_{r,i} \left(\frac{h_{yv,i}}{2} + \frac{h_{y\omega,i}}{b} \right) + h_{y\theta,i} \sum_{j=1}^{i-1} \Delta t_j \frac{\omega_{r,j}}{b} \right\} \\
\mathbf{\Gamma}_{\mathbf{y}3} &= \sum_{i=1}^{n} \left\{ -h_{y\omega,i} \frac{{}^{O_i}\omega}{b} - h_{y\theta,i} \sum_{j=1}^{i-1} \Delta t_j \frac{{}^{O_j}\omega}{b} \right\}
\end{aligned}
$$

Note this derivation cannot be applied in computing covariance matrix $\mathbf{R}_O$, because the noise $\mathbf{n}_{\omega,\tau}$ has different values for every iteration unlike $\mathbf{x}_{OI}$.

B.3 Full Jacobian of Wheel Odometry

First, we show the Jacobian of wheel measurement model (see Eq. (28)) in respect to the state:

$$
\begin{aligned}
\frac{\partial \mathbf{h}_O}{\partial \tilde{\mathbf{x}}_k} &= (\mathbf{H}_{O1}\mathbf{H}_{O,k}^S + \mathbf{H}_{O2}\mathbf{H}_{O,k-1}^S)\tilde{\mathbf{x}}_k \\
\mathbf{H}_{O1} &= \begin{bmatrix} \mathbf{e}_3^\top & \mathbf{0}_{1\times3} \\ \mathbf{0}_{2\times3} & \Lambda_E^{O_{k-1}}\mathbf{R} \end{bmatrix}, \mathbf{H}_{O2} = \begin{bmatrix} -\mathbf{e}_3^\top {}^{O_k}_{O_{k-1}}\mathbf{R} & \mathbf{0}_{1\times3} \\ \Lambda \lfloor {}^{O_{k-1}}\mathbf{p}_{O_k} \rfloor & -\Lambda_E^{O_{k-1}}\mathbf{R} \end{bmatrix}
\end{aligned}
$$

where $\mathbf{H}_{O1}$ and $\mathbf{H}_{O2}$ are the Jacobian matrices of $\mathbf{h}(\cdot)$ in respect to $\{O_k\}$ and $\{O_{k-1}\}$; $\mathbf{H}_{O,k}^S$ is the Jacobian matrix of $\{O_k\}$ in respect to the $\{I_k\}$ and the wheel spatiotemporal extrinsic calibration parameters (see Appendix E for the definition).

Finally, we construct the full Jacobian matrix of wheel:

$$
\mathbf{H}_O = \frac{\partial \mathbf{h}_O}{\partial \tilde{\mathbf{x}}_k} - \frac{\partial \mathbf{g}_O}{\partial \tilde{\mathbf{x}}_{OI}} = \mathbf{H}_{O1}\mathbf{H}_{O,k}^S + \mathbf{H}_{O2}\mathbf{H}_{O,k-1}^S - \mathbf{G}_O \frac{\partial \tilde{\mathbf{x}}_{OI}}{\partial \tilde{\mathbf{x}}_k}
$$

Note that $\mathbf{G}_O$ has the minus sign in front of it because $\mathbf{g}_O$ is on the left hand side of the equation while $\mathbf{h}_O$ is on the right.

C Jacobians of LiDAR Measurements

Here we drive the Jacobian matrix of the LiDAR measurement model (see Eq. (34)). First, we linearize the model in respect to each map point ${}^M\mathbf{p}_n$, the new point ${}^M\mathbf{p}_F$ (transformed from $\{L_k\}$ to $\{M\}$), and the plane ${}^M\mathbf{\Pi}$:

$$
\tilde{\mathbf{z}}_L \approx \begin{bmatrix} \mathbf{0}_{1\times3} \\ \vdots \\ \mathbf{0}_{1\times3} \\ {}^M\mathbf{\Pi}^\top \end{bmatrix} {}^M\tilde{\mathbf{p}}_F + \begin{bmatrix} {}^M\mathbf{p}_{n_1}^\top \\ \vdots \\ {}^M\mathbf{p}_{n_m}^\top \\ {}^M\mathbf{p}_F^\top \end{bmatrix} {}^M\tilde{\mathbf{\Pi}} + \begin{bmatrix} {}^M\mathbf{\Pi}^\top\mathbf{n}_{n_1} \\ \vdots \\ {}^M\mathbf{\Pi}^\top\mathbf{n}_{n_m} \\ 0 \end{bmatrix}
$$

where $\mathbf{n}_n$ is zero mean Gaussian noise model of the map points. To complete the chain of the Jacobian matrix, we compute the Jacobian of Eq. (35) (here we show the case where the map $\{M\}$ is anchored at $\{I_{k-1}\}$, thus, $\{M\} = \{L_{k-1}\}$ and ${}^M\mathbf{p}_F = {}^{L_{k-1}}\mathbf{p}_F$):

$$
\begin{aligned}
{}^M\tilde{\mathbf{p}}_F &= \underbrace{\begin{bmatrix} \lfloor {}^{L_{k-1}}\mathbf{p}_F \rfloor & -{}^{L_{k-1}}_E\mathbf{R} \end{bmatrix}}_{\mathbf{H}_{L1}} \begin{bmatrix} {}^{L_{k-1}}_E\tilde{\boldsymbol{\theta}} \\ {}^E\tilde{\mathbf{p}}_{L_{k-1}} \end{bmatrix} \\
&\quad + \underbrace{\begin{bmatrix} {}^{L_{k-1}}_{L_k}\mathbf{R} \lfloor {}^{L_k}\mathbf{p}_F \rfloor & -{}^{L_k}_E\mathbf{R} \end{bmatrix}}_{\mathbf{H}_{L2}} \begin{bmatrix} {}^{L_k}_E\tilde{\boldsymbol{\theta}} \\ {}^E\tilde{\mathbf{p}}_{L_k} \end{bmatrix} + {}^{L_{k-1}}_{L_k}\mathbf{R}\mathbf{n}_F \\
&= (\mathbf{H}_{L1}\mathbf{H}_{L,k-1}^A + \mathbf{H}_{L2}\mathbf{H}_{L,k}^A)\tilde{\mathbf{x}}_k + {}^{L_{k-1}}_{L_k}\mathbf{R}\mathbf{n}_F
\end{aligned}
$$

where $\mathbf{n}_F$ is the zero mean Gaussian noise of the measurement. Note $\mathbf{H}_{L,k-1}^A$ and $\mathbf{H}_{L,k}^A$ are the Jacobians of LiDAR poses at t_{k-1} (actually the map) and t_k in respect to the state $\tilde{\mathbf{x}}_k$ where the definitions can be found in Appendix F (linear interpolation) or Appendix G (high-order interpolation).

Finally, we get the following linear system formulation that is a function of the state, the plane, and the noises:

$$
\tilde{\mathbf{z}}_L = \underbrace{\begin{bmatrix} \mathbf{0}_{1\times3} \\ \vdots \\ \mathbf{0}_{1\times3} \\ {}^M\mathbf{\Pi}^\top(\mathbf{H}_{L1}\mathbf{H}_{L,k-1}^A + \mathbf{H}_{L2}\mathbf{H}_{L,k}^A) \end{bmatrix}}_{\mathbf{H}_L} \tilde{\mathbf{x}}_k
$$

$$+\underbrace{\begin{bmatrix} {}^{M}\mathbf{p}_{n_1}^{\top} \\ \vdots \\ {}^{M}\mathbf{p}_{n_k}^{\top} \\ {}^{M}\mathbf{p}_{F}^{\top} \end{bmatrix}}_{\mathbf{H}_{\mathbf{\Pi}}} {}^{M}\tilde{\mathbf{\Pi}} + \underbrace{\begin{bmatrix} {}^{M}\mathbf{\Pi}^{\top}\mathbf{n}_{n_1} \\ \vdots \\ {}^{M}\mathbf{\Pi}^{\top}\mathbf{n}_{n_k} \\ {}^{M}\mathbf{\Pi}^{\top}{}_{L_k}^{L_{k-1}}\mathbf{R}\mathbf{n}_{F} \end{bmatrix}}_{\mathbf{n}_L}$$

Though the above linear system can be directly used to update the state, we additionally perform Cholesky decomposition of the last noise term (in the covariance matrix form $\mathbf{R}_L = \mathbf{L}\mathbf{L}^{\top}$) and multiply the inverse of lower triangular matrix ($\mathbf{L}^{-1}$) to the above equation (a.k.a whitening) to make the following null space projection (see Eq. (38)) simple and construct the noise covariance matrix efficiently without tracking the map and measurement noises. To be more specific, we get the following after the operation (corresponds to the Eq. (37)):

$$\tilde{\mathbf{z}}'_L = \mathbf{H}'_L\tilde{\mathbf{x}}_k + \mathbf{H}'_{\mathbf{\Pi}}{}^{M}\tilde{\mathbf{\Pi}} + \mathbf{n}'_L$$

Note the noise $\mathbf{n}'_L$ is the standard Gaussian ($\mathcal{N}(\mathbf{0}, \mathbf{I})$).

D GNSS Measurements

D.1 Jacobians

The Jacobian matrix of GNSS measurement model (see Eq. (44)) can be simply represented with two matrices (after initialization):

$$\mathbf{H}_G := \frac{\partial \mathbf{z}_G}{\partial \mathbf{x}_k} = [\mathbf{0}_3 \ \ \mathbf{I}_3]\mathbf{H}^{A}_{G,k} = \mathbf{H}_g\mathbf{H}^{A}_{G,k} \tag{94}$$

where $\mathbf{H}_g$ is the derivative of the measurement with respect to the GNSS sensor pose, and the definition of remaining chain $\mathbf{H}^{A}_{G,k}$, the GNSS sensor pose to the interpolating IMU poses and the spatiotemporal extrinsic calibration parameters, can be found in Appendix F (linear interpolation) or Appendix G (high-order interpolation).

D.2 Observability Analysis: State in $\{W\}$

For concise presentation, here we consider a simplified case where the state in the local world ${}^{W}\mathbf{x}_k$ only contains one IMU pose, linear velocity, and $\{W\}$ to $\{E\}$ transformation with perfectly synchronized and calibrated sensors (identity transformation from IMU to GNSS), while the results can be extended to general cases:

$${}^{W}\mathbf{x}_k = ({}^{I_k}_{W}\mathbf{R}, \ {}^{W}\mathbf{p}_{I_k}, \ {}^{W}\mathbf{v}_{I_k}, \ {}^{E}_{W}\mathbf{R}, \ {}^{E}\mathbf{p}_{W}) \tag{95}$$

The corresponding error state transition matrix can be shown as ($\Delta t = t_k - t_0$):

$$\mathbf{\Phi}^{W}(t_k, t_0) = \begin{bmatrix} {}^{I_k}_{W}\mathbf{R}{}^{W}_{I_0}\mathbf{R} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\lfloor {}^{W}\mathbf{p}_{I_k} - {}^{W}\mathbf{p}_{I_0} - {}^{W}\mathbf{v}_{I_0}\Delta t + \frac{1}{2}\mathbf{g}\Delta t^2 \rfloor {}^{W}_{I_0}\mathbf{R} & \mathbf{I}_3 & \Delta t\mathbf{I}_3 & \mathbf{0} \\ -\lfloor {}^{W}\mathbf{v}_{I_k} - {}^{W}\mathbf{v}_{I_0} + \mathbf{g}\Delta t \rfloor {}^{W}_{I_0}\mathbf{R} & \mathbf{0} & \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_6 \end{bmatrix}$$

The measurement model and the corresponding Jacobian matrix in respect to the local state (see Eq. (95)) are:

$$\mathbf{z}_G = {}^{E}\mathbf{p}_W + {}^{E}_{W}\mathbf{R}{}^{W}\mathbf{p}_{I_k} + \mathbf{n}_G$$

$$\mathbf{H}^{W}_{G} = \begin{bmatrix} \mathbf{0}_3 & {}^{E}_{W}\mathbf{R} & \mathbf{0}_3 & \mathbf{0}_3 & \lfloor {}^{E}_{W}\mathbf{R}{}^{W}\mathbf{p}_{I_k} \rfloor & \mathbf{I}_3 \end{bmatrix}$$

Now we can construct the observability matrix $\mathcal{O}^{W}$:

$$\mathcal{O}^{W} = \begin{bmatrix} \mathbf{H}^{W}_{G_0} \\ \mathbf{H}^{W}_{G_1}\mathbf{\Phi}^{W}(t_1, t_0) \\ \vdots \\ \mathbf{H}^{W}_{G}\mathbf{\Phi}^{W}(t_k, t_0) \\ \vdots \end{bmatrix}$$

The $(k+1)$th row matrix of $\mathcal{O}^{W}$ can be shown as:

$$\begin{aligned} \mathcal{O}^{W}_{k+1} &= \mathbf{H}^{W}_{G}\mathbf{\Phi}^{W}(t_k, t_0) = \begin{bmatrix} \mathbf{\Gamma}_1 & \mathbf{\Gamma}_2 & \mathbf{\Gamma}_3 & \mathbf{\Gamma}_4 & \mathbf{I}_3 \end{bmatrix} \\ \mathbf{\Gamma}_1 &= -{}^{E}_{W}\mathbf{R}\lfloor {}^{W}\mathbf{p}_{I_k} - {}^{W}\mathbf{p}_{I_0} - {}^{W}\mathbf{v}_{I_0}\Delta t + \frac{1}{2}\mathbf{g}\Delta t^2 \rfloor {}^{W}_{I_0}\mathbf{R} \\ \mathbf{\Gamma}_2 &= {}^{E}_{W}\mathbf{R} \\ \mathbf{\Gamma}_3 &= {}^{E}_{W}\mathbf{R}\Delta t \\ \mathbf{\Gamma}_4 &= \lfloor {}^{E}_{W}\mathbf{R}{}^{W}\mathbf{p}_{I_k} \rfloor \end{aligned}$$

It is easily can be shown that the matrix shown below is the null space of $\mathcal{O}^{W}_{k+1}$ by calculating $\mathcal{O}^{W}_{k+1} \times \mathbf{N}^{W} = \mathbf{0}$.

$$\mathbf{N}^{W} = \begin{bmatrix} \mathbf{0}_3 & -{}^{I_0}_{W}\mathbf{R}\mathbf{g} \\ \mathbf{I}_3 & \lfloor {}^{W}\mathbf{p}_{I_0} \rfloor \mathbf{g} \\ \mathbf{0}_3 & \lfloor {}^{W}\mathbf{v}_{I_0} \rfloor \mathbf{g} \\ \mathbf{0}_3 & -{}^{E}_{W}\mathbf{R}\mathbf{g} \\ -{}^{E}_{W}\mathbf{R} & \mathbf{0}_{3\times 1} \end{bmatrix}$$

As $\mathbf{N}^{W}$ is the null space of any row matrices of $\mathcal{O}^{W}$, $\mathbf{N}^{W}$ is the null space of the local state (see Eq. (95)). By inspection, the first column of $\mathbf{N}^{W}$ corresponds to the translation of $\{W\}$ to $\{E\}$ and the second column to the rotation of $\{W\}$ with respect to $\{E\}$ along the axis of gravity.

D.3 Observability Analysis: State in $\{E\}$

In analogy to Appendix D.2, here we also investigate the minimal state that has the frame of reference in $\{E\}$ as:

$${}^{E}\mathbf{x}_k = ({}^{I_k}_{E}\mathbf{R}, \ {}^{E}\mathbf{p}_{I_k}, \ {}^{E}\mathbf{v}_{I_k}, \ {}^{E}_{W}\mathbf{R}, \ {}^{E}\mathbf{p}_{W}) \tag{96}$$

The corresponding error state transition matrix is:

$$\mathbf{\Phi}^{E}(t_k, t_0) = \begin{bmatrix} {}^{I_k}_{E}\mathbf{R}{}^{E}_{I_0}\mathbf{R} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\lfloor {}^{E}\mathbf{p}_{I_k} - {}^{E}\mathbf{p}_{I_0} - {}^{E}\mathbf{v}_{I_0}\Delta t + \frac{1}{2}\mathbf{g}\Delta t^2 \rfloor {}^{E}_{I_0}\mathbf{R} & \mathbf{I}_3 & \Delta t\mathbf{I}_3 & \mathbf{0} \\ -\lfloor {}^{E}\mathbf{v}_{I_k} - {}^{E}\mathbf{v}_{I_0} + \mathbf{g}\Delta t \rfloor {}^{E}_{I_0}\mathbf{R} & \mathbf{0} & \mathbf{I}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_6 \end{bmatrix}$$

The measurement model and the corresponding Jacobian matrix in respect to the local state (see Eq. (96)) are:

$$\mathbf{z}_G = {}^{E}\mathbf{p}_{I_k} + \mathbf{n}_G$$

$$\mathbf{H}^{E}_{G} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_{3\times 9} \end{bmatrix}$$

Clearly, the multiplication of $\mathcal{O}^{E}_{k+1} := \mathbf{H}^{E}_{G}\mathbf{\Phi}^{E}(t_k, t_0)$ with $\mathbf{N}^{W}$ does not yield a zero matrix which means the 4 unobservable directions of INS (Kelly and Sukhatme 2011) are now observable.

D.4 State Transformation from $\{W\}$ to $\{E\}$

As the IMU biases and the calibration parameters remain the same (their frame of the references are fixed to, such as, IMU and sensor frame), thus have identity matrix in this Jacobian matrix of transformation, here we show the minimal relevant state case:

$$ {}^{W}\mathbf{x}_k = ({}^{I_k}_{W}\mathbf{R}, {}^{W}\mathbf{p}_{I_k}, {}^{W}\mathbf{v}_{I_k}, {}^{E}_{W}\mathbf{R}, {}^{E}\mathbf{p}_{W}) $$

We linearize the state transform function (see Eq. (86)) at current estimate to achieve the Jacobian matrix $\mathbf{\Psi}$ and propagate the error state with it as:

$$ {}^{E}\tilde{\mathbf{x}}_k \leftarrow \mathbf{\Psi}\, {}^{W}\tilde{\mathbf{x}}_k $$

where

$$ \mathbf{\Psi} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & {}^{I_k}_{W}\mathbf{R}\,{}^{E}_{W}\mathbf{R}^{\top} & \mathbf{0}_3 \\ \mathbf{0}_3 & {}^{E}_{W}\mathbf{R} & \mathbf{0}_3 & \lfloor {}^{E}_{W}\mathbf{R}\,{}^{W}\mathbf{p}_{I_k} \rfloor & \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & {}^{E}_{W}\mathbf{R} & \lfloor {}^{E}_{W}\mathbf{R}\,{}^{W}\mathbf{v}_{I_k} \rfloor & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix} $$

Please refer to our previous tech report (Lee et al. 2019) for the full derivations.

E Jacobians of Synchronous Sensors

In general, there is no reason a sensor measurement is synchronous to other sensors or to the state. However, in some cases, we can chose the sampling time of the measurements to get the synchronized sensor measurements. The preintegrated wheel measurement (see Eq. (27)), for example, is one of the case where we decide the measurement integration period and get the synchronized pose measurement. Therefore, here we introduce how the Jacobian matrices of the pose measurement of the synchronous sensors is formulated.

Assume we have a global pose measurement of sensor $\{X\}$ at time t_k. The measurement can be modeled with a IMU pose $\{I_k\}$ and spatial extrinsic calibration parameters $({}^{X}_{I}\mathbf{R}, {}^{X}\mathbf{p}_I)$ between the IMU and the sensor as:

$$ \mathbf{z}_{X_k} := \begin{bmatrix} {}^{X_k}_{E}\boldsymbol{\theta} \\ {}^{E}\mathbf{p}_{X_k} \end{bmatrix} = \begin{bmatrix} \mathrm{Log}({}^{X}_{I}\mathbf{R}\,{}^{I_k}_{E}\mathbf{R}) \\ {}^{E}\mathbf{p}_{I_k} + {}^{E}_{I_k}\mathbf{R}\,{}^{I}\mathbf{p}_X \end{bmatrix} + \mathbf{n}_{X_k} \tag{97} $$

We first drive the derivative of the measurement model in respect to the involved parameters:

$$ \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = {}^{X}_{I}\mathbf{R} \qquad \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = \mathbf{I}_3 \tag{98} $$

$$ \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = -{}^{E}_{I_k}\mathbf{R}\lfloor {}^{I}\mathbf{p}_X \rfloor \qquad \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{E}\mathbf{p}_{I_k}} = \mathbf{I}_3 \tag{99} $$

$$ \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = {}^{E}_{X_k}\mathbf{R}\lfloor {}^{X}\mathbf{p}_I \rfloor \qquad \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{X}\mathbf{p}_I} = -{}^{E}_{X_k}\mathbf{R} \tag{100} $$

To account for the difference between sensor clocks and measurement delay, we model an unknown constant time offset between the IMU clock and the sensor clock: ${}^{I}t_k = {}^{X}t_k + {}^{X}t_I$, where ${}^{I}t_k$ and ${}^{X}t_k$ are the times when measurement $\mathbf{z}_{S_k}$ was collected in the IMU and the sensor's clocks, and ${}^{X}t_I$ is the time offset between the two time references. To get the synchronized pose measurement at IMU times ${}^{I}t_k$, we use the current best estimate of the time offset ${}^{X}\hat{t}_I$ and get the measurement at ${}^{X}t_k = {}^{I}t_k - {}^{X}\hat{t}_I$, whose corresponding time in the IMU clock is:

$$ {}^{I}\bar{t}_k := {}^{I}t_k - {}^{X}\hat{t}_I + {}^{X}t_I = {}^{I}t_k + {}^{X}\tilde{t}_I $$

Note that the above equation revels the error of the time offset ${}^{X}\tilde{t}_I$ within the chosen measurement time ${}^{I}\bar{t}_k$. We employ the following first-order approximation to account for this small time-offset error (Li and Mourikis 2014a):

$$ {}^{I({}^{I}\bar{t}_k)}_{E}\mathbf{R} = {}^{I({}^{I}t_k + {}^{X}\tilde{t}_I)}_{E}\mathbf{R} \approx (\mathbf{I} - \lfloor {}^{I_k}\boldsymbol{\omega}\, {}^{X}\tilde{t}_I \rfloor)\, {}^{I_k}_{E}\mathbf{R} $$
$$ {}^{E}\mathbf{p}_{I({}^{I}\bar{t}_k)} = {}^{E}\mathbf{p}_{I({}^{I}t_k + {}^{X}\tilde{t}_I)} \approx {}^{E}\mathbf{p}_{I_k} + {}^{E}\mathbf{v}_{I_k}\, {}^{X}\tilde{t}_I $$

By replacing the above approximation to Eq. (97), we get the following additional Jacobians which are related to the time offset:

$$ \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{X}t_I} = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_k}_{E}\boldsymbol{\theta}}{\partial {}^{X}t_I} = {}^{X}_{I}\mathbf{R}\, {}^{I_k}\boldsymbol{\omega} $$

$$ \begin{aligned} \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{X}t_I} &= \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{E}\mathbf{p}_{I_k}} \frac{\partial {}^{E}\mathbf{p}_{I_k}}{\partial {}^{X}t_I} + \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_k}_{E}\boldsymbol{\theta}}{\partial {}^{X}t_I} \\ &= {}^{E}\mathbf{v}_{I_k} - {}^{E}_{I_k}\mathbf{R}\lfloor {}^{I}\mathbf{p}_X \rfloor\, {}^{I_k}\boldsymbol{\omega} \end{aligned} $$

Finally, we get the following full Jacobian matrix of the synchronized measurement:

$$ \begin{aligned} \mathbf{H}^{S}_{X,k} &= \frac{\partial \mathbf{z}_{X_k}}{\partial \mathbf{x}_k} \\ &= \begin{bmatrix} \underbrace{\begin{matrix} \mathbf{H}^{S}_{00} & \mathbf{0} \\ \mathbf{H}^{S}_{10} & \mathbf{H}^{S}_{11} \end{matrix}}_{\{I_k\}} & \mathbf{0} & \cdots & \underbrace{\begin{matrix} \mathbf{H}^{S}_{07} \\ \mathbf{H}^{S}_{17} \end{matrix}}_{{}^{X}_{I}\mathbf{R}} & \underbrace{\begin{matrix} \mathbf{0} \\ \mathbf{H}^{S}_{18} \end{matrix}}_{{}^{X}\mathbf{p}_I} & \underbrace{\begin{matrix} \mathbf{H}^{S}_{09} \\ \mathbf{H}^{S}_{19} \end{matrix}}_{{}^{X}t_I} & \mathbf{0} & \cdots \end{bmatrix} \end{aligned} $$

$$ \mathbf{H}^{S}_{00} = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = {}^{X}_{I}\mathbf{R} $$

$$ \mathbf{H}^{S}_{07} = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = \mathbf{I}_3 $$

$$ \mathbf{H}^{S}_{09} = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{X}t_I} = {}^{X}_{I}\mathbf{R}\, {}^{I_k}\boldsymbol{\omega} $$

$$ \mathbf{H}^{S}_{10} = \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = -{}^{E}_{I_k}\mathbf{R}\lfloor {}^{I}\mathbf{p}_X \rfloor $$

$$ \mathbf{H}^{S}_{11} = \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{E}\mathbf{p}_{I_k}} = \mathbf{I}_3 $$

$$ \mathbf{H}^{S}_{17} = \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = {}^{E}_{X_k}\mathbf{R}\lfloor {}^{X}\mathbf{p}_I \rfloor $$

$$ \mathbf{H}^{S}_{18} = \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{I}\mathbf{p}_X} = -{}^{E}_{X_k}\mathbf{R} $$

$$ \mathbf{H}^{S}_{19} = \frac{\partial {}^{E}\mathbf{p}_{X_k}}{\partial {}^{X}t_I} = {}^{E}\mathbf{v}_{I_k} - {}^{E}_{I_k}\mathbf{R}\lfloor {}^{I}\mathbf{p}_X \rfloor\, {}^{I_k}\boldsymbol{\omega} $$

F Jacobians of Asynchronous Sensors: Linear Interpolation

Here we assume an asynchronous global pose measurement (see Eq. (46)) of a sensor $\{X\}$ is given at time t'_k ($t_{k\text{-}1} \leq t'_k + {}^{X}t_I \leq t_k$) and drive the full Jacobian of the measurement model in respect to the two bounding IMU

poses ($\{I_{k-1}\}$ and $\{I_k\}$) and the spatiotemporal extrinsic calibration parameters (${}^X_I\mathbf{R}$, ${}^X\mathbf{p}_I$, and ${}^Xt_I$) using the linear interpolation. The measurement model and the interpolation can be written as:

$$\mathbf{z}_{X_k} = \begin{bmatrix} {}^{X_{k'}}_{E}\boldsymbol{\theta} \\ {}^E\mathbf{p}_{X_{k'}} \end{bmatrix} = \begin{bmatrix} \mathrm{Log}({}^X_I\mathbf{R}\,{}^{I_{k'}}_{E}\mathbf{R}) \\ {}^E\mathbf{p}_{I_{k'}} + {}^E_{I_{k'}}\mathbf{R}\,{}^I\mathbf{p}_X \end{bmatrix} + \mathbf{n}_{X_k}$$
$${}^{I_{k'}}_{E}\mathbf{R} = \mathrm{Exp}(\lambda \mathrm{Log}({}^{I_k}_{E}\mathbf{R}\,{}^{E}_{I_{k-1}}\mathbf{R}))\,{}^{I_{k-1}}_{E}\mathbf{R}$$
$${}^E\mathbf{p}_{I_{k'}} = (1-\lambda)\,{}^E\mathbf{p}_{I_{k-1}} + \lambda\,{}^E\mathbf{p}_{I_k}$$
$$\lambda = (t'_k + {}^Xt_I - t_{k-1})/(t_k - t_{k-1})$$

The derivative of the measurement model in respect to the involved parameters are the same as synchronized measurement (see Eq. (98), (99), and (100)). Then the derivative of the interpolated IMU pose in respect to the bounding IMU poses and the temporal extrinsic calibration parameter are derived as:

$$\frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_{k-1}}_{E}\boldsymbol{\theta}} = \boldsymbol{\Upsilon}_1 = \mathbf{J_r}(\lambda {}^{I_{k-1}}_{I_k}\boldsymbol{\theta})(\mathbf{J_l^{-1}}(\lambda {}^{I_{k-1}}_{I_k}\boldsymbol{\theta}) - \lambda \mathbf{J_l^{-1}}({}^{I_{k-1}}_{I_k}\boldsymbol{\theta}))$$
$$\frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = \boldsymbol{\Upsilon}_2 = \lambda \mathbf{J_r}(\lambda {}^{I_{k-1}}_{I_k}\boldsymbol{\theta})\mathbf{J_r^{-1}}({}^{I_{k-1}}_{I_k}\boldsymbol{\theta})$$
$$\frac{\partial {}^E\mathbf{p}_{I_{k'}}}{\partial {}^E\mathbf{p}_{I_{k-1}}} = (1-\lambda)\mathbf{I}_3 \qquad \frac{\partial {}^E\mathbf{p}_{I_{k'}}}{\partial {}^E\mathbf{p}_{I_k}} = \lambda \mathbf{I}_3$$
$$\frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^Xt_I} = \frac{{}^{I_{k-1}}_{I_k}\boldsymbol{\theta}}{t_k - t_{k-1}} \qquad \frac{\partial {}^E\mathbf{p}_{I_{k'}}}{\partial {}^Xt_I} = \frac{{}^E\mathbf{p}_{I_k} - {}^E\mathbf{p}_{I_{k-1}}}{t_k - t_{k-1}}$$

where $\mathbf{J_l}$ and $\mathbf{J_r}$ are the left and right Jacobian matrices of SO(3) (Chirikjian 2011). Finally, we apply the chain rule to get the full Jacobian matrix $\mathbf{H}^A_{X,k}$:

$$\mathbf{H}^A_{X,k} = \frac{\partial \mathbf{z}_{X_k}}{\partial \mathbf{x}_k} = \begin{bmatrix} \mathbf{H}^A_1 & \mathbf{0} & \mathbf{0} & \mathbf{H}^A_2 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{H}^A_3 & \mathbf{0} & \mathbf{H}^A_4 & \cdots \\ \underbrace{\mathbf{H}^A_5 \;\; \mathbf{H}^A_6}_{\{I_k\}} & \mathbf{0} & \underbrace{\mathbf{H}^A_7 \;\; \mathbf{H}^A_8}_{\{I_{k-1}\}} & \mathbf{0} & \cdots & \underbrace{\mathbf{H}^A_9}_{{}^X_I\mathbf{R}} \; \underbrace{\mathbf{H}^A_{10}}_{{}^X\mathbf{p}_I} \; \underbrace{\mathbf{H}^A_{11}}_{{}^Xt_I} & \cdots \end{bmatrix}$$
$$\mathbf{H}^A_1 = \frac{\partial {}^{X_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = {}^X_I\mathbf{R}\boldsymbol{\Upsilon}_2$$
$$\mathbf{H}^A_2 = \frac{\partial {}^{X_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_{k-1}}_{E}\boldsymbol{\theta}} = {}^X_I\mathbf{R}\boldsymbol{\Upsilon}_1$$
$$\mathbf{H}^A_3 = \frac{\partial {}^{X_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = \mathbf{I}_3$$
$$\mathbf{H}^A_4 = \frac{\partial {}^{X_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^At_I} = {}^X_I\mathbf{R}\frac{1}{t_k - t_{k-1}} {}^{I_{k-1}}_{I_k}\boldsymbol{\theta}$$
$$\mathbf{H}^A_5 = \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = -{}^E_{I_{k'}}\mathbf{R}\lfloor {}^I\mathbf{p}_X \rfloor \boldsymbol{\Upsilon}_2$$
$$\mathbf{H}^A_6 = \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^E\mathbf{p}_{I_{k'}}} \frac{\partial {}^E\mathbf{p}_{I_{k'}}}{\partial {}^E\mathbf{p}_{I_k}} = \lambda \mathbf{I}_3$$
$$\mathbf{H}^A_7 = \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^{I_{k-1}}_{E}\boldsymbol{\theta}} = -{}^E_{I_{k'}}\mathbf{R}\lfloor {}^I\mathbf{p}_X \rfloor \boldsymbol{\Upsilon}_1$$
$$\mathbf{H}^A_8 = \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^E\mathbf{p}_{I_{k'}}} \frac{\partial {}^E\mathbf{p}_{I_{k'}}}{\partial {}^E\mathbf{p}_{I_{k-1}}} = (1-\lambda)\mathbf{I}_3$$
$$\mathbf{H}^A_9 = \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = {}^E_{X_{k'}}\mathbf{R}\lfloor {}^X\mathbf{p}_I \rfloor$$
$$\mathbf{H}^A_{10} = \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^X\mathbf{p}_I} = -{}^E_{X_{k'}}\mathbf{R}$$
$$\mathbf{H}^A_{11} = \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_{k'}}_{E}\boldsymbol{\theta}}{\partial {}^At_I} + \frac{\partial {}^E\mathbf{p}_{X_{k'}}}{\partial {}^E\mathbf{p}_{I_{k'}}} \frac{\partial {}^E\mathbf{p}_{I_{k'}}}{\partial {}^At_I} = \frac{{}^E\mathbf{p}_{I_k} - {}^E\mathbf{p}_{I_{k-1}} - {}^E_{I_{k'}}\mathbf{R}\lfloor {}^I\mathbf{p}_X \rfloor {}^{I_{k-1}}_{I_k}\boldsymbol{\theta}}{t_k - t_{k-1}}$$

G Jacobians of Asynchronous Sensors: High-order Interpolation

Here we assume an asynchronous global pose measurement (see Eq. (46)) of a sensor $\{X\}$ is given at time t_k and drive the full Jacobian of the measurement model in respect to the IMU poses (say $\{I_0\}, \ldots, \{I_n\}$) used in $n+1$ order interpolation and the spatiotemporal extrinsic calibration parameters (${}^X_I\mathbf{R}$, ${}^X\mathbf{p}_I$, and ${}^Xt_I$). The measurement model and the interpolation can be written as (Eckenhoff et al. 2021):

$$\mathbf{z}_{X_k} = \begin{bmatrix} {}^{X_k}_{E}\boldsymbol{\theta} \\ {}^E\mathbf{p}_{X_k} \end{bmatrix} = \begin{bmatrix} \mathrm{Log}({}^X_I\mathbf{R}\,{}^{I_k}_{E}\mathbf{R}) \\ {}^E\mathbf{p}_{I_k} + {}^E_{I_k}\mathbf{R}\,{}^I\mathbf{p}_X \end{bmatrix} + \mathbf{n}_{X_k}$$
$${}^{I_k}_{E}\mathbf{R} = \mathrm{Exp}(\sum_{i=1}^{n} \mathbf{a}_i \Delta t_k^i)\,{}^{I_0}_{E}\mathbf{R}$$
$${}^E\mathbf{p}_{I_k} = {}^E\mathbf{p}_{I_0} + \sum_{i=1}^{n} \mathbf{b}_i \Delta t_k^i$$

where $\Delta t_k = t_k + {}^Xt_I - t_0$, Finally, we apply the chain rule to get the full Jacobian matrix $\mathbf{H}^A_{X,k}$:

$$\mathbf{H}^A_{X,k} = \frac{\partial \mathbf{z}_{X_k}}{\partial \mathbf{x}_k} = \begin{bmatrix} \cdots & \mathbf{H}^A_1 & \mathbf{0} & \cdots & \mathbf{H}^A_2 & \mathbf{0} & \cdots & \mathbf{H}^A_3 & \mathbf{0} & \mathbf{H}^A_4 & \cdots \\ \cdots & \underbrace{\mathbf{H}^A_5 \;\; \mathbf{H}^A_6}_{\{I_i\}} & \cdots & \underbrace{\mathbf{H}^A_7 \;\; \mathbf{H}^A_8}_{\{I_0\}} & \cdots & \underbrace{\mathbf{H}^A_9}_{{}^X_I\mathbf{R}} \; \underbrace{\mathbf{H}^A_{10}}_{{}^X\mathbf{p}_I} & \underbrace{\mathbf{H}^A_{11}}_{{}^Xt_I} & \cdots \end{bmatrix}$$
$$\mathbf{H}^A_1 = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_i}_{E}\boldsymbol{\theta}} = -{}^X_I\mathbf{R}\mathbf{J_l}(\mathbf{M}_t\mathbf{a})\mathbf{M}_t\mathbf{V}^{-1}\mathbf{J}_{\mathbf{A},\mathbf{0}}$$
$$\mathbf{H}^A_2 = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_0}_{E}\boldsymbol{\theta}} = -{}^X_I\mathbf{R}(\mathbf{J_l}(\mathbf{M}_t\mathbf{a})\mathbf{M}_t\mathbf{V}^{-1}\mathbf{J}_{\mathbf{A},\mathbf{i}} + \mathrm{Exp}(\mathbf{M}_t\mathbf{a}))$$
$$\mathbf{H}^A_3 = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = \mathbf{I}_3$$
$$\mathbf{H}^A_4 = \frac{\partial {}^{X_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_k}_{E}\boldsymbol{\theta}}{\partial {}^At_I} = -{}^X_I\mathbf{R}\mathbf{J_l}(\mathbf{M}_t\mathbf{a})\mathbf{M}_{dt}\mathbf{a}$$
$$\mathbf{H}^A_5 = \frac{\partial {}^E\mathbf{p}_{X_k}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} = {}^E_{I_k}\mathbf{R}\lfloor {}^I\mathbf{p}_X \rfloor \mathbf{J_l}(\mathbf{M}_t\mathbf{a})\mathbf{M}_t\mathbf{V}^{-1}\mathbf{J}_{\mathbf{A},\mathbf{0}}$$
$$\mathbf{H}^A_6 = \frac{\partial {}^E\mathbf{p}_{X_k}}{\partial {}^E\mathbf{p}_{I_k}} \frac{\partial {}^E\mathbf{p}_{I_k}}{\partial {}^E\mathbf{p}_{I_k}} = \mathbf{I}_3 + \mathbf{M}_t\mathbf{V}^{-1}\mathbf{J}_{\mathbf{B},\mathbf{0}}$$
$$\mathbf{H}^A_7 = \frac{\partial {}^E\mathbf{p}_{X_k}}{\partial {}^{I_k}_{E}\boldsymbol{\theta}} \frac{\partial {}^{I_k}_{E}\boldsymbol{\theta}}{\partial {}^{I_{k-1}}_{E}\boldsymbol{\theta}} = {}^E_{I_k}\mathbf{R}\lfloor {}^I\mathbf{p}_X \rfloor (\mathbf{J_l}(\mathbf{M}_t\mathbf{a})\mathbf{M}_t\mathbf{V}^{-1}\mathbf{J}_{\mathbf{A},\mathbf{i}} + \mathrm{Exp}(\mathbf{M}_t\mathbf{a}))$$
$$\mathbf{H}^A_8 = \frac{\partial {}^E\mathbf{p}_{X_k}}{\partial {}^E\mathbf{p}_{I_k}} \frac{\partial {}^E\mathbf{p}_{I_k}}{\partial {}^E\mathbf{p}_{I_{k-1}}} = \mathbf{M}_t\mathbf{V}^{-1}\mathbf{J}_{\mathbf{B},\mathbf{i}}$$
$$\mathbf{H}^A_9 = \frac{\partial {}^E\mathbf{p}_{X_k}}{\partial {}^{X}_{I}\boldsymbol{\theta}} = {}^E_{X_k}\mathbf{R}\lfloor {}^X\mathbf{p}_I \rfloor$$

$$\mathbf{H}_{10}^{A} = \frac{\partial^{E}\mathbf{p}_{X_k}}{\partial^{X}\mathbf{p}_{I}} = -{}^{E}_{X_k}\mathbf{R}$$

$$\begin{aligned}\mathbf{H}_{11}^{A} &= \frac{\partial^{E}\mathbf{p}_{X_k}}{\partial^{I_k}_{E}\boldsymbol{\theta}}\frac{\partial^{I_k}_{E}\boldsymbol{\theta}}{\partial^{A}t_{I}} + \frac{\partial^{E}\mathbf{p}_{X_k}}{\partial^{E}\mathbf{p}_{I_k}}\frac{\partial^{E}\mathbf{p}_{I_k}}{\partial^{A}t_{I}} \\ &= {}^{E}_{I_k}\mathbf{R}\lfloor {}^{I}\mathbf{p}_{X}\rfloor \mathbf{J}_{\mathbf{l}}(\mathbf{M}_{t}\mathbf{a})\mathbf{M}_{dt}\mathbf{a} + \mathbf{M}_{dt}\mathbf{b}\end{aligned}$$

where

$$\mathbf{M}_{t} = \begin{bmatrix}\Delta t_k\mathbf{I}_3 & \Delta t_k^2\mathbf{I}_3 & \cdots & \Delta t_k^n\mathbf{I}_3\end{bmatrix}$$

$$\mathbf{M}_{dt} = \begin{bmatrix}\mathbf{I}_3 & 2\Delta t_k\mathbf{I}_3 & \cdots & n\Delta t_k^{n-1}\mathbf{I}_3\end{bmatrix}$$

$$\mathbf{V} = \begin{bmatrix}\Delta t_1 & \cdots & \Delta t_1^n \\ \vdots & \ddots & \vdots \\ \Delta t_n & \cdots & \Delta t_n^n\end{bmatrix}$$

$$\mathbf{a} = \begin{bmatrix}\mathbf{a}_1^{\top} & \mathbf{a}_2^{\top} & \cdots & \mathbf{a}_n^{\top}\end{bmatrix}^{\top}$$

$$\mathbf{b} = \begin{bmatrix}\mathbf{b}_1^{\top} & \mathbf{b}_2^{\top} & \cdots & \mathbf{b}_n^{\top}\end{bmatrix}^{\top}$$

$$\mathbf{J}_{\mathbf{A},\mathbf{0}} = \begin{bmatrix}-\mathbf{J}_{\mathbf{l}}^{-\mathbf{1}}(\mathrm{Log}({}^{I_1}_{E}\mathbf{R}^{E}_{I_0}\mathbf{R})) \\ \vdots \\ -\mathbf{J}_{\mathbf{l}}^{-\mathbf{1}}(\mathrm{Log}({}^{I_n}_{E}\mathbf{R}^{E}_{I_0}\mathbf{R}))\end{bmatrix}$$

$$\mathbf{J}_{\mathbf{A},\mathbf{i}} = \begin{bmatrix}\mathbf{0}_3 \\ \vdots \\ \mathbf{J}_{\mathbf{l}}^{-\mathbf{1}}(\mathrm{Log}({}^{I_i}_{E}\mathbf{R}^{E}_{I_0}\mathbf{R}))^{I_i}_{E}\mathbf{R}^{E}_{I_0}\mathbf{R} \\ \vdots \\ \mathbf{0}_3\end{bmatrix}$$

$$\mathbf{J}_{\mathbf{B},\mathbf{0}} = \begin{bmatrix}-\mathbf{I}_3 & \cdots & -\mathbf{I}_3\end{bmatrix}^{\top}$$

$$\mathbf{J}_{\mathbf{B},\mathbf{i}} = \begin{bmatrix}\mathbf{0}_3 & \cdots & \mathbf{I}_3 & \cdots & \mathbf{0}_3\end{bmatrix}^{\top}$$