

R²LIVE: A Robust, Real-time, LiDAR-Inertial-Visual tightly-coupled state Estimator and mapping

Jiarong Lin, Chunran Zheng, Wei Xu, and Fu Zhang

Abstract—In this letter, we propose a robust, real-time tightly-coupled multi-sensor fusion framework, which fuses measurements from LiDAR, inertial sensor, and visual camera to achieve robust and accurate state estimation. Our proposed framework is composed of two parts: the filter-based odometry and factor graph optimization. To guarantee real-time performance, we estimate the state within the framework of error-state iterated Kalman-filter, and further improve the overall precision with our factor graph optimization. Taking advantage of measurements from all individual sensors, our algorithm is robust enough to various visual failure, LiDAR-degenerated scenarios, and is able to run in real time on an on-board computation platform, as shown by extensive experiments conducted in indoor, outdoor, and mixed environments of different scale (see attached video¹). Moreover, the results show that our proposed framework can improve the accuracy of state-of-the-art LiDAR-inertial or visual-inertial odometry. To share our findings and to make contributions to the community, we open source our codes on our Github².

I. INTRODUCTION

With the capacity of estimating ego-motion in six degrees of freedom (DOF) and simultaneously building dense and high precision maps of surrounding environments, LiDAR-based SLAM has been widely applied in the field of autonomous driving vehicles [1], drones [2, 3], and etc. With the development of LiDAR technologies, the emergence of low-cost LiDARs (e.g., Livox LiDAR [4]) makes LiDAR more accessible. Following this trend, a number of related works [5]–[8] draw the attention of the community to this field of research. However, LiDAR-based SLAM methods easily fail (i.e., degenerate) in those scenarios with few available geometry features, which is more critical for those LiDARs with small FoV [9]. In this work, to address the degeneration problems of LiDAR-based odometry, we propose a LiDAR-inertial-visual fusion framework to obtain the state estimation of higher robustness and accuracy. The main contributions of our work are:

- We take advantage of measurements from LiDAR, inertial and camera sensors and fuse them in a tight-coupled way. Experiments show that our method is robust enough in various challenging scenarios with aggressive motion, sensor failure, and even in narrow tunnel-like environments with a large number of moving objects and small LiDAR field of view.

J. Lin, C. Zheng, W. Xu and F. Zhang are with the Department of Mechanical Engineering, Hong Kong University, Hong Kong SAR., China.
{jiarong.lin, zhengcr, wuwei, fuzhang}@hku.hk

¹https://youtu.be/9lqRHmLN_MA

²<https://github.com/hku-mars/r2live>

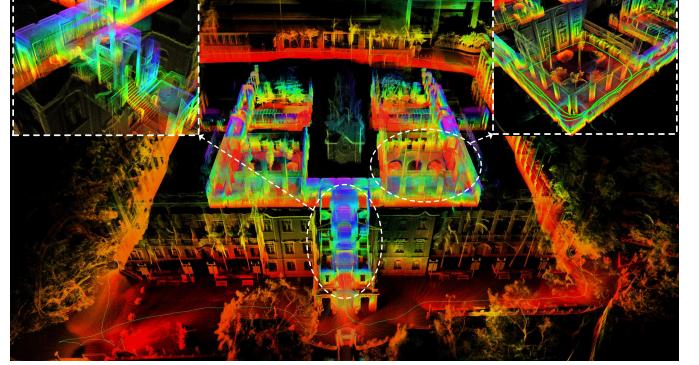


Fig. 1: We use our proposed method to reconstruct a high precision, large scale, indoor-outdoor, dense 3D maps of the main building of the University of Hong Kong (HKU). The green path is the computed trajectory and the 3D points are colored by height.

- We propose a framework with a high-rate filter-based odometry and a low-rate factor graph optimization. The filter-based odometry fuses the measurements of LiDAR, inertial, and camera sensors within an error-state iterated Kalman filter to achieve real-time performance. The factor graph optimization refines a local map of keyframe poses and visual landmark positions.
- By tightly fusing different types of sensors, we achieve high-accuracy state estimation. Experiment results show that our system is accurate enough to be used to reconstruct large-scale, indoor-outdoor dense 3D maps of building structures (see Fig. 1).

Our system is carefully engineered and open sourced¹ to benefit the whole robotics community.

II. RELATED WORK

In this section, we review existing works closely related to our work, including LiDAR-only odometry and mapping, LiDAR-Inertial fusion and LiDAR-Inertial-Visual methods.

A. LiDAR Odometry and Mapping

Zhang *et al* [10] first proposed a LiDAR odometry and mapping framework, LOAM, that combines ICP method [11] with point-to-plane and point-to-edge distance. It achieves good odometry and mapping performance by running the two modules at different rates. To make the algorithm run in real time at a computation limited platform, Shan *et al* [12] propose a lightweight and ground-optimized LOAM (LeGO-LOAM), which discards unreliable features in the step of ground plane segmentation. These works are mainly based on multi-line spinning LiDARs. Our previous works [9, 13] develop an accurate and robust algorithm by considering the low-level

physical properties of solid-state LiDARs with small FOV. However, these methods solely based on LiDAR measurements and are very vulnerable to featureless environments or other degenerated scenarios.

B. LiDAR-Inertial Odometry

The existing works of LiDAR-Inertial fusion can be categorized into two classes: loosely-coupled and the tightly-coupled. Loosely-coupled methods deal with two sensors separately to infer their motion constraints while tightly-coupled approaches directly fuse lidar and inertial measurements through joint-optimization. Compared with loosely-coupled methods, tightly-coupled methods show higher robustness and accuracy, therefore drawing increasing research interests recently. For example, authors in [14] propose LIOM which uses a graph optimization based on the priors from LiDAR-Inertial odometry and a rotation-constrained refinement method. Compared with the former algorithm, LIO-SAM [15] optimizes a sliding window of keyframe poses in a factor graph to achieve higher accuracy. Similarly, Li *et al.* propose LiLi-OM [16] for both conventional and solid-state LiDARs based on sliding window optimization. LINS [17] is the first tightly-coupled LIO that solves the 6 DOF ego-motion via iterated Kalman filtering. To lower the high computation load in calculating the Kalman gain, our previous work FAST-LIO [5] proposes a new formula of the Kalman gain computation, the resultant computation complexity depends on the state dimension instead of measurement dimension. The work achieves up to 50 Hz odometry and mapping rate while running on embedded computers onboard a UAV.

C. LiDAR-Inertial-Visual Odometry

On the basis of LiDAR-Inertial methods, LiDAR-Inertial-Visual odometry incorporating measurements from visual sensors shows higher robustness and accuracy. In the work of [18], the LiDAR measurements are used to provide depth information for camera images, forming a system similar to RGB-D camera that can leverage existing visual SLAM works such as ORB-SLAM2 [19]. This is a loosely-coupled method as it ignores the direct constraints on state imposed by LiDAR measurements. Zuo *et al* [20] propose a LIC-fusion framework combining IMU measurements, sparse visual features, and LiDAR plane and edge features with online spatial and temporal calibration based on the MSCKF framework, which is claimed more accurate and robust than state-of-the-art methods. In quick succession, their further work termed LIC-Fusion 2.0 [21] refines a novel plane-feature tracking algorithm across multiple LiDAR scans within a sliding-window to make LiDAR scan matching more robust.

To the best of our knowledge, our work is the first open sourced tightly-coupled LiDAR-inertial-visual fusion system. By fusing different types of sensor measurements, we achieve state estimation of higher accuracy and robustness. Extensive results show that our system is more accurate and robust than state-of-the-art LiDAR-inertial and Visual-inertial fusion estimator (e.g., FAST-LIO [5], and VINS-mono [22]).

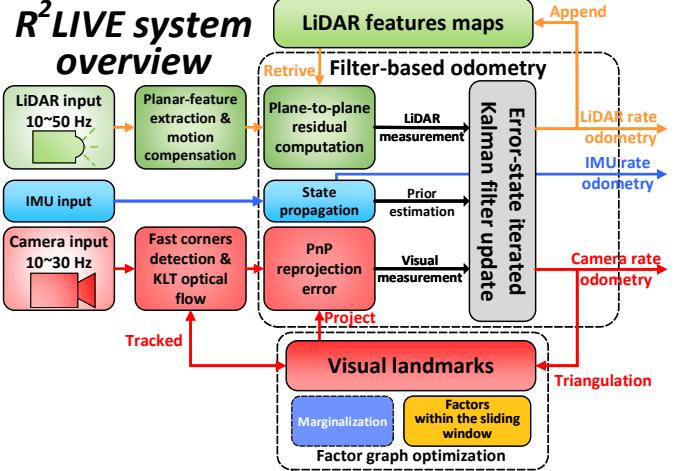


Fig. 2: The overview of our proposed method.

III. THE OVERVIEW OF OUR SYSTEM

The overview of our system is shown in Fig. 2, where the filter-based odometry taking advantage of the measurements from LiDAR, camera and inertial sensor, estimates the state within the framework of error-state iterated Kalman filter as detailed in Section IV. To further improve the accuracy of visual measurements, we leverage the factor graph optimization to refine the visual landmarks within a local sliding window as detailed in Section V.

IV. FILTER-BASED ODOMETRY

A. The boxplus “ \boxplus ” and boxminus “ \boxminus ” operator

In this paper, we make use of the “ \boxplus ” and “ \boxminus ” operations encapsulated on a manifold \mathcal{M} to simplify the notations and derivations. Let \mathcal{M} be the manifold on which the system state lies. Since a manifold is locally homeomorphic to \mathbb{R}^n , where n is the dimension of \mathcal{M} , we can use two operators, “ \boxplus ” and “ \boxminus ”, establishing a bijective map between the local neighborhood of \mathcal{M} and its tangent space \mathbb{R}^n [23]:

$$\boxplus : \mathcal{M} \times \mathbb{R}^n \rightarrow \mathcal{M}, \quad \boxminus : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{R}^n \quad (1)$$

For the compound manifold $\mathcal{M} = SO(3) \times \mathbb{R}^n$, we have:

$$\begin{bmatrix} \mathbf{R} \\ \mathbf{a}_1 \end{bmatrix} \boxplus \begin{bmatrix} \mathbf{r} \\ \mathbf{a}_2 \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{R} \cdot \text{Exp}(\mathbf{r}) \\ \mathbf{a}_1 + \mathbf{a}_2 \end{bmatrix}, \quad \begin{bmatrix} \mathbf{R}_1 \\ \mathbf{a}_1 \end{bmatrix} \boxminus \begin{bmatrix} \mathbf{R}_2 \\ \mathbf{a}_2 \end{bmatrix} \triangleq \begin{bmatrix} \text{Log}(\mathbf{R}_2^T \mathbf{R}_1) \\ \mathbf{a}_1 - \mathbf{a}_2 \end{bmatrix}$$

where $\mathbf{r} \in \mathbb{R}^3$, $\mathbf{a}_1, \mathbf{a}_2 \in \mathbb{R}^n$, $\text{Exp}(\cdot)$ and $\text{Log}(\cdot)$ denote the Rodrigues' transformation between the rotation matrix and rotation vector³.

B. Continuous-time kinematic model

In our current work, we assume that the time offsets among all the three sensors, LiDAR, camera, and IMU, are pre-calibrated. Furthermore, we assume the extrinsic between LiDAR and IMU are known as they are usually integrated and calibrated in factory, but estimate the camera IMU extrinsic online. Moreover, a LiDAR typically scans points sequentially, and points in a LiDAR frame could be measured at different body pose. This motion is compensated by IMU back propagation as shown in [5], hence points in a LiDAR frame are

³https://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula

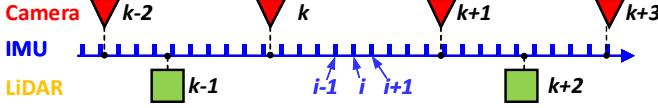


Fig. 3: Illustration of the input data sequences, where the frame rate of IMU, camera, and LiDAR is 200 Hz, 20 Hz and 10 Hz, respectively. The notation i denotes the index of IMU data while k denotes the index of LiDAR or camera measurements.

assumed to be measured at the same time. With these, the input data sequences of our system can be simplified into Fig. 3.

We assume that the IMU, LiDAR, and camera sensors are rigidly attached together with the extrinsic between LiDAR and IMU (LiDAR frame w.r.t. IMU frame) as ${}^I\mathbf{T}_L = ({}^I\mathbf{R}_L, {}^I\mathbf{p}_L)$, and the extrinsic between camera and IMU (camera frame w.r.t. IMU frame) is ${}^I\mathbf{T}_C = ({}^I\mathbf{R}_C, {}^I\mathbf{p}_C)$. For the sake of convenience, we take IMU as the body frame, which leads to the following continuous kinematic model:

$$\begin{aligned} {}^G\dot{\mathbf{p}}_I &= {}^G\mathbf{v}_I, {}^G\dot{\mathbf{v}}_I = {}^G\mathbf{R}_I(\mathbf{a}_m - \mathbf{b}_a - \mathbf{n}_a) + {}^G\mathbf{g}, \\ {}^G\dot{\mathbf{R}}_I &= {}^G\mathbf{R}_I [\boldsymbol{\omega}_m - \mathbf{b}_g - \mathbf{n}_g]_{\times}, \quad \mathbf{b}_g = \mathbf{n}_{bg}, \mathbf{b} = \mathbf{n}_{ba} \end{aligned} \quad (2)$$

where ${}^G(\cdot)$ denotes a vector represented in the global frame (i.e. the first gravitational aligned IMU frame [22]), ${}^G\mathbf{R}_I$ and ${}^G\mathbf{p}_I$ are the attitude and position of the IMU relative to the global frame, ${}^G\mathbf{g}$ is the gravitational acceleration⁴, $\boldsymbol{\omega}_m$ and \mathbf{a}_m are the raw gyroscope and accelerometer readings, \mathbf{n}_a and \mathbf{n}_g are the white noise of IMU measurement, \mathbf{b}_a and \mathbf{b}_g are the bias of gyroscope and accelerometer, which are modelled as random walk driven by Gaussian noise \mathbf{n}_{bg} and \mathbf{n}_{ba} .

C. Discrete IMU model

We discretize the continuous model (2) at the IMU rate. Let \mathbf{x}_i be the state vector at the i -th IMU measurement:

$$\mathbf{x}_i = [{}^G\mathbf{R}_{I_i}^T \quad {}^G\mathbf{p}_{I_i}^T \quad {}^I\mathbf{R}_{C_i}^T \quad {}^I\mathbf{p}_{C_i}^T \quad {}^G\mathbf{v}_i^T \quad \mathbf{b}_{\mathbf{g}_i}^T \quad \mathbf{b}_{\mathbf{a}_i}^T]^T$$

Discretizing (2) by zero-order holder (i.e., IMU measurements over one sampling time period Δt are constant), we obtain

$$\mathbf{x}_{i+1} = \mathbf{x}_i \boxplus (\Delta t \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i)) \quad (3)$$

where

$$\begin{aligned} \mathbf{u}_i &= [\boldsymbol{\omega}_{m_i}^T \quad \mathbf{a}_{m_i}^T]^T, \quad \mathbf{w}_i = [\mathbf{n}_{\mathbf{g}_i}^T \quad \mathbf{n}_{\mathbf{a}_i}^T \quad \mathbf{n}_{\mathbf{b}_{\mathbf{g}_i}}^T \quad \mathbf{n}_{\mathbf{b}_{\mathbf{a}_i}}^T]^T \\ \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i) &= \begin{bmatrix} \boldsymbol{\omega}_{m_i} - \mathbf{b}_{\mathbf{g}_i} - \mathbf{n}_{\mathbf{g}_i} \\ {}^G\mathbf{v}_i \\ \mathbf{0}_{3 \times 1} \\ \mathbf{0}_{3 \times 1} \\ {}^G\mathbf{R}_{I_i}(\mathbf{a}_{m_i} - \mathbf{b}_{\mathbf{a}_i} - \mathbf{n}_{\mathbf{a}_i}) + {}^G\mathbf{g} \\ \mathbf{n}_{\mathbf{b}_{\mathbf{g}_i}} \\ \mathbf{n}_{\mathbf{b}_{\mathbf{a}_i}} \end{bmatrix} \end{aligned}$$

D. Propagation

In our work, we leverage an on-manifold iterated error state Kalman filter [24] to estimate the state vector \mathbf{x}_i , in which the state estimation error $\delta\hat{\mathbf{x}}_i$ is characterized in the tangent space of the state estimate $\hat{\mathbf{x}}_i$:

⁴ ${}^G\mathbf{g} = [0, 0, 9.805]$ at Hong Kong SAR, China.

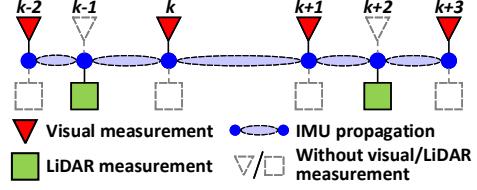


Fig. 4: The illustration of the update of our error-state iterated Kalman filter.

$$\begin{aligned} \delta\hat{\mathbf{x}}_i &\triangleq \mathbf{x}_i \boxminus \hat{\mathbf{x}}_i \\ &= \left[{}^G\delta\hat{\mathbf{r}}_{I_i}^T \quad {}^G\delta\hat{\mathbf{p}}_{I_i}^T \quad {}^I\delta\hat{\mathbf{r}}_{C_i}^T \quad {}^I\delta\hat{\mathbf{p}}_{C_i}^T \quad {}^G\delta\hat{\mathbf{v}}_i^T \quad \delta\hat{\mathbf{b}}_{\mathbf{g}_i}^T \quad \delta\hat{\mathbf{b}}_{\mathbf{a}_i}^T \right]^T \\ &\sim \mathcal{N}(\mathbf{0}_{21 \times 1}, \Sigma_{\delta\hat{\mathbf{x}}_i}) \end{aligned} \quad (4)$$

Note that $\delta\hat{\mathbf{x}}_i \in \mathbb{R}^{21}$ is in minimum dimension (the system dimension 21) and is a random vector with covariance $\Sigma_{\delta\hat{\mathbf{x}}_i}$. ${}^G\delta\hat{\mathbf{r}}_{I_i}^T$ and ${}^I\delta\hat{\mathbf{r}}_{C_i}^T$ are:

$${}^G\delta\hat{\mathbf{r}}_{I_i} = \text{Log}({}^G\hat{\mathbf{R}}_{I_i}^T {}^G\mathbf{R}_{I_i}), \quad {}^I\delta\hat{\mathbf{r}}_{C_i} = \text{Log}({}^I\hat{\mathbf{R}}_{C_i}^T {}^I\mathbf{R}_{C_i})$$

Once receiving a new IMU measurement, the state estimate is propagated by setting the process noise in (3) to zero:

$$\hat{\mathbf{x}}_{i+1} = \hat{\mathbf{x}}_i \boxplus (\Delta t \cdot \mathbf{f}(\hat{\mathbf{x}}_i, \mathbf{u}_i, \mathbf{0})). \quad (5)$$

The associated estimation error is propagated in the linearized error space as follows (see [24] for more details):

$$\begin{aligned} \delta\hat{\mathbf{x}}_{i+1} &= \mathbf{x}_{i+1} \boxminus \hat{\mathbf{x}}_{i+1} \\ &= (\mathbf{x}_i \boxplus (\Delta t \cdot \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i))) \boxminus (\hat{\mathbf{x}}_i \boxplus (\Delta t \cdot \mathbf{f}(\hat{\mathbf{x}}_i, \mathbf{u}_i, \mathbf{0}))) \\ &\approx \mathbf{F}_{\delta\hat{\mathbf{x}}} \delta\hat{\mathbf{x}} + \mathbf{F}_{\mathbf{w}} \mathbf{w}_i \\ &\sim \mathcal{N}(\mathbf{0}_{21 \times 1}, \Sigma_{\delta\hat{\mathbf{x}}_{i+1}}) \end{aligned} \quad (6)$$

where:

$$\begin{aligned} \Sigma_{\delta\hat{\mathbf{x}}_{i+1}} &= \mathbf{F}_{\delta\hat{\mathbf{x}}} \Sigma_{\delta\hat{\mathbf{x}}} \mathbf{F}_{\delta\hat{\mathbf{x}}}^T + \mathbf{F}_{\mathbf{w}} \mathbf{Q} \mathbf{F}_{\mathbf{w}}^T \\ \mathbf{F}_{\delta\hat{\mathbf{x}}} &= \frac{\partial (\delta\hat{\mathbf{x}}_{i+1})}{\partial \delta\hat{\mathbf{x}}_i} \Big|_{\delta\hat{\mathbf{x}}_i=0, \mathbf{w}_i=0}, \quad \mathbf{F}_{\mathbf{w}} = \frac{\partial (\delta\hat{\mathbf{x}}_{i+1})}{\partial \mathbf{w}_i} \Big|_{\delta\hat{\mathbf{x}}_i=0, \mathbf{w}_i=0} \end{aligned} \quad (7)$$

with \mathbf{Q} is the diagonal covariance matrix of noise, and the exact values of $\mathbf{F}_{\delta\hat{\mathbf{x}}}$ and $\mathbf{F}_{\mathbf{w}}$ are computed in Appendix. A.

The two propagations in (5) and (7) start from the optimal state and covariance estimate after fusing the most recent LiDAR/camera measurement (e.g., the k -th measurement, see Section IV-I), and repeat until receiving the next LiDAR/camera measurement (e.g., the $(k+1)$ -th measurement). The relation between time index i and k is shown in Fig. 3.

E. The prior distribution

Let the two propagations in (5) and (7) stop at the $(k+1)$ -th LiDAR/camera measurement (see Fig. 4), and the propagated state estimate and covariance are $\hat{\mathbf{x}}_{k+1}$ and $\Sigma_{\delta\hat{\mathbf{x}}_{k+1}}$, respectively. They essentially impose a prior distribution for the state \mathbf{x}_{k+1} before fusing the $(k+1)$ -th measurement as below:

$$\mathbf{x}_{k+1} \boxminus \hat{\mathbf{x}}_{k+1} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\delta\hat{\mathbf{x}}_{k+1}}). \quad (8)$$

F. Initialization of iterated update

The prior distribution in (8) will be fused with the LiDAR or camera measurements to produce a maximum a-posterior (MAP) estimate (denoted as $\check{\mathbf{x}}_{k+1}$) of \mathbf{x}_{k+1} . The MAP estimate $\check{\mathbf{x}}_{k+1}$ is initialized as the prior estimate $\hat{\mathbf{x}}_{k+1}$ and is refined iteratively due to the nonlinear nature of the problem.

In each iteration, the error $\delta\check{\mathbf{x}}_{k+1}$ between the true state \mathbf{x}_{k+1} and the current estimate $\check{\mathbf{x}}_{k+1}$, defined as

$$\delta\check{\mathbf{x}}_{k+1} \triangleq \mathbf{x}_{k+1} \boxminus \check{\mathbf{x}}_{k+1}, \quad (9)$$

will be solved by minimizing the posterior distribution considering the prior in (8) and LiDAR/visual measurements. Therefore, the prior distribution in terms of \mathbf{x}_{k+1} represented by (8) should be transformed to an equivalent prior distribution in terms of $\delta\check{\mathbf{x}}_{k+1}$:

$$\begin{aligned} \mathbf{x}_{k+1} \boxminus \hat{\mathbf{x}}_{k+1} &= (\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}) \boxminus \hat{\mathbf{x}}_{k+1} \\ &\approx \check{\mathbf{x}}_{k+1} \boxminus \hat{\mathbf{x}}_{k+1} + \mathcal{H}\delta\check{\mathbf{x}}_{k+1} \\ &\sim \mathcal{N}(\mathbf{0}, \Sigma_{\delta\check{\mathbf{x}}_{k+1}}), \end{aligned} \quad (10)$$

where

$$\mathcal{H} = \frac{(\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}) \boxminus \hat{\mathbf{x}}_{k+1}}{\partial \delta\check{\mathbf{x}}_{k+1}} \Big|_{\delta\check{\mathbf{x}}_{k+1}=0}$$

is computed in detail in Appendix B, (10) essentially imposes a prior distribution to $\delta\check{\mathbf{x}}_{k+1}$ as below:

$$\delta\check{\mathbf{x}}_{k+1} \sim \mathcal{N}(-\mathcal{H}^{-1}(\check{\mathbf{x}}_{k+1} \boxminus \hat{\mathbf{x}}_{k+1}), \mathcal{H}^{-1}\Sigma_{\delta\check{\mathbf{x}}_{k+1}}\mathcal{H}^{-T}) \quad (11)$$

G. LiDAR measurement

If the $(k+1)$ -th measurement is a LiDAR frame, we extract planar feature points from the raw 3D points as in [5] and compensate the in-frame motion as in Section IV-B. Denote \mathcal{L}_{k+1} the set of feature points after motion compensation, we compute the residual of each feature point ${}^L\mathbf{p}_j \in \mathcal{L}_{k+1}$ where j is the index of feature point and the superscript L denotes that the point is represented in the LiDAR-reference frame.

With $\check{\mathbf{x}}_{k+1}$ being the current estimate of \mathbf{x}_{k+1} , we can transform ${}^L\mathbf{p}_j$ from LiDAR frame to the global frame ${}^G\mathbf{p}_j = {}^G\check{\mathbf{R}}_{I_{k+1}}({}^I\mathbf{R}_L {}^L\mathbf{p}_j + {}^I\mathbf{p}_L) + {}^G\check{\mathbf{p}}_{I_{k+1}}$. As the previous LOAM pipeline does in [5, 9], we search for the nearest planar feature points in the map and use them to fit a plane with normal \mathbf{u}_j and an in-plane point \mathbf{q}_j , the measurement residual is:

$$\mathbf{r}_l(\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_j) = \mathbf{u}_j^T ({}^G\mathbf{p}_j - \mathbf{q}_j) \quad (12)$$

Let \mathbf{n}_j be the measurement noise of the point ${}^L\mathbf{p}_j$, we can obtain the true point location ${}^L\mathbf{p}_j^{\text{gt}}$ by compensating the noise from ${}^L\mathbf{p}_j$:

$${}^L\mathbf{p}_j = {}^L\mathbf{p}_j^{\text{gt}} + \mathbf{n}_j, \mathbf{n}_j \sim \mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{n}_j}). \quad (13)$$

This true point location together with the true state \mathbf{x}_{k+1} should lead to zero residual in (12), i.e.,

$$\mathbf{0} = \mathbf{r}_l(\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_j^{\text{gt}}) \approx \mathbf{r}_l(\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_j) + \mathbf{H}_j^l \delta\check{\mathbf{x}}_{k+1} + \boldsymbol{\alpha}_j, \quad (14)$$

which constitutes a posteriori distribution for $\delta\check{\mathbf{x}}_{k+1}$. In (14), \mathbf{x}_{k+1} is parameterized by its error $\delta\check{\mathbf{x}}_{k+1}$ defined in (9) and $\boldsymbol{\alpha}_j \sim \mathcal{N}(\mathbf{0}, \Sigma_{\boldsymbol{\alpha}_j})$:

$$\begin{aligned} \mathbf{H}_j^l &= \frac{\partial \mathbf{r}_l(\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_j)}{\partial \delta\check{\mathbf{x}}_{k+1}} \Big|_{\delta\check{\mathbf{x}}_{k+1}=0} \\ \boldsymbol{\Sigma}_{\boldsymbol{\alpha}_j} &= \mathbf{F}_{\mathbf{p}_j} \boldsymbol{\Sigma}_{\mathbf{n}_j} \mathbf{F}_{\mathbf{p}_j}^T \\ \mathbf{F}_{\mathbf{p}_j} &= \left(\frac{\partial \mathbf{r}_l(\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_j)}{\partial {}^L\mathbf{p}_j} \right) = {}^G\check{\mathbf{R}}_{I_{k+1}} {}^I\mathbf{R}_L \end{aligned} \quad (15)$$

The detailed computation of \mathbf{H}_j^l can be found in Appendix C.

H. Visual measurement

If the $(k+1)$ -th frame is a camera image, we extract the FAST corner feature points \mathcal{C}_{k+1} from the undistorted image and use KLT optical flow to track feature points in \mathcal{C}_{k+1} seen by keyframes in the current sliding window (Section V). If a feature point in \mathcal{C}_{k+1} is lost or has not been yet tracked before, we triangulate the new feature point in 3D space (visual landmarks) with the optimal estimated camera poses.

The reprojection errors between visual landmarks and its tracked feature points in the $(k+1)$ -th frame are used for updating the current state estimate $\check{\mathbf{x}}_{k+1}$. For an extracted corner point ${}^C\mathbf{p}_s = [u_s \ v_s]^T \in \mathcal{C}_{k+1}$ where s is the index of corner point, its correspondence landmark in 3D space is denoted as ${}^G\mathbf{P}_s$, then the measurement residual of ${}^C\mathbf{p}_s$ is:

$$\begin{aligned} {}^C\mathbf{P}_s &= ({}^G\check{\mathbf{R}}_{I_{k+1}} {}^I\check{\mathbf{R}}_{C_{k+1}})^T {}^G\mathbf{P}_s - {}^I\check{\mathbf{R}}_{C_{k+1}}^T {}^I\check{\mathbf{p}}_{C_{k+1}} \\ &\quad - ({}^G\check{\mathbf{R}}_{I_{k+1}} {}^I\check{\mathbf{R}}_{C_{k+1}})^T {}^G\check{\mathbf{p}}_{I_{k+1}} \\ \mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_s, {}^G\mathbf{P}_s) &= {}^C\mathbf{p}_s - \boldsymbol{\pi}({}^G\mathbf{P}_s) \end{aligned} \quad (16)$$

where $\boldsymbol{\pi}(\cdot)$ is the pin-hole projection model.

Now considering the measurement noise, we have:

$${}^G\mathbf{P}_s = {}^G\mathbf{P}_s^{\text{gt}} + \mathbf{n}_{\mathbf{P}_s}, \mathbf{n}_{\mathbf{P}_s} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{n}_{\mathbf{P}_s}}) \quad (17)$$

$${}^C\mathbf{p}_s = {}^C\mathbf{p}_s^{\text{gt}} + \mathbf{n}_{\mathbf{p}_s}, \mathbf{n}_{\mathbf{p}_s} \sim \mathcal{N}(\mathbf{0}, \Sigma_{\mathbf{n}_{\mathbf{p}_s}}) \quad (18)$$

where ${}^G\mathbf{P}_s^{\text{gt}}$ and ${}^C\mathbf{p}_s^{\text{gt}}$ are the true value of ${}^G\mathbf{P}_s$ and ${}^C\mathbf{p}_s$, respectively. With these, we obtain the first order Taylor expansion of the true zero residual $\mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_s^{\text{gt}})$ as:

$$\begin{aligned} \mathbf{0} &= \mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_s^{\text{gt}}, {}^G\mathbf{P}_s^{\text{gt}}) \\ &\approx \mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_s, {}^G\mathbf{P}_s) + \mathbf{H}_s^c \delta\check{\mathbf{x}}_{k+1} + \boldsymbol{\beta}_s, \end{aligned} \quad (19)$$

which constitutes another posteriori distribution for $\delta\check{\mathbf{x}}_{k+1}$. In (19), $\boldsymbol{\beta}_s \sim \mathcal{N}(\mathbf{0}, \Sigma_{\boldsymbol{\beta}_s})$ and:

$$\begin{aligned} \mathbf{H}_s^c &= \frac{\partial \mathbf{r}_c(\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_s, {}^G\mathbf{P}_s)}{\partial \delta\check{\mathbf{x}}_{k+1}} \Big|_{\delta\check{\mathbf{x}}_{k+1}=0} \\ \boldsymbol{\Sigma}_{\boldsymbol{\beta}_s} &= \boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{p}_s}} + \mathbf{F}_{\mathbf{p}_s} \boldsymbol{\Sigma}_{\mathbf{n}_{\mathbf{p}_s}} \mathbf{F}_{\mathbf{p}_s}^T \\ \mathbf{F}_{\mathbf{p}_s} &= \frac{\partial \mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_s, {}^G\mathbf{P}_s)}{\partial {}^G\mathbf{P}_s} \end{aligned} \quad (20)$$

The detailed computation of \mathbf{H}_s^c and $\mathbf{F}_{\mathbf{p}_s}$ is given in appendix D

I. Update of error-state iterated Kalman filter

Combining the prior distribution (11), the posterior distribution due to LiDAR measurement (14) and the posterior distribution due to visual measurement (19), we obtain the maximum a posterior (MAP) estimation of $\delta\check{\mathbf{x}}_{k+1}$:

$$\begin{aligned} \min_{\delta\check{\mathbf{x}}_{k+1}} & \left(\|\check{\mathbf{x}}_{k+1} \boxminus \hat{\mathbf{x}}_{k+1} + \mathcal{H}\delta\check{\mathbf{x}}_{k+1}\|_{\Sigma_{\delta\check{\mathbf{x}}_{k+1}}}^2 \right. \\ & + \sum_{j=1}^{m_l} \left\| \mathbf{r}_l(\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_j) + \mathbf{H}_j^l \delta\check{\mathbf{x}}_{k+1} \right\|_{\Sigma_{\boldsymbol{\alpha}_j}}^2 \\ & \left. + \sum_{s=1}^{m_c} \left\| \mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_s, {}^G\mathbf{P}_s) + \mathbf{H}_s^c \delta\check{\mathbf{x}}_{k+1} \right\|_{\Sigma_{\boldsymbol{\beta}_s}}^2 \right) \end{aligned}$$

where $\|\mathbf{x}\|_{\Sigma}^2 = \mathbf{x}^T \Sigma^{-1} \mathbf{x}$ is the squared Mahalanobis distance with covariance Σ . Notice that the measurements of LiDAR

and camera may not appear at the same time instant (see Fig. 4), therefore m_l (or m_c) could be zero in the above optimization. Denote

$$\begin{aligned} \mathbf{H}^T &= [\mathbf{H}_1^l, \dots, \mathbf{H}_{m_l}^l, \mathbf{H}_1^c, \dots, \mathbf{H}_{m_c}^c]^T \\ \mathbf{R} &= \text{diag}(\Sigma_{\alpha_1}, \dots, \Sigma_{\alpha_{m_l}}, \Sigma_{\beta_1}, \dots, \Sigma_{\beta_{m_c}}) \\ \check{\mathbf{z}}_{k+1}^T &= \left[\mathbf{r}_l(\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_1), \dots, \mathbf{r}_l(\check{\mathbf{x}}_{k+1}, {}^L\mathbf{p}_{m_l}), \right. \\ &\quad \left. \mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_1), \dots, \mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C\mathbf{p}_{m_c}) \right] \\ \mathbf{P} &= (\mathcal{H})^{-1} \Sigma_{\delta\check{\mathbf{x}}_{k+1}} (\mathcal{H})^{-T} \end{aligned} \quad (21)$$

Following [5], we have the Kalman gain computed as:

$$\mathbf{K} = (\mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} + \mathbf{P}^{-1})^{-1} \mathbf{H}^T \mathbf{R}^{-1} \quad (22)$$

Then we can update the state estimate as:

$$\check{\mathbf{x}}_{k+1} = \check{\mathbf{x}}_{k+1} \boxplus (-\mathbf{K}\check{\mathbf{z}}_{k+1} - (\mathbf{I} - \mathbf{K}\mathbf{H})(\mathcal{H})^{-1}(\check{\mathbf{x}}_{k+1} \boxminus \hat{\mathbf{x}}_{k+1}))$$

The above process (Section IV-G to Section IV-I) is iterated until convergence (i.e., the update is smaller than a given threshold). The converged state estimate is then used to (1) project points in the new LiDAR frame to the world frame and append them to the existing point cloud map; (2) triangulate new visual landmarks of the current frame if it is a keyframe; (3) serve as the starting point of the propagation in Section IV-D for the next cycle:

$$\hat{\mathbf{x}}_{k+1} = \check{\mathbf{x}}_{k+1}, \quad \hat{\Sigma}_{\delta\check{\mathbf{x}}_{k+1}} = (\mathbf{I} - \mathbf{K}\mathbf{H}) \check{\Sigma}_{\delta\check{\mathbf{x}}_{k+1}}$$

V. FACTOR GRAPH OPTIMIZATION

As mentioned in Section IV-I, untracked visual landmarks in the newly added keyframe are triangulated to create new visual landmarks. This triangulation is usually of low precision due to keyframe pose estimation error. To further improve the quality of visual landmarks, keyframe poses, and simultaneously calibrate the time offset between the camera and LiDAR-IMU subsystem, we leverage a factor graph optimization for optimizing the camera-poses and the visual landmarks within a sliding window of image keyframes.

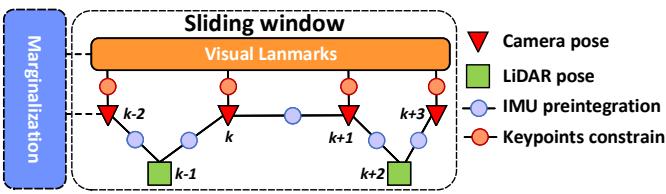


Fig. 5: Our factor graph optimization.

Our factor graph optimization is similar to VINS-Mono [22], but further incorporates pose constraints due to LiDAR measurements as shown in Fig. 5. Constraints due to IMU preintegration are also included to connect the LiDAR factor with camera factor. To keep the back-end optimization lightweight, the LiDAR poses in the pose graph are fixed and the LiDAR raw point measurements are not engaged in the pose graph optimization.

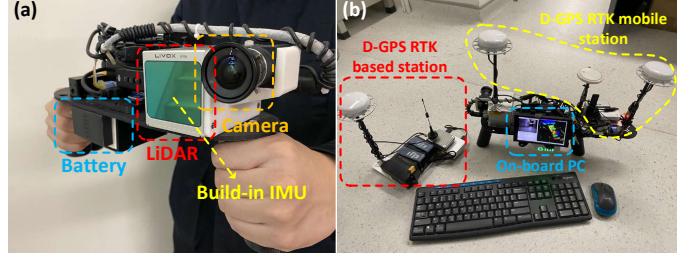


Fig. 6: Our handheld device for data collection, (a) shows our minimum system, with a total weight 2.09 Kg; (b) an additional D-GPS RTK system is used to evaluate the precision.

VI. EXPERIMENTS AND RESULTS

A. Our device for data sampling

Our handheld device for data collection is shown in Fig. 6 (a), which includes the power supply unit, the onboard DJI *manifold-2c*⁵ computation platform (equipped with an *Intel i7-8550u* CPU and 8 GB RAM), a global shutter camera, and a *LiVOX AVIA*⁶ LiDAR. The FoV of the camera is $82.9^\circ \times 66.5^\circ$ while the FoV of LiDAR is $70.4^\circ \times 77.2^\circ$. To quantitatively evaluate the precision of our algorithm (Section VI-E), we install a Differential-GPS (D-GPS) real-time kinematic (RTK) system⁷ on our device, shown in Fig. 6 (b).

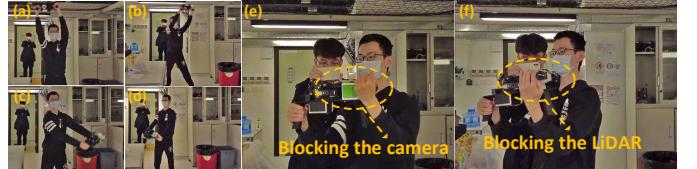


Fig. 7: We evaluate the robustness of our algorithm under aggressive motion (sequence in a-d) and sensor failure by intentionally blocking the camera (e) and LiDAR sensor (f).

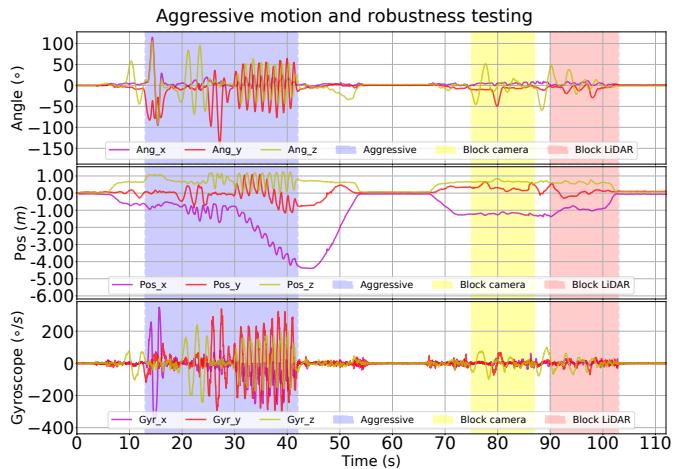


Fig. 8: Our estimated pose and the raw gyroscope reading of Experiment-1. The shaded area in blue, yellow and red represent the phase of aggressive motion, camera-failure and LiDAR-failure, respectively.

⁵<https://www.dji.com/manifold-2c>

⁶<https://www.livoxtech.com/avia>

⁷<https://www.dji.com/d-rtk>

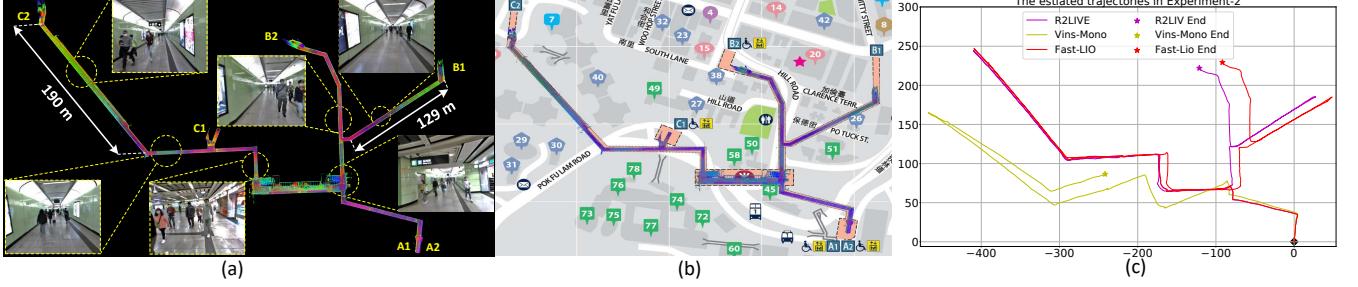


Fig. 9: We evaluate our algorithm in a Hong Kong MTR station consisting of cluttered lobby and very long narrow tunnels, as shown in (a). The tunnel is up to 190 meters long and is filled with moving pedestrians, making it extremely challenging for both LiDAR-based and camera-based SLAM methods. (b): the map built by our system is well aligned with the street map of the MTR station. (c) Trajectory comparison among our system “R2LIVE”, the LiDAR-inertial system “Fast-LIO” , and visual-inertial system “VINS-Mono”. The starting point is marked with \blacklozenge while the ending point of each trajectory is marked with \star . “VINS-Mono” stopped at middle way due to the failure of feature tracking.

B. Experiment-1: Robustness evaluation with aggressive motion and sensor failure

In this experiment, we evaluate the robustness of our algorithm in aggressive motion (see Fig. 7 (a~d)), where the maximum angular speed reaches up to $300^\circ/s$ (see the gyroscope readings in Fig. 8). In addition, we also simulate drastic sensor failures by intentionally blocking the camera (see Fig. 7(e)) and LiDAR (see Fig. 7(f)).

The results of our estimated attitude and position are shown in Fig. 8, where we shade the different testing phases with different colors. It is seen that our estimated trajectory can tightly track the actual motion even in severe rotation and translation. Even when the camera or LiDAR occasionally failed, the estimated trajectory is still very smooth and exhibits no noticeable degradation. These results show that our algorithm is robust to endure aggressive motion and sensor failures. We refer readers to the accompanying video on YouTube⁸ showing details of the experiment.

C. Experiment-2: Robustness evaluation in a narrow tunnel-like environments with moving objects

In this experiment, we challenge one of the most difficult scenarios in the scope of camera-based and LiDAR-based SLAM, a MTR station, the *HKU station*⁹. It is a typical narrow tunnel-like environment (see the left figure of Fig. 9 (a)), with the maximum length reaching 190 meters. Moreover, there are many moving pedestrians frequently showing in the sensor views. All these factors make the SLAM extremely challenging: 1) the long tunnel structure significantly reduces the geometric features for LiDAR-based SLAM, especially when walking along the tunnel or turning the LiDAR around at one end of the tunnel; 2) The highly repeated visual feature (pattern on the floor) easily fail the matching and tracking of visual features in visual SLAM; 3) The many moving pedestrians could cause outliers to the already few point- and visual-features.

Despite these challenges, our system is robust enough to survive in this scene. In Fig. 9 (b), we align our point cloud data with the *HKU station street map*¹⁰, and find them match tightly. This demonstrates that our localization is of high

robustness and accuracy. Moreover, we plot our estimated trajectory together with that of Vins-Mono¹¹ and Fast-LIO¹² in Fig. 9 (c), where we can see that our method achieves the best overall performance in this experiment.

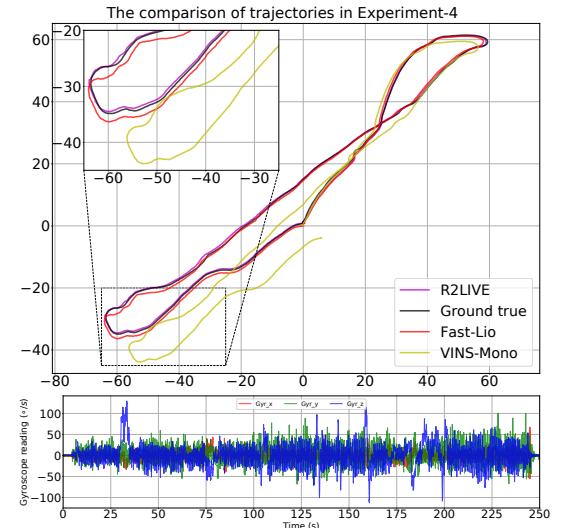


Fig. 10: The upper figure plots the different trajectories in experiments-4, while the lower figure shows the raw gyroscope reading.

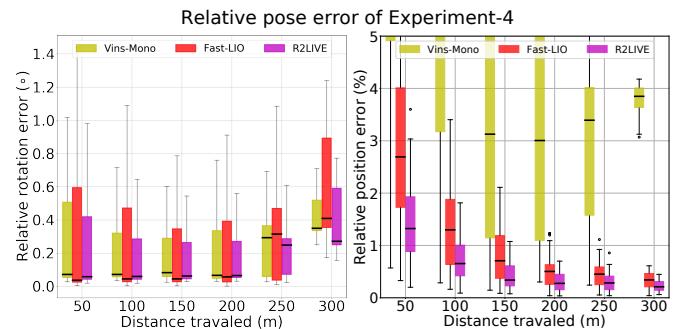


Fig. 11: The relative pose error in experiment-4. At the 300 meters traveled distance, the median of pose error of “Vins-Mono”, “Fast-LIO” and “R2LIVE” are ($0.35^\circ, 3.84\%$), ($0.41^\circ, 0.34\%$) and ($0.27^\circ, 0.21\%$) respectively.

⁸https://youtu.be/9lqRHmlN_MA

⁹https://en.wikipedia.org/wiki/HKU_station

¹⁰<https://www.mtr.com.hk/archive/en/services/maps/hku.pdf>

¹¹<https://github.com/HKUST-Aerial-Robotics/VINS-Mono>

¹²https://github.com/hku-mars/FAST_LIO/

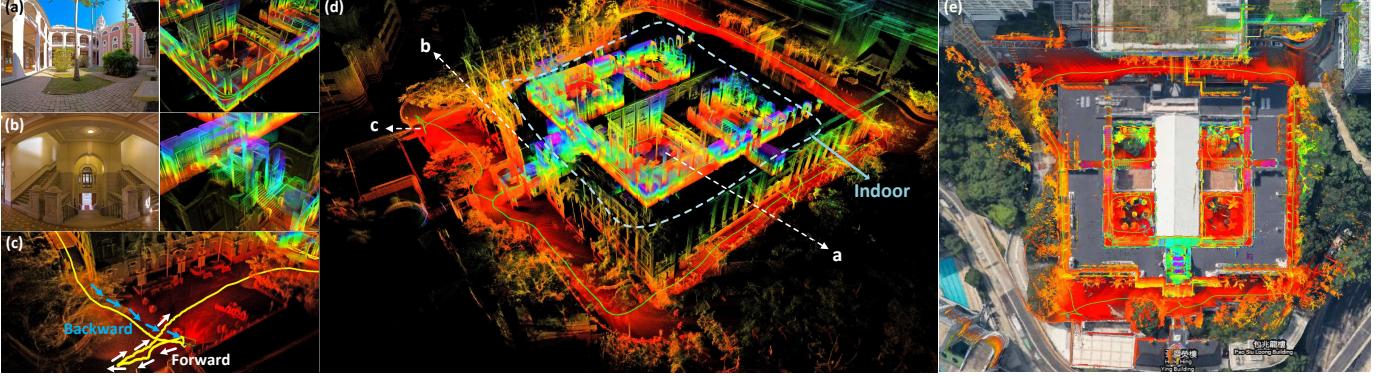


Fig. 12: The reconstructed 3D maps in Experiment-3 are shown in (d), and the detail point cloud with the correspondence panorama images are shown in (a) and (b). (c) shows that our algorithm can close the loop by itself (returning the starting point) without any additional processing (e.g. loop closure). In (e), we merge our map with the satellite image to further examine the accuracy of our system.

D. Experiment-3: High precision mapping large-scale indoor & outdoor urban environment

In this experiment, we show that our proposed method is able to reconstruct a dense 3D, high precision, large-scale indoor-outdoor map of urban environments. We collect data of the *HKU main building* exterior and interior. The real-time reconstructed 3D maps is shown in Fig. 12, in which the clear and high-quality point cloud demonstrates that our proposed method is of high accuracy. What worth to be mentioned is, without any additional processing (i.e., loop closure), our algorithm can still close the loop by itself (see Fig. 12 (c)) after traversing 876 meters, which demonstrates that our proposed method is extremely low drift. Finally, we overlay our map on the satellite image and find them match tightly (see Fig. 12 (e)).

E. Experiment-4: Quantitative evaluation of precision using D-GPS RTK

In this experiment, we quantitatively evaluate the precision of Vins-Mono (IMU+Camera), Fast-LIO (IMU+LiDAR) and our algorithm by comparing their estimated trajectory with the ground truth trajectory (see the upper figure of Fig. 10) obtained from a real-time Differential-GPS (D-GPS) Kinematic (RTK). The data has the maximum angular velocity reaching $130^\circ/s$ (see the gyroscope reading in Fig. 10). We calculate translational and rotational errors for all possible subsequences of length (50,100,150,...,300) meters, with the relative pose error (RPE) among these methods shown in Fig. 11.

F. Run time analysis

The average time consumption in experiment 1~4 are listed in TABLE I, which demonstrates that R²LIVE can achieve real-time localization on both the desktop PC and embedded computation platform. Notice that the factor graph optimization runs on a separate thread and therefore is allowed to run at a lower rate.

VII. CONCLUSION

In this letter, we propose a robust, real-time LiDAR-inertial-visual fusion framework based on a high-rate filter-based odometry and factor graph optimization. We fuse the

PC/on-board	Exp-1 (ms)	Exp-2 (ms)	Exp-3 (ms)	Exp-4 (ms)
LI-Odom	8.81 / 15.98	11.54 / 25.30	14.91 / 30.64	10.92 / 24.37
VI-Odom	7.84 / 13.92	8.84 / 19.10	9.57 / 19.56	8.99 / 20.16
FG-OPM	26.10 / 45.35	30.20 / 65.25	29.25 / 58.08	27.98 / 60.43

TABLE I: The average running time of R²LIVE in Experiment-1~4 (Exp-1~4) on desktop PC (with Intel i7-9700K CPU and 32GB RAM) and on-board computer (with Intel i7-8550u CPU and 8GB RAM). The items "LI-Odom", "VI-Odom" and "FG-OPM" represent the average time for LiDAR-inertial fusion, Visual-Inertial fusion, and factor graph optimization, respectively, in our system.

measurements of LiDAR, inertial, and camera sensors within an error-state iterated Kalman filter and use the factor graph optimization to refine a sliding window of image keyframes and visual landmarks. Our system was tested in large-scale, indoor-outdoor, and narrow tunnel-like environments, challenging sensor failure and aggressive motion. In all the tests, our method achieves a high level of accuracy and robustness in localization and mapping.

REFERENCES

- [1] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2011, pp. 163–168.
- [2] A. Bry, A. Bachrach, and N. Roy, "State estimation for aggressive flight in gps-denied environments using onboard sensing," in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 1–8.
- [3] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.
- [4] Z. Liu, F. Zhang, and X. Hong, "Low-cost retina-like robotic lidars based on incommensurable scanning," *IEEE Transactions on Mechatronics*, 2021, in press. [Online]. Available: <https://arxiv.org/pdf/2006.11034.pdf>
- [5] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, 2021, in press. [Online]. Available: <https://arxiv.org/pdf/2010.08196.pdf>
- [6] J. Lin, X. Liu, and F. Zhang, "A decentralized framework for simultaneous calibration, localization and mapping with multiple lidars," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4870–4877.
- [7] Z. Liu and F. Zhang, "Balm: Bundle adjustment for lidar mapping," *IEEE Robotics and Automation Letters*, 2021, in press. [Online]. Available: <https://arxiv.org/pdf/2010.08215.pdf>
- [8] X. Liu and F. Zhang, "Extrinsic calibration of multiple lidars of small fov in targetless environments," *IEEE Robotics and Automation Letters*, 2021, in press.

- [9] J. Lin and F. Zhang, “Loam livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3126–3131.
- [10] J. Zhang and S. Singh, “Loam: Lidar odometry and mapping in real-time,” in *Robotics: Science and Systems*, vol. 2, no. 9, 2014.
- [11] K.-L. Low, “Linear least-squares optimization for point-to-plane icp surface registration,” *Chapel Hill, University of North Carolina*, vol. 4, no. 10, pp. 1–3, 2004.
- [12] T. Shan and B. Englöt, “Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4758–4765.
- [13] J. Lin and F. Zhang, “A fast, complete, point cloud based loop closure for lidar odometry and mapping,” *arXiv preprint arXiv:1909.11811*, 2019.
- [14] H. Ye, Y. Chen, and M. Liu, “Tightly coupled 3d lidar inertial odometry and mapping,” in *2019 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.
- [15] T. Shan, B. Englöt, D. Meyers, W. Wang, C. Ratti, and D. Rus, “Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping,” *arXiv preprint arXiv:2007.00258*, 2020.
- [16] K. Li, M. Li, and U. D. Hanebeck, “Towards high-performance solid-state-lidar-inertial odometry and mapping,” *arXiv preprint arXiv:2010.13150*, 2020.
- [17] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, “Lins: A lidar-inertial state estimator for robust and efficient navigation,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 8899–8906.
- [18] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, “Camvox: A low-cost and accurate lidar-assisted visual slam system,” *arXiv preprint arXiv:2011.11357*, 2020.
- [19] R. Mur-Artal and J. D. Tardós, “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,” *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.
- [20] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, “Lic-fusion: Lidar-inertial-camera odometry,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5848–5854.
- [21] X. Zuo, Y. Yang, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, “Lic-fusion 2.0: Lidar-inertial-camera odometry with sliding-window plane-feature tracking,” in *IROS 2020*, 2020.
- [22] T. Qin, P. Li, and S. Shen, “Vins-mono: A robust and versatile monocular visual-inertial state estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
- [23] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, “Integrating generic sensor fusion algorithms with sound state representations through encapsulation of manifolds,” *Information Fusion*, vol. 14, no. 1, pp. 57–77, 2013.
- [24] D. He, W. Xu, and F. Zhang, “Embedding manifold structures into kalman filters,” *arXiv preprint arXiv:2102.03804*, 2021.
- [25] J. Sola, “Quaternion kinematics for the error-state kalman filter,” *arXiv preprint arXiv:1711.02508*, 2017.
- [26] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2017.

APPENDIX

A. Computation of $\mathbf{F}_{\delta\hat{\mathbf{x}}}$ and \mathbf{F}_w

$$\begin{aligned} \mathbf{F}_{\delta\hat{\mathbf{x}}} &= \left. \frac{\partial(\delta\hat{\mathbf{x}}_{i+1})}{\partial\delta\hat{\mathbf{x}}_i} \right|_{\delta\hat{\mathbf{x}}_i=0, \mathbf{w}_i=0} \\ &= \begin{bmatrix} \text{Exp}(-\hat{\omega}_i\Delta t) & 0 & 0 & 0 & -\mathbf{J}_r(\hat{\omega}_i\Delta t)\Delta t & 0 \\ 0 & \mathbf{I} & 0 & 0 & \mathbf{I}\Delta t & 0 \\ 0 & 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{I} & 0 & 0 \\ -{}^G\hat{\mathbf{R}}_{I_i}[\hat{\mathbf{a}}_i]_{\times}\Delta t & 0 & 0 & 0 & \mathbf{I} & -{}^G\hat{\mathbf{R}}_{I_i}\Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{I} \end{bmatrix} \\ \mathbf{F}_w &= \left. \frac{\partial(\delta\hat{\mathbf{x}}_{i+1})}{\partial\mathbf{w}_i} \right|_{\delta\hat{\mathbf{x}}_i=0, \mathbf{w}_i=0} \\ &= \begin{bmatrix} -\mathbf{J}_r(\hat{\omega}_i\Delta t)\Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -{}^G\hat{\mathbf{R}}_{I_i}\Delta t & 0 & 0 \\ 0 & 0 & \mathbf{I}\Delta t & 0 \\ 0 & 0 & 0 & \mathbf{I}\Delta t \end{bmatrix} \end{aligned}$$

where $\hat{\omega}_i = \omega_{m_i} - \mathbf{b}_{g_i}$, $\hat{\mathbf{a}}_i = \mathbf{a}_{m_i} - \mathbf{b}_{a_i}$, and $\mathbf{J}_r(\cdot)$ are called the *right Jacobian matrix* of $SO(3)$:

$$\mathbf{J}_r(\mathbf{r}) = \mathbf{I} - \frac{1 - \cos||\mathbf{r}||}{||\mathbf{r}||^2} [\mathbf{r}]_{\times} + \frac{||\mathbf{r}|| - \sin(||\mathbf{r}||)}{||\mathbf{r}||^3} [\mathbf{r}]_{\times}^2, \mathbf{r} \in \mathbb{R}^3$$

For the detailed derivation of $\mathbf{F}_{\delta\hat{\mathbf{x}}}$ and \mathbf{F}_w , please refer to the Section. B of our supplementary material¹³.

B. The computation of \mathcal{H}

$$\begin{aligned} \mathcal{H} &= \frac{(\hat{\mathbf{x}}_{k+1} \boxplus \delta\hat{\mathbf{x}}_{k+1}) \boxminus \hat{\mathbf{x}}_{k+1}}{\partial\delta\hat{\mathbf{x}}_{k+1}}|_{\delta\hat{\mathbf{x}}_{k+1}=0} \\ &= \begin{bmatrix} \mathbf{A} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{0} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{9 \times 9} \end{bmatrix} \end{aligned}$$

where the 3×3 matrix $\mathbf{A} = \mathbf{J}_r^{-1}(\text{Log}({}^G\check{\mathbf{R}}_{I_{k+1}} {}^T G\check{\mathbf{R}}_{I_{k+1}}))$ and $\mathbf{B} = \mathbf{J}_r^{-1}(\text{Log}({}^I\check{\mathbf{R}}_{C_{k+1}} {}^T I\check{\mathbf{R}}_{C_{k+1}}))$. $\mathbf{J}_r^{-1}(\cdot)$ are called the *inverse right Jacobian matrix* of $SO(3)$:

$$\mathbf{J}_r^{-1}(\mathbf{r}) = \mathbf{I} + \frac{1}{2} [\mathbf{r}]_{\times} + \left(\frac{1}{||\mathbf{r}||^2} - \frac{1 + \cos(||\mathbf{r}||)}{2||\mathbf{r}|| \sin(||\mathbf{r}||)} \right) [\mathbf{r}]_{\times}^2 \quad (23)$$

C. The computation of \mathbf{H}_j^l

$$\mathbf{H}_j^l = \mathbf{u}_j^T [-{}^G\check{\mathbf{R}}_{I_{k+1}} [\mathbf{P}_a]_{\times} \quad \mathbf{I}_{3 \times 3} \quad \mathbf{0}_{3 \times 15}]$$

where $\mathbf{P}_a = {}^I\check{\mathbf{R}}_L {}^L\mathbf{p}_j + {}^I\mathbf{p}_L$.

For the detailed derivation of \mathbf{H}_j^l , please refer to the Section. D of our supplementary material.

D. The computation of \mathbf{H}_s^c and $\mathbf{F}_{\mathbf{P}_s}$

$$\begin{aligned} \mathbf{H}_s^c &= -\mathbf{F}_A \cdot \mathbf{F}_B \\ \mathbf{F}_{\mathbf{P}_s} &= -\mathbf{F}_A \cdot \mathbf{F}_C \end{aligned}$$

with:

$$\mathbf{F}_A = \frac{1}{C P_{sz}} \begin{bmatrix} f_x & 0 & -f_x \frac{C P_{sx}}{C P_{sz}} \\ 0 & f_y & -f_y \frac{C P_{sy}}{C P_{sz}} \end{bmatrix} \quad (24)$$

$$\mathbf{F}_B = [\mathbf{M}_A \quad \mathbf{M}_B \quad \mathbf{M}_C \quad \mathbf{M}_D \quad \mathbf{0}_{3 \times 12}]$$

$$\mathbf{F}_C = ({}^G\check{\mathbf{R}}_{I_{k+1}} {}^I\check{\mathbf{R}}_C)^T \quad (25)$$

where f_x and f_y are the focal length, c_x and c_y are the principal point offsets in image plane, and the 3×3 matrix \mathbf{M}_A , \mathbf{M}_B and \mathbf{M}_C are:

$$\begin{aligned} \mathbf{M}_A &= -({}^I\check{\mathbf{R}}_{C_{k+1}})^T ({}^G\check{\mathbf{R}}_{I_{k+1}})^T \\ \mathbf{M}_B &= ({}^I\check{\mathbf{R}}_{C_{k+1}})^T [{}^G\check{\mathbf{R}}_{I_{k+1}}^T ({}^G\mathbf{P}_s - {}^G\check{\mathbf{p}}_{I_{k+1}})]_{\times} \\ \mathbf{M}_C &= [{}^C\mathbf{P}_s]_{\times} \\ \mathbf{M}_D &= -({}^I\check{\mathbf{R}}_C)^T \end{aligned} \quad (26)$$

For the detailed derivation of \mathbf{H}_s^c and $\mathbf{F}_{\mathbf{P}_s}$, please refer to Section. E of our supplementary material.

¹³https://github.com/hku-mars/r2live/blob/master/supply/r2live_Supplementary_material.pdf

Supplementary Material: R²LIVE: A Robust, Real-time, LiDAR-Inertial-Visual tightly-coupled state Estimator and mapping

A. Perturbation on SO(3)

In this appendix, we will use the following approximation of perturbation $\delta \mathbf{r} \rightarrow \mathbf{0}$ on $SO(3)$ [25, 26]:

$$\text{Exp}(\mathbf{r} + \delta \mathbf{r}) \approx \text{Exp}(\mathbf{r}) \text{Exp}(\mathbf{J}_r(\mathbf{r}) \delta \mathbf{r})$$

$$\text{Exp}(\mathbf{r}) \text{Exp}(\delta \mathbf{r}) \approx \text{Exp}(\mathbf{r} + \mathbf{J}_r^{-1}(\mathbf{r}) \delta \mathbf{r})$$

$$\mathbf{R} \cdot \text{Exp}(\delta \mathbf{r}) \cdot \mathbf{u} \approx \mathbf{R} (\mathbf{I} + [\delta \mathbf{r}]_{\times}) \mathbf{u} = \mathbf{R} \mathbf{u} - \mathbf{R} [\mathbf{u}]_{\times} \delta \mathbf{r}$$

where $\mathbf{u} \in \mathbb{R}^3$ and we use $[\cdot]_{\times}$ denote the skew-symmetric matrix of vector (\cdot) ; $\mathbf{J}_r(\mathbf{r})$ and $\mathbf{J}_r^{-1}(\mathbf{r})$ are called the *right Jacobian* and the *inverse right Jacobian* of $SO(3)$, respectively.

$$\mathbf{J}_r(\mathbf{r}) = \mathbf{I} - \frac{1 - \cos(\|\mathbf{r}\|)}{\|\mathbf{r}\|^2} [\mathbf{r}]_{\times} + \frac{\|\mathbf{r}\| - \sin(\|\mathbf{r}\|)}{\|\mathbf{r}\|^3} [\mathbf{r}]_{\times}^2$$

$$\mathbf{J}_r^{-1}(\mathbf{r}) = \mathbf{I} + \frac{1}{2} [\mathbf{r}]_{\times} + \left(\frac{1}{\|\mathbf{r}\|^2} - \frac{1 + \cos(\|\mathbf{r}\|)}{2\|\mathbf{r}\| \sin(\|\mathbf{r}\|)} \right) [\mathbf{r}]_{\times}^2$$

B. Computation of $\mathbf{F}_{\delta \hat{\mathbf{x}}}$ and $\mathbf{F}_{\mathbf{w}}$

Combing (4) and (6), we have:

$$\begin{aligned} \delta \hat{\mathbf{x}}_{i+1} &= \mathbf{x}_{i+1} \boxplus \hat{\mathbf{x}}_{i+1} \\ &= (\mathbf{x}_i \boxplus (\Delta t \cdot \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i))) \boxplus (\hat{\mathbf{x}}_i \boxplus (\Delta t \cdot \mathbf{f}(\hat{\mathbf{x}}_i, \mathbf{u}_i, \mathbf{0}))) \\ &= \left[\begin{array}{c} \text{Log} \left(\left({}^G \hat{\mathbf{R}}_{I_i} \text{Exp}(\hat{\omega}_i \Delta t) \right)^T \cdot \left({}^G \hat{\mathbf{R}}_{I_i} \text{Exp}({}^G \delta \hat{\mathbf{r}}_{I_i}) \text{Exp}(\hat{\omega}_i \Delta t) \right) \right) \\ {}^G \delta \hat{\mathbf{p}}_{I_i} + {}^G \delta \hat{\mathbf{v}}_i \Delta t \\ {}^I \delta \hat{\mathbf{r}}_{C_i} \\ {}^G \delta \hat{\mathbf{v}}_i + \left({}^G \hat{\mathbf{R}}_{I_i} \text{Exp}({}^G \delta \hat{\mathbf{r}}_{I_i}) \right) \mathbf{a}_i \Delta t - {}^G \hat{\mathbf{R}}_{I_i} \hat{\mathbf{a}}_i \Delta t \\ {}^G \delta \hat{\mathbf{b}}_{\mathbf{g}_i} + \mathbf{n}_{\mathbf{b}_{\mathbf{g}_i}} \Delta t \\ {}^G \delta \hat{\mathbf{b}}_{\mathbf{a}_i} + \mathbf{n}_{\mathbf{b}_{\mathbf{a}_i}} \Delta t \end{array} \right] \end{aligned}$$

with:

$$\hat{\omega}_i = \omega_{m_i} - \hat{\mathbf{b}}_{\mathbf{g}_i}, \quad \omega_i = \hat{\omega}_i - \delta \hat{\mathbf{b}}_{\mathbf{g}_i} - \mathbf{n}_{\mathbf{g}_i} \quad (\text{S1})$$

$$\hat{\mathbf{a}}_i = \mathbf{a}_{m_i} - \hat{\mathbf{b}}_{\mathbf{a}_i}, \quad \mathbf{a}_i = \hat{\mathbf{a}}_i - \delta \hat{\mathbf{b}}_{\mathbf{a}_i} - \mathbf{n}_{\mathbf{a}_i} \quad (\text{S2})$$

And we have the following simplification and approximation from Section A.

$$\begin{aligned} &\text{Log} \left(\left({}^G \hat{\mathbf{R}}_{I_i} \text{Exp}(\hat{\omega}_i \Delta t) \right)^T \cdot \left({}^G \hat{\mathbf{R}}_{I_i} \text{Exp}({}^G \delta \hat{\mathbf{r}}_{I_i}) \text{Exp}(\hat{\omega}_i \Delta t) \right) \right) \\ &= \text{Log} \left(\text{Exp}(\hat{\omega}_i \Delta t)^T \cdot \left(\text{Exp}({}^G \delta \hat{\mathbf{r}}_{I_i}) \cdot \text{Exp}(\hat{\omega}_i \Delta t) \right) \right) \\ &\approx \text{Log} \left(\text{Exp}(\hat{\omega}_i \Delta t)^T \text{Exp}({}^G \delta \hat{\mathbf{r}}_{I_i}) \text{Exp}(\hat{\omega}_i \Delta t) \cdot \text{Exp}(-\mathbf{J}_r(\hat{\omega}_i \Delta t) (\delta \hat{\mathbf{b}}_{\mathbf{g}_i} + \mathbf{n}_{\mathbf{g}_i}) \Delta t) \right) \\ &\approx \text{Exp}(\hat{\omega}_i \Delta t) \cdot {}^G \delta \hat{\mathbf{r}}_{I_i} - \mathbf{J}_r(\hat{\omega}_i \Delta t) \delta \hat{\mathbf{b}}_{\mathbf{g}_i} \Delta t - \mathbf{J}_r(\hat{\omega}_i \Delta t) \mathbf{n}_{\mathbf{g}_i} \Delta t \\ &\quad \left({}^G \hat{\mathbf{R}}_{I_i} \text{Exp}({}^G \delta \hat{\mathbf{r}}_{I_i}) \right) \mathbf{a}_i \Delta t \\ &\approx \left({}^G \hat{\mathbf{R}}_{I_i} \left(\mathbf{I} + [{}^G \delta \hat{\mathbf{r}}_{I_i}]_{\times} \right) \right) (\hat{\mathbf{a}}_i - \delta \hat{\mathbf{b}}_{\mathbf{a}_i} - \mathbf{n}_{\mathbf{a}_i}) \Delta t \\ &\approx {}^G \hat{\mathbf{R}}_{I_i} \hat{\mathbf{a}}_i \Delta t - {}^G \hat{\mathbf{R}}_{I_i} \delta \hat{\mathbf{b}}_{\mathbf{a}_i} \Delta t - {}^G \hat{\mathbf{R}}_{I_i} \mathbf{n}_{\mathbf{a}_i} \Delta t - {}^G \hat{\mathbf{R}}_{I_i} [\hat{\mathbf{a}}_i]_{\times} {}^G \delta \hat{\mathbf{r}}_{I_i} \Delta t \end{aligned}$$

To conclude, we have the computation of $\mathbf{F}_{\delta \hat{\mathbf{x}}}$ and $\mathbf{F}_{\mathbf{w}}$ as follow:

$$\begin{aligned} \mathbf{F}_{\delta \hat{\mathbf{x}}} &= \frac{\partial (\delta \hat{\mathbf{x}}_{i+1})}{\partial \delta \hat{\mathbf{x}}_i} \Big|_{\delta \hat{\mathbf{x}}_i=0, \mathbf{w}_i=0} \\ &= \left[\begin{array}{cccccc} \text{Exp}(-\hat{\omega}_i \Delta t) & 0 & 0 & 0 & -\mathbf{J}_r(\hat{\omega}_i \Delta t) \Delta t & 0 \\ 0 & \mathbf{I} & 0 & 0 & \mathbf{I} \Delta t & 0 \\ 0 & 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbf{I} & 0 & 0 \\ -{}^G \hat{\mathbf{R}}_{I_i} [\hat{\mathbf{a}}_i]_{\times} \Delta t & 0 & 0 & 0 & \mathbf{I} & -{}^G \hat{\mathbf{R}}_{I_i} \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{I} \end{array} \right] \end{aligned}$$

$$\begin{aligned} \mathbf{F}_{\mathbf{w}} &= \frac{\partial (\delta \hat{\mathbf{x}}_{i+1})}{\partial \mathbf{w}_i} \Big|_{\delta \hat{\mathbf{x}}_i=0, \mathbf{w}_i=0} \\ &= \left[\begin{array}{cccc} -\mathbf{J}_r(\hat{\omega}_i \Delta t) \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -{}^G \hat{\mathbf{R}}_{I_i} \Delta t & 0 & 0 \\ 0 & 0 & \mathbf{I} \Delta t & 0 \\ 0 & 0 & 0 & \mathbf{I} \Delta t \end{array} \right] \end{aligned}$$

C. The computation of \mathcal{H}

Recalling (15), we have:

$$\begin{aligned} \mathcal{H} &= \frac{(\check{\mathbf{x}}_{k+1} \boxplus \delta \check{\mathbf{x}}_{k+1}) \boxminus \hat{\mathbf{x}}_{k+1}}{\partial \delta \check{\mathbf{x}}_{k+1}} \Big|_{\delta \check{\mathbf{x}}_{k+1}=0} \\ &= \left[\begin{array}{ccccc} \mathbf{A} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{0} & \mathbf{B} & \mathbf{0} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0}_{3 \times 9} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{9 \times 9} \end{array} \right] \end{aligned}$$

with the 3×3 matrix $\mathbf{A} = \mathbf{J}_r^{-1}(\text{Log}({}^G \hat{\mathbf{R}}_{I_{k+1}} {}^T {}^G \check{\mathbf{R}}_{I_{k+1}}))$ and $\mathbf{B} = \mathbf{J}_r^{-1}(\text{Log}({}^I \hat{\mathbf{R}}_{C_{k+1}} {}^T {}^I \check{\mathbf{R}}_{C_{k+1}}))$.

D. The computation of \mathbf{H}_j^l

Recalling (12) and (15), we have:

$$\begin{aligned} \mathbf{r}_l(\check{\mathbf{x}}_{k+1} \boxplus \delta \check{\mathbf{x}}_{k+1}, {}^L \mathbf{p}_j) &= \mathbf{u}_j^T ({}^G \check{\mathbf{p}}_{I_{k+1}} + {}^G \delta \check{\mathbf{p}}_{I_{k+1}} - \\ &\quad \mathbf{q}_j + {}^G \check{\mathbf{R}}_{I_{k+1}} \text{Exp}({}^G \delta \check{\mathbf{r}}_{I_{k+1}}) ({}^I \mathbf{R}_L {}^L \mathbf{p}_j + {}^I \mathbf{p}_L)) \quad (\text{S3}) \end{aligned}$$

And with the small perturbation approximation, we get:

$$\begin{aligned} &{}^G \check{\mathbf{R}}_{I_{k+1}} \text{Exp}({}^G \delta \check{\mathbf{r}}_{I_{k+1}}) \mathbf{P}_{\mathbf{a}} \\ &\approx {}^G \check{\mathbf{R}}_{I_{k+1}} \left(\mathbf{I} + [{}^G \delta \check{\mathbf{r}}_{I_{k+1}}]_{\times} \right) \mathbf{P}_{\mathbf{a}} \\ &= {}^G \check{\mathbf{R}}_{I_{k+1}} \mathbf{P}_{\mathbf{a}} - {}^G \check{\mathbf{R}}_{I_{k+1}} [{}^G \delta \check{\mathbf{r}}_{I_{k+1}}]_{\times} \mathbf{P}_{\mathbf{a}} \quad (\text{S4}) \end{aligned}$$

where $\mathbf{P}_{\mathbf{a}} = {}^I \mathbf{R}_L {}^L \mathbf{p}_j + {}^I \mathbf{p}_L$. Combining (S3) and (S4) together we can obtain:

$$\mathbf{H}_j^l = \mathbf{u}_j^T [-{}^G \check{\mathbf{R}}_{I_{k+1}} [{}^G \delta \check{\mathbf{r}}_{I_{k+1}}]_{\times} \mathbf{I}_{3 \times 3} \quad \mathbf{0}_{3 \times 15}]$$

E. The computation of \mathbf{H}_s^c and $\mathbf{F}_{\mathbf{P}_s}$

Recalling (16), we have:

$${}^C \mathbf{P}_s = \mathbf{P}_C(\check{\mathbf{x}}_{k+1}, {}^C \mathbf{P}_s) = [{}^C P_{sx} \quad {}^C P_{sy} \quad {}^C P_{sz}]^T$$

where the function $\mathbf{P}_C(\check{\mathbf{x}}_{k+1}, {}^C \mathbf{P}_s)$ is:

$${}^C \mathbf{P}_s = ({}^G \check{\mathbf{R}}_{I_{k+1}} {}^I \check{\mathbf{R}}_{C_{k+1}})^T {}^C \mathbf{P}_s - {}^I \check{\mathbf{R}}_{C_{k+1}} {}^I \check{\mathbf{p}}_{C_{k+1}} \quad (\text{S5})$$

$$- ({}^G \check{\mathbf{R}}_{I_{k+1}} {}^I \check{\mathbf{R}}_{C_{k+1}})^T {}^G \check{\mathbf{p}}_{I_{k+1}} \quad (\text{S6})$$

From (20), we have:

$$\mathbf{r}_c(\check{\mathbf{x}}_{k+1}, {}^C \mathbf{p}_s, {}^C \mathbf{P}_s) = {}^C \mathbf{p}_s - \boldsymbol{\pi}({}^C \mathbf{P}_s)$$

$$\boldsymbol{\pi}({}^C \mathbf{P}_s) = \left[f_x \frac{{}^C P_{sx}}{{}^C P_{sz}} + c_x \quad f_y \frac{{}^C P_{sy}}{{}^C P_{sz}} + c_y \right]^T \quad (\text{S7})$$

where f_x and f_y are the focal length, c_x and c_y are the principal point offsets in image plane.

For conveniently, we omit the $(\cdot)|_{\delta\check{\mathbf{x}}_{k+1}=\mathbf{0}}$ in the following derivation, and we have:

$$\mathbf{H}_s^c = -\frac{\partial \boldsymbol{\pi}(^C\mathbf{P}_s)}{\partial ^C\mathbf{P}_s} \cdot \frac{\partial \mathbf{P}_C(\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s)}{\partial \delta\check{\mathbf{x}}_{k+1}} \quad (\text{S8})$$

$$\mathbf{F}_{\mathbf{P}_s} = -\frac{\partial \boldsymbol{\pi}(^C\mathbf{P}_s)}{\partial ^C\mathbf{P}_s} \cdot \frac{\partial \mathbf{P}_C(\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s)}{\partial ^G\mathbf{P}_s} \quad (\text{S9})$$

where:

$$\frac{\partial \boldsymbol{\pi}(^C\mathbf{P}_s)}{\partial ^C\mathbf{P}_s} = \frac{1}{^C P_{sz}} \begin{bmatrix} f_x & 0 & -f_x \frac{^C P_{sx}}{^C P_{sz}} \\ 0 & f_y & -f_y \frac{^C P_{sy}}{^C P_{sz}} \end{bmatrix} \quad (\text{S10})$$

$$\frac{\partial \mathbf{P}_b(\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s)}{\partial ^G\mathbf{P}_s} = \left({}^G \check{\mathbf{R}}_{I_{k+1}} {}^I \check{\mathbf{R}}_C \right)^T \quad (\text{S11})$$

According to Section A, we have the following approximation of $\mathbf{P}_C(\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s)$:

$$\begin{aligned} & \mathbf{P}_C(\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s) \\ &= \left({}^G \check{\mathbf{R}}_{I_{k+1}} \text{Exp}\left({}^G \delta\check{\mathbf{r}}_{I_{k+1}}\right) {}^I \check{\mathbf{R}}_{C_{k+1}} \text{Exp}\left({}^I \delta\check{\mathbf{r}}_{C_{k+1}}\right) \right)^T {}^G \mathbf{P}_s \\ & - \left({}^I \check{\mathbf{R}}_{C_{k+1}} \text{Exp}\left({}^I \delta\check{\mathbf{r}}_{C_{k+1}}\right) \right)^T \left({}^I \check{\mathbf{p}}_{C_{k+1}} + {}^I \delta\check{\mathbf{p}}_{C_{k+1}} \right) \\ & - \left({}^G \check{\mathbf{R}}_{I_{k+1}} \text{Exp}\left({}^G \delta\check{\mathbf{r}}_{I_{k+1}}\right) {}^I \check{\mathbf{R}}_{C_{k+1}} \text{Exp}\left({}^I \delta\check{\mathbf{r}}_{C_{k+1}}\right) \right)^T \left({}^G \check{\mathbf{p}}_{I_{k+1}} + {}^G \delta\check{\mathbf{p}}_{I_{k+1}} \right) \\ & \approx \mathbf{P}_b(\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s) - \left({}^I \check{\mathbf{R}}_{C_{k+1}} \right)^T \left({}^G \check{\mathbf{R}}_{I_{k+1}} \right)^T {}^G \delta\check{\mathbf{r}}_{I_{k+1}} \\ & + \left({}^I \check{\mathbf{R}}_{C_{k+1}} \right)^T \left[{}^G \check{\mathbf{R}}_{I_{k+1}}^T ({}^G \mathbf{P}_s - {}^G \check{\mathbf{p}}_{I_{k+1}}) \right]_{\times} {}^G \delta\check{\mathbf{p}}_{I_{k+1}} \\ & + \left[\mathbf{P}_b(\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s) \right]_{\times} {}^I \delta\check{\mathbf{r}}_{C_{k+1}} - \left({}^I \check{\mathbf{R}}_C \right)^T {}^I \delta\check{\mathbf{p}}_{C_{k+1}} \end{aligned} \quad (\text{S12})$$

With this, we can derive:

$$\frac{\partial \mathbf{P}_C(\check{\mathbf{x}}_{k+1} \boxplus \delta\check{\mathbf{x}}_{k+1}, ^G\mathbf{P}_s)}{\partial \delta\check{\mathbf{x}}_{k+1}} = [\mathbf{M}_A \ \mathbf{M}_B \ \mathbf{M}_C \ \mathbf{M}_D \ \mathbf{0}_{3 \times 12}] \quad (\text{S13})$$

$$\begin{aligned} \mathbf{M}_A &= -\left({}^I \check{\mathbf{R}}_{C_{k+1}} \right)^T \left({}^G \check{\mathbf{R}}_{I_{k+1}} \right)^T \\ \mathbf{M}_B &= \left({}^I \check{\mathbf{R}}_{C_{k+1}} \right)^T \left[{}^G \check{\mathbf{R}}_{I_{k+1}}^T ({}^G \mathbf{P}_s - {}^G \check{\mathbf{p}}_{I_{k+1}}) \right]_{\times} \\ \mathbf{M}_C &= \left[{}^C \mathbf{P}_s \right]_{\times} \\ \mathbf{M}_D &= -\left({}^I \check{\mathbf{R}}_C \right)^T \end{aligned} \quad (\text{S14})$$

Substituting (S10), (S11) and (S13) into (S8) and (S9), we finish the computation of \mathbf{H}_s^c and $\mathbf{F}_{\mathbf{P}_s}$.