# TC$^2$LI-SLAM: A Tightly-Coupled Camera-LiDAR-Inertial SLAM System

Yunze Tong, Xuebo Zhang, *Senior Member, IEEE*, Runhua Wang, Zhixing Song, Songyang Wu, Shiyong Zhang, Youwei Wang, and Jing Yuan

***Abstract*—In this letter, we propose TC$^2$LI-SLAM, a novel and tightly-coupled camera-LiDAR-inertial SLAM system with a separated front end and a unified back end. The front end accepts non-identical frequency inputs and performs real-time pose estimation while incorporating different inputs into keyframes for the back end, benefiting from our novel keyframe-oriented data synchronization method. To improve the accuracy of the back-end optimization, we innovatively leverage multi-sensor consecutive co-vision constraints in a keyframe-based local bundle adjustment problem. To bridge the gap between different residual terms from different sensors in the optimization problem, we propose a cross-modal residual standardization method that can be easily applied. Furthermore, the proposed system can degrade into TC$^2$L-SLAM, a tightly-coupled camera-LiDAR SLAM system without IMU, to cope with situations where inertial measurements are not available or reliable. Finally, both TC$^2$LI-SLAM and TC$^2$L-SLAM are tested on public and self-recorded datasets. The results show their superior performance in terms of accuracy compared with the state-of-the-art SLAM systems, with at most 33% accuracy improvement over ORB-SLAM3 on the KITTI dataset.**

***Index Terms*—SLAM, sensor fusion.**

## I. Introduction

TODAY'S most commonly used sensors in simultaneous localization and mapping (SLAM) systems are cameras, light detection and ranging (LiDAR) sensors, and inertial measurement units (IMU). Visual SLAMs feature cameras as their input sensors. Monocular cameras are low-cost and can provide dense features with rich texture information. In addition, stereo cameras are capable of ranging and do not suffer from scale drift, which is a widely recognized problem in monocular visual SLAMs [1], [2], [3]. Thus, visual SLAMs, especially stereo ones, are highly accurate in certain environments. However, due to high dependence on image textures and a small field of view (FOV), they often fail in sparse-textured environments or under violent movements [4].

On the other hand, LiDAR SLAMs present more stable performance and can achieve higher accuracy. Most of them use spinning LiDARs, which provide direct depth information with a much larger FOV. On the other hand, some use solid-state LiDARs which provide a smaller FOV and denser features. However, LiDAR SLAMs rely highly on geometric information, thus may fail in scenes with obscure geometric features [5], [6]. In addition, LiDAR sensors suffer from motion distortion under high-speed movements, which would increase the drift in large-scale environments.

Unlike the above sensors, IMUs can provide linear acceleration and angular velocity measurements at a high frequency, and thus can effectively adapt to high-speed motion scenes. However, IMUs cannot sense the external environment and suffer from inevitably time-varying bias, which can only be compensated by measurements from external sensing sensors like LiDARs and cameras.

In summary, tightly fusing measurements from different sensors can reduce trajectory drift and improve overall performance. To this end, we feature the following contributions in this letter:

1) We propose a real-time and accurate camera-LiDAR-inertial SLAM system, namely TC$^2$LI-SLAM, that can be degraded into TC$^2$L-SLAM when inertial measurements are either unavailable or unreliable.
2) An efficient keyframe-oriented data synchronization algorithm is designed, pairing data with different frequencies for keyframe-based tightly-coupled processing.
3) We design a tightly-coupled bundle adjustment module to improve accuracy, where both visual and LiDAR residuals are built with consecutive co-vision constraints.
4) We propose a handy cross-modal residual standardization method to adapt residuals from different sensors into a unified optimization problem.

The rest of the letter is organized as follows. Section II reviews some related works to comparatively highlight our contribution and differences from others. Section III presents the overview of the proposed system. Section IV–VI provides details of the proposed approaches in the system. In Section VII, we test our proposed SLAM systems on both public and self-recorded datasets, and then further analyze the test results.

Manuscript received 5 March 2024; accepted 25 June 2024. Date of publication 10 July 2024; date of current version 17 July 2024. This article was recommended for publication by Associate Editor R. Sagawa and Editor P. Vasseur upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62293513/62293510, in part by the Natural Science Foundation of Tianjin under Grant 22JCZDJC00810, and in part by the Fundamental Research Funds for the Central Universities. *(Corresponding author: Xuebo Zhang.)*

The authors are with the College of Artificial Intelligence, Institute of Robotics and Automatic Information System, and the Tianjin Key Laboratory of Intelligent Robotics, Nankai University, Tianjin 300350, China (e-mail: tongyunze@mail.nankai.edu.cn; zhangxuebo@nankai.edu.cn).

Our open-source implementation is available at https://github.com/sigerson-925/TC2LI-SLAM.

Video available at https://youtu.be/oN1kkvkBdX8.

Digital Object Identifier 10.1109/LRA.2024.3426375

## II. Related Works

Earlier attempts researchers made focus on single-sensor methods. LiDAR SLAMs, represented by LOAM [5], provide the relative pose between consecutive scans via scan-to-scan point cloud registration. The pose is further refined by scan-to-map registration. Visual SLAMs can be divided into direct methods and indirect methods. Direct methods compute the relative pose between consecutive images by minimizing the photometric error. Classic direct methods like LSD-SLAM [8] perform tracking and mapping directly with image pixel intensities, with a strong assumption of constant luminosity, which might not be satisfied in practical scenarios. On the other hand, indirect methods like ORB-SLAM [9], [10] minimize the re-projection error between paired features, which is more time-consuming due to feature extraction and association. Some researchers combine direct and indirect methods to achieve a balance between efficiency and accuracy [11], [12], [13].

In challenging environments, single-sensor approaches may fail due to degraded measurements. Drawn by better robustness and accuracy of multi-sensor SLAM systems, researchers have been working on fusing different sensors to improve overall performance in recent years. Earlier ideas were to fuse inertial measurements on top of single-sensor methods, where inertial measurements are used for pose prediction, which is further refined with measurements from other sensors [14], [15], [16], [17], [18], [19], [20]. LIO-SAM [21] extends LOAM with IMU and adds a factor graph in the back end to further improve accuracy. FAST-LIO [22] proposes a novel approach for the Kalman gain computation in the error-state iterated Kalman filter (ESIKF) framework. FAST-LIO2 [23] further improves its performance by introducing an incremental KD-Tree structure.

VINS-Mono [16] is a monocular visual inertial odometry system, which uses the Lucas-Kanade optical flow method for tracking, and optimizes tracked poses based on a 4-DoF pose graph. ORB-SLAM3 [3] is an accurate and robust open-source visual-inertial SLAM system, including three main threads, namely tracking, local mapping, and loop closing. It should be noted that the robustness of ORB-SLAM3 under long-time and large-scale scenes is further improved by its atlas system and place recognition algorithm.

To achieve higher performance, researchers start to investigate the fusion of LiDAR and camera, some with an additional IMU. These methods can be categorized into loosely-coupled methods and tightly-coupled methods. The former focuses on robustness and efficiency, while the latter aims to obtain higher localization and mapping accuracy.

Some works, such as V-LOAM [24], loosely fuse camera and LiDAR, using visual odometry to provide initial guesses for LiDAR pose estimation. Others, like LIMO [25], DEMO [26], CamVox [27] and NALO-VOM [28], utilize LiDAR measurements to provide depth for monocular cameras. LVI-SAM [29] comprises two independent subsystems: LiDAR depth information aids in reconstructing visual features, while visual odometry offers initial guesses for point cloud registration. The back end performs factor graph optimization, incorporating IMU preintegration, odometry, and loop closure constraints. LVIO-Fusion [30] includes additional GPS constraints in the factor graph optimization and introduces a segmented optimization method for urban traffic scenes.

Recent works devote more to tightly coupling these sensors. LIC-fusion [31] tightly couples inertial measurements, sparse visual features, and LiDAR features in the multi-state constraint Kalman filter (MSCKF) framework for state estimation. Its upgrade version LIC-fusion 2.0 [32] tracks planar features in a sliding window and applies them in the optimization of pose estimation. TVL-SLAM [33] has independent visual and LiDAR front ends, with a large-scale bundle adjustment module in the back end, resulting in huge time consumption. R$^2$LIVE [34] estimates the real-time state in a filter-based odometry, and further improves the precision with a factor graph optimization. R$^3$LIVE [35] and FAST-LIVO [36] tightly fuse a LiDAR and a camera in terms of map construction, where the LiDAR-inertial odometry and the visual-inertial odometry leverage the same map to respectively optimize their poses. They can be adapted to both spinning LiDARs and solid-state LiDARs. mVIL-Fusion [37] fuses a monocular camera, IMU and LiDAR in a three-level framework, and its back end smoothes a pose-only factor graph. In addition, there are some learning-based approaches [38], [39] as well, which aim to explore the recent AI techniques.

Notably, the main challenges of the existing multi-sensor fusion SLAM systems and our solutions to them are:

- Existing tightly-coupled systems barely incorporate co-visual information from both images and LiDAR scans in their joint optimizations. By fully utilizing multi-sensor co-vision information, our systems achieve higher localization accuracy.
- Back-end optimization methods are mostly keyframe-based, while none of the above systems provide a detailed design of a keyframe-oriented data synchronization method like ours.
- Only TVL-SLAM [33] investigate the standardization of residual definitions, in which only the standardization of state update methods is included. By addressing the effect of coordinate systems, our standardization method achieve better accuracy, with better adaptability to multi-sensor fusion SLAM systems.

## III. System Overview

The pipeline of the proposed TC$^2$LI-SLAM is shown in Fig. 1. Its separated front end contains a camera subsystem and a parallel LiDAR subsystem, each taking the corresponding sensor data and inertial measurements as inputs. The camera subsystem performs frame-to-frame pose estimation by minimizing re-projection errors of the reconstructed image features, using the relative pose produced by IMU preintegration as an initial guess. The LiDAR subsystem first processes inertial measurements and then compensates for the motion distortion in the LiDAR scan. Then, it performs a scan-to-map pose estimation, minimizing the point-to-plane errors for each valid LiDAR point, while filtering outliners. The remaining valid points will serve as planar features in the back end. Using the estimated pose, all input LiDAR

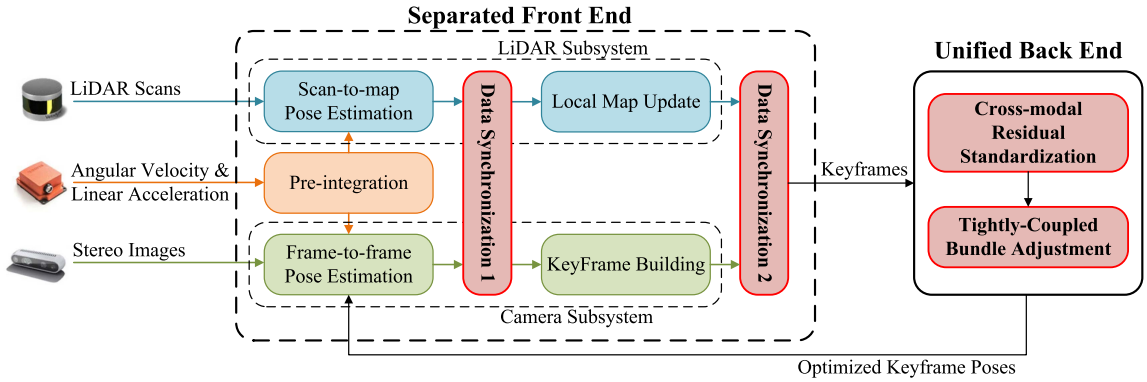



Fig. 1. General pipeline of the proposed method. The inputs from LiDAR, stereo camera and IMU first go through the separated front end, which contains two subsystems. All the processed data go through the keyframe-oriented data synchronization process running in the middle and the end of the front end. Finally, the keyframes are sent to a uniformed back end, where a keyframe-based tightly-coupled bundle adjustment is performed.

points are registered into the local map, which is stored in an ikd-Tree [23].

The proposed keyframe-oriented data synchronization approach operates in two stages in the front end to incorporate multi-sensor information into keyframes, while ensuring the real-time efficiency of data exchange. These keyframes are selected from image frames because we place more importance on maintaining co-visibility between consecutive keyframe images. In the unified back end, after cross-modal residual standardization, their contained information will be utilized in the proposed keyframe-based tightly-coupled bundle adjustment module. Due to high computational consumption, the back end runs at a lower frequency.

Furthermore, when inertial measurements are unavailable or considered unreliable, the proposed TC²LI-SLAM can be degraded into TC²L-SLAM, a tightly-coupled camera-LiDAR SLAM system. The LiDAR subsystem no longer performs pose estimation, yet still builds a local map and extracts point cloud features using camera poses.

As for implementation details, we made the following modifications on top of the open-source ORB-SLAM3 pipeline:

1) We add a parallel-threaded LiDAR front end to the tracking thread of ORB-SLAM3, which is modified from the open-source FAST-LIO2 [23].
2) We implement our proposed two-stage data synchronization algorithm in the middle and the end of the two parallel front-end threads, and manage to keep the real-time performance of the entire front end.
3) We add a LiDAR residual in the original local bundle adjustment module of ORB-SLAM3 (solve with g2o [40]). The co-vision-based LiDAR residual is implemented following BALM [41], with details shown in section V. Moreover, The residual standardization in section VI is implemented in the computation of Jacobian matrixes.

## IV. Keyframe-Oriented Data Synchronization

Input measurements from cameras and LiDARs are often captured at neither the same moment nor the same frequency. Considering that keyframes with information from both input sources are required in the back-end joint optimization, here we propose a keyframe-oriented data synchronization method, illustrated in Fig. 2. We incorporate information from both images and LiDAR scans into keyframes selected from regular image frames. As a result, image features and LiDAR planar features in any keyframe can be considered captured simultaneously at the keyframe's timestamp. In addition, the proposed method keeps the real-time performance of the front end, which is demonstrated in section VII-D. Inertial measurements are not included in the data synchronization, since the camera subsystem has already processed them based on image timestamps.

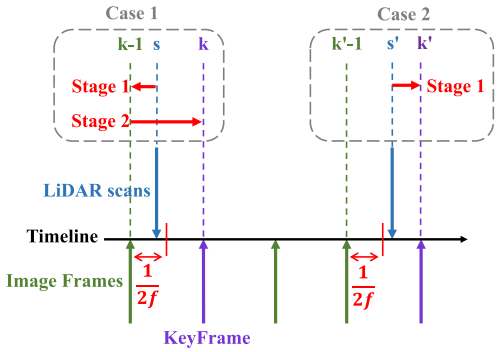



Fig. 2. Pipeline of our keyframe-oriented two-stage data synchronization algorithm and the two cases mentioned in Section IV. Every incoming LiDAR scan is paired to its nearest neighbor image frame. When a keyframe is built, it utilizes the planar features in the nearest previous scan as its own.

Since keyframes are selected from image frames, their closest LiDAR scan might be far away on timestamps. Thus, directly interpolating the scan with keyframe poses leads to inevitable bias. To ensure accuracy and efficiency, we divide the synchronization method into two stages, by first pairing each new scan with its closest image frame, then pairing them to keyframes when necessary.

### A. Stage 1: Pairing Scans to Regular Frames

After a LiDAR scan $j$ is received, it will be paired to an image frame $i$. For case 1 in Fig. 2, where the time delay between

the LiDAR scan and the latest image frame $k-1$ is less than $1/(2f)$, with $f$ being the camera's input frequency, scan $s$ is paired to frame $k-1$. Otherwise, as in case 2 in Fig. 2, scan $s'$ is paired to the next coming frame $k'$. After the pose estimation of the image frame $i$, where $i = k-1$ in case 1 and $i = k'$ in case 2, we convert the planar features of scan $j$ into frame $i$'s timestamp $t_i$, where $j = s$ in case 1 and $j = s'$ in case 2. Since we have the pose of the paired image frame ${}^{w}\mathbf{T}_i$ and the pose of the previous image frame and ${}^{w}\mathbf{T}_{i-1}$, the camera pose ${}^{w}\mathbf{T}_j$ at LiDAR timestamp $t_j$ is interpolated as

$$ {}^{w}\mathbf{T}_j = {}^{w}\mathbf{T}_{i-1} \cdot \mathbf{Exp}(\phi t), \tag{1} $$

where $\phi = Log({}^{w}\mathbf{T}_{i-1}^{-1} \cdot {}^{w}\mathbf{T}_i)$, $t = \dfrac{t_j - t_{i-1}}{t_i - t_{i-1}}$, and $\mathbf{Exp}(\cdot)$ is the exponential mapping of Lie algebra. Then we can transform each planar feature point ${}^{l}p_i$ to ${}^{l}p'_i$ as

$$ {}^{l}p'_i = \left({}^{c}\mathbf{T}_l{}^{-1} \cdot {}^{w}\mathbf{T}_i^{-1} \cdot {}^{w}\mathbf{T}_j \cdot {}^{c}\mathbf{T}_l\right) {}^{l}p_i, \tag{2} $$

where ${}^{c}\mathbf{T}_l$ is the LiDAR's extrinsic referring to the camera. Finally, the planar features can be considered captured at $t_i$.

### B. Stage 2: Pairing Scans to KeyFrames

This stage takes place at the end of the front end, whenever the front end selects a new keyframe from a regular image frame. For each new keyframe, we check if it has a paired LiDAR scan. In case 2, where the selected keyframe $k'$ has a paired LiDAR scan $s'$, no further process is required. In case 1, where keyframe $k$ was not paired, we find the former image frame $k-1$ with a paired LiDAR scan $s$. Since the planar features of scan $s$ were previously converted to the timestamp of $k-1$, they can easily be converted to the keyframe's timestamp with eq.(2), where $i = k$ and ${}^{w}\mathbf{T}_j$ is replaced by the keyframe pose ${}^{w}T_{k-1}$. Finally, all the keyframes will have planar feature points considered captured at their timestamp.

## V. Tightly-Coupled Bundle Adjustment

After data synchronization, the obtained keyframe contains a pose, re-constructed image features, LiDAR planar features and inertial measurements. Together with other keyframes in a local window, it is leveraged to perform a tightly-coupled bundle adjustment to refine the keyframe poses. To build the tightly-coupled optimization problem, a joint residual is constructed with information from all input sensors.

### A. Joint Residual Construction

Given a local window contains $k$ keyframes with their poses $\mathcal{T}_k = \{{}^{w}\mathbf{T}_1 \cdots {}^{w}\mathbf{T}_k\}$ and $p$ reconstructed image feature points $\mathcal{M} = \{\mathbf{m}_1 \cdots \mathbf{m}_p\}$, the local bundle adjustment is performed by minimizing the joint residual in eq.(3). $\mathcal{L}_i$ is the LiDAR residual in section V-B, $\mathcal{C}_{ij}$ is the re-projection error between image features and reconstructed feature points, and $\mathcal{I}_{i-1,i}$ is the IMU pre-integrate residual between the keyframe $i-1$ and $i$ following Froster's work [15]. Information matrices, $\Sigma_{\mathcal{I}_i}$, $\Sigma_{\mathcal{C}_{ij}}$ and $\Sigma_{\mathcal{L}_i}$, determine the weight of each residual. In the case of TC$^2$L-SLAM, the IMU residual will be removed.

$$ \min_{\mathcal{T}_k,\mathcal{M}} \left( \sum_{i=2}^{k} \left\| \mathcal{I}_{i-1,i} \right\|^2_{\Sigma_{\mathcal{I}_i}} + \sum_{j=1}^{p} \sum_{i=1}^{k} \left\| \mathcal{C}_{ij} \right\|_{\Sigma_{\mathcal{C}_{ij}}} + \sum_{i=1}^{k} \left\| \mathcal{L}_i \right\|_{\Sigma_{\mathcal{L}_i}} \right). \tag{3} $$

### B. LiDAR Residual Based on Co-Vision

To form the LiDAR residual in a bundle adjustment manner, we leverage the correlation information between planar features from different scans laying on the same actual plane. Name ${}^{j}\mathbf{p}_{ijk}$ as the position of the $k$th feature point in the $j$th scan in its local frame, lying on the $i$th plane. ${}^{j}\mathbf{p}_{ijk}$ can be transformed into the global coordinate system $w$ using the pose of the $j$th scan ${}^{w}\mathbf{T}_j = ({}^{w}\mathbf{R}_j, {}^{w}\mathbf{t}_j) \in SO(3) \times \mathbb{R}^3$, resulting in ${}^{w}\mathbf{p}_{ijk} = {}^{w}\mathbf{R}_j {}^{j}\mathbf{p}_{ijk} + {}^{w}\mathbf{t}_j$. The constrain generated from the point-to-plane error of $\mathbf{p}_{ijk}$ can be define as

$$ (\mathcal{T}^*, \mathbf{n}_i^*, \mathbf{q}_i^*) = \arg\min_{\mathcal{T},\mathbf{n},\mathbf{q}} \frac{1}{N_i} \sum_{j=1}^{M_p} \sum_{k=1}^{N_{ij}} \left\| \mathbf{n}_i^T \left( {}^{w}\mathbf{p}_{ijk} - \mathbf{q}_i \right) \right\|_2^2, \tag{4} $$

where $M_p$ is the total amount of scans and $N_{ij}$ is the total amount of $i$th plane's feature points in the $j$th scan. $\mathcal{T} = ({}^{w}\mathbf{T}_1, \ldots, {}^{w}\mathbf{T}_{M_p})$ represents the collection of the $M_p$ scans' poses, while $\mathbf{n}_i$ and $\mathbf{q}_i$ are the normal and the point of the $i$th plane.

By leveraging the parameter elimination method proposed by BALM [41], we manage to simplify the problem as

$$ \begin{aligned} &(\mathcal{T}^*, \mathbf{n}_i^*, \mathbf{q}_i^*) \\ &\quad = \arg\min_{\mathcal{T}} \left( \min_{\mathbf{n},\mathbf{q}} \frac{1}{N_i} \sum_{j=1}^{M_p} \sum_{k=1}^{N_{ij}} \left\| \mathbf{n}_i^T \left( {}^{w}\mathbf{p}_{ijk} - \mathbf{q}_i \right) \right\|_2^2 \right) \\ &\quad = \arg\min_{\mathcal{T}} \lambda_3(\mathbf{A}_i) \quad \text{when } \mathbf{n}_i^* = \mathbf{u}_3, \mathbf{q}_i^* = \overline{\mathbf{p}}_i, \end{aligned} \tag{5} $$

where $\lambda_k(\mathbf{A}_i)$ is the $k$th largest eigen value of $\mathbf{A}_i$, while $\mathbf{u}_k$ is $\lambda_k$'s eigen vector. detailed definitions of $\mathbf{A}_i$ and $\overline{\mathbf{p}}_i$ are

$$ \mathbf{A}_i \triangleq \frac{1}{N_i} \sum_{j=1}^{M_p} \sum_{k=1}^{N_{ij}} \left( {}^{w}\mathbf{p}_{ijk} - \overline{\mathbf{p}}_i \right), \quad \overline{\mathbf{p}}_i \triangleq \frac{1}{N_i} \sum_{j=1}^{M_p} \sum_{k=1}^{N_{ij}} {}^{w}\mathbf{p}_{ijk}. \tag{6} $$

Finally, the LiDAR residual can be defined as

$$ \mathbf{r}_{l_i} = \sum_{i=1}^{N_s} w_i \cdot r_{s_i}, r_{s_i} = \lambda_3(\mathbf{A}_i), \tag{7} $$

where $N_s$ is the total amount of planes and $w_i$ is the weight of every plane, affected by the amount of points lying on it.

## VI. Cross-Modal Residual Standardization

LiDAR and visual residuals are defined with different state variables, and thus their coordinate systems and their updating methods need to be unified. Here we propose a two-step cross-modal standardization approach, taking TC$^2$LI-SLAM as

an example. The LiDAR residual is defined with LiDAR sensor pose under global coordinate system ${}^w\mathbf{T}_l = ({}^w\mathbf{R}_l, {}^w\mathbf{t}_l) \in SO(3) \times \mathbb{R}^3$, while the camera residual and the IMU preintegration residual are defined with IMU pose ${}^w\mathbf{T}_b \in SE(3)$. To standardize them, we convert the LiDAR residual to be defined with ${}^w\mathbf{T}_b$. Notably, This method can be easily applied to any multi-modal SLAM/odometry system given state definitions, without redefining the residuals.

### A. Standardization of Coordinate Systems

The first step is to standardize the coordinate systems to the IMU frame. Define the incremental LiDAR state, which is the Lie algebra of ${}^w\mathbf{T}_l$, as ${}^w\mathbf{x}_l = ({}^w\boldsymbol{\phi}_l, {}^w\mathbf{t}_l) \in so(3) \times \mathbb{R}^3$. Similiarly, we define a hypothetical incremental IMU state ${}^w\mathbf{x}'_b = ({}^w\boldsymbol{\phi}'_b, {}^w\mathbf{t}'_b) \in so(3) \times \mathbb{R}^3$. Given the original Jacobian Matrix $\mathbf{J}$ of the LiDAR residual, the converted Jacobian matrix $\mathbf{J}'$ can be defined as

$$\mathbf{J}' = \frac{\partial \mathbf{r}_l}{\partial {}^w\mathbf{x}_l} \cdot \frac{\partial {}^w\mathbf{x}_l}{\partial {}^w\mathbf{x}'_b} = \mathbf{J} \cdot \frac{\partial {}^w\mathbf{x}_l}{\partial {}^w\mathbf{x}'_b}, \tag{8}$$

where $\frac{\partial {}^w\mathbf{x}_l}{\partial {}^w\mathbf{x}'_b}$ can be derived as (9), with $\mathcal{J}_l({}^w\boldsymbol{\phi}_l)$ being the right Jacobian of Lie algebra ${}^w\boldsymbol{\phi}_l$ and $({}^b\mathbf{t}_l)^\wedge$ being the skew-symmetric matrix of ${}^b\mathbf{t}_l$.

$$\frac{\partial {}^w\mathbf{x}_l}{\partial {}^w\mathbf{x}'_b} = \begin{bmatrix} \frac{\partial {}^w\boldsymbol{\phi}_l}{\partial {}^w\boldsymbol{\phi}'_b} & \frac{\partial {}^w\boldsymbol{\phi}_l}{\partial {}^w\mathbf{t}'_b} \\ \frac{\partial {}^w\mathbf{t}_l}{\partial {}^w\boldsymbol{\phi}'_b} & \frac{\partial {}^w\mathbf{t}_l}{\partial {}^w\mathbf{t}'_b} \end{bmatrix} = \begin{bmatrix} \mathcal{J}_l^{-1}({}^w\boldsymbol{\phi}_l){}^w\mathbf{R}'_b & 0 \\ -{}^w\mathbf{R}'_b({}^b\mathbf{t}_l)^\wedge & \mathbf{I} \end{bmatrix}. \tag{9}$$

### B. Standardization of State Update Methods

${}^w\mathbf{x}_l$ and ${}^w\mathbf{x}'_b$ mentioned above follows the update method $\oplus_1$, with $\ominus_1$ being its inverse operation:

$$[\mathbf{R} \mid \mathbf{t}] \oplus_1 \delta x_1 = [\mathbf{R}_1 \mid \mathbf{t}_1], \quad \delta x_1 = \begin{bmatrix} \delta w \\ \delta t \end{bmatrix},$$

$$[\mathbf{R}_1 \mid \mathbf{t}_1] = [\mathbf{R}\,\mathrm{Exp}(\delta w) \mid t + \delta t], \tag{10}$$

$$[\mathbf{R} \mid t] \ominus_1 [\mathbf{R}' \mid t'] = \begin{bmatrix} \log\left(\mathbf{R}^T\mathbf{R}'\right) \\ t' - t \end{bmatrix}. \tag{11}$$

However, residuals and Jacobians of the actual incremental IMU state ${}^w\mathbf{x}_b \in se(3)$ are defined with $\oplus_2$ under the right-multiply exponential map:

$$[\mathbf{R} \mid \mathbf{t}] \oplus_2 \delta x_2 = [\mathbf{R}_2 \mid \mathbf{t}_2], \quad \delta x_2 = \begin{bmatrix} \delta w \\ \delta v \end{bmatrix},$$

$$\begin{bmatrix} \mathbf{R}_2 & \mathbf{t}_2 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \mathrm{Exp}(\delta x_2). \tag{12}$$

After the standardization of coordinate systems, we have

$$f\left({}^w\mathbf{x}'_b \oplus_1 \delta x_1\right) \approx f({}^w\mathbf{x}'_b) + \mathbf{J}'\delta x_1. \tag{13}$$

Following TVL-SLAM [33], we have

$$f\left({}^w\mathbf{x}_b \oplus_2 \delta x_2\right) \approx f({}^w\mathbf{x}_b) + \mathbf{J}'\mathbf{J}_{12}\delta x_1, \tag{14}$$

where $\mathbf{J}_{12}$ can be computed as

$$\mathbf{J}_{12} = \frac{\partial \left[\mathbf{R} \mid \mathbf{t}\right] \ominus_1 \left(\left[\mathbf{R} \mid \mathbf{t}\right] \oplus_2 \delta x_2\right)}{\partial \delta x_2}$$

$$= \frac{\partial \begin{bmatrix} \log\left(\mathbf{R}^T\mathbf{R}\,\mathrm{Exp}(\delta w)\right) \\ \mathbf{R} \cdot \delta v + \mathbf{t} - \mathbf{t}' \end{bmatrix}}{\partial \delta x_2}$$

$$= \begin{bmatrix} \mathbf{J}_r(\log(\mathbf{R}^T\mathbf{R})) & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix}. \tag{15}$$

Finally, after the two-stage standardization, the Jacobian matrix of the LiDAR residual is achieved by $\mathbf{J_m} = \mathbf{J}'\mathbf{J}_{12}$.

## VII. Experiment and Results

This section tests the proposed method on both public and self-recorded datasets. The results are compared with several state-of-the-art multi-sensor SLAM/odometry systems. For accuracy evaluations, we focus on the final optimized keyframe poses. Since our system does not include a loop-closing module, to be fair, the loop-closing modules of all the comparison systems are turned off.

### A. Evaluation on Public Datasets

The evaluation on public datasets is conducted on KITTI [7]. KITTI dataset is collected on a car, equipped with a PointGray Flea2 grayscale camera (stereo), a Velodyne HDL-64E LiDAR, and an OXTS RT3003 which provides IMU and GPS measurements. Ground truth poses and extrinsic calibrations are also provided. Here we compare TC²LI-SLAM with state-of-the-art multi-sensor fusion SLAM algorithms, LVI-SAM [29], Fast-LIO2 [22], Fast-LIVO [36] and ORB-SLAM3 [3] in stereo-inertial mode (ORB-SLAM3-VI). To make FAST-LIVO work well on KITTI, we import the velodyne handler from FAST-LIO2 into its preprocess module. As for TC²L-SLAM, we compare it with A-LOAM [5] and ORB-SLAM3 in stereo-only mode (ORB-SLAM3-V). Finally, We evaluate the test results with the mean value and the root mean square error (RMSE) of the absolute pose error (APE) of the whole trajectory in comparison to the ground truth, computed with the EVO evaluation tool [42]. Notably, since ORB-SLAM3 and our systems carry randomness, we run them 5 times on each sequence and take the average result for evaluation.

We intended to use 00-10 odometry sequences in the KITTI dataset, but some of the sequences have a few interruptions in their IMU data, which demonstrates the necessity of implementing TC²L-SLAM. Thus, the evaluation of TC²LI-SLAM is conducted on intact sequences. As the results shown in Table I, TC²LI-SLAM performs better on all the sequences except seq. 04. Especially on seq. 09, it reduces the APE of ORB-SLAM3 by 20%. This is because seq. 09 is collected in residential areas, resulting in more reliable planar features provided by the buildings. Its poor performance on seq. 04 is most likely caused by the fact that the car is moving very fast on a highway, resulting in strong IMU constraints for a good enough optimization convergence. Thus, adding the LiDAR co-vision residual in this scenario might cause a deviation of the optimization result. In addition, the high-speed motion also causes FAST-LIVO to drift considerably on seq. 04 and 10, with fewer map points being tracked in the images.

TABLE I
APE OF SLAM SYSTEMS WITH IMU ON KITTI DATASET IN METERS

| Seq. | | 04 | 07 | 09 | 10 |
|---|---|---|---|---|---|
| FAST-LIO2 | Mean | 2.58 | 2.53 | 10.46 | 20.82 |
| | RMSE | 3.13 | 2.90 | 12.96 | 24.61 |
| ORB-SLAM3-VI | Mean | **0.45** | 2.40 | 4.87 | 4.00 |
| | RMSE | **0.48** | 2.90 | 5.56 | 4.59 |
| LVI-SAM | Mean | 3.04 | 2.36 | 10.32 | 11.76 |
| | RMSE | 3.99 | 2.92 | 11.78 | 14.29 |
| FAST-LIVO | Mean | 9.68 | 2.64 | 9.10 | 23.84 |
| | RMSE | 9.76 | 2.77 | 10.85 | 27.38 |
| TC$^2$LI-SLAM | Mean | 0.51 | **2.26** | **3.88** | **3.62** |
| | RMSE | 0.54 | **2.75** | **4.15** | **4.16** |

TABLE II
APE OF SLAM SYSTEMS WITHOUT IMU ON KITTI DATASET IN METERS

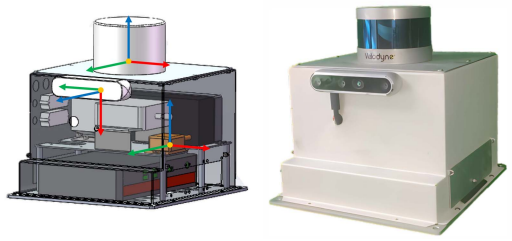
| Seq. | A-LOAM | | ORB-SLAM3-V | | TC$^2$L-SLAM | |
|---|---|---|---|---|---|---|
| | Mean | RMSE | Mean | RMSE | Mean | RMSE |
| 00 | **8.43** | 9.28 | 8.80 | 9.24 | 8.57 | **8.99** |
| 01 | 117.44 | 144.23 | 38.34 | 43.22 | **34.31** | **37.46** |
| 02 | 123.08 | 159.80 | 8.12 | 10.23 | **7.80** | **9.95** |
| 03 | 5.85 | 7.20 | 2.81 | 3.16 | **2.37** | **2.69** |
| 04 | 2.62 | 3.54 | 0.48 | 0.53 | **0.44** | **0.47** |
| 05 | 4.57 | 5.33 | 3.70 | 4.16 | **3.33** | **3.76** |
| 06 | 3.12 | 3.84 | 3.18 | 3.49 | **2.86** | **3.07** |
| 07 | 1.72 | **1.88** | 2.53 | 3.11 | **1.68** | 1.98 |
| 08 | 18.63 | 19.52 | **11.89** | 12.32 | 11.92 | **12.27** |
| 09 | 6.22 | 7.35 | 5.95 | 6.59 | **4.70** | **5.37** |
| 10 | 10.50 | 12.21 | 3.89 | **4.57** | **3.84** | 4.60 |

We evaluate the performance of our degenerated camera-LiDAR SLAM system TC$^2$L-SLAM on all 11 sequences. The results are shown in Table II. Our system performs better on all the sequences, except for being slightly worse on a few sequences. Notably, On seq. 07, TC$^2$L-SLAM, although not entirely the best, reaches almost the same level as A-LOAM, reducing the APE of ORB-SLAM3 by 33%. This perfectly demonstrates that when LiDAR measurements provide higher quality constraints than the visual features, our tightly-coupled bundle adjustment method can compensate for the lack of visual constraints by leveraging LiDAR co-vision constraints, and thus achieve a higher localization accuracy. It's worth mentioning that, on seq. 04, TC$^2$L-SLAM performs even better than ORB-SLAM3-VI. Moreover, we notice that on the seq. 07, SLAM systems incorporating IMU show worse performance. This should be explained by the possible flawed inertial measurements in our seq. 07 data. These strongly demonstrate the need for TC$^2$L-SLAM.

### B. Evaluation on Self-Recorded Datasets

The self-recorded dataset contains two sequences collected on a vehicle-like ground robot with a sensor unit consisting of an Intel RealSense D455 camera, an Xsens MTi-300 IMU and a Velodyne VLP-16 LiDAR, with details shown in Fig. 3. An Intel next unit of computing (NUC) and a built-in battery are incorporated inside the unit for sensor driving and power supply. In addition, since there's no ground truth, we made the start and end of each sequence overlap for further evaluation. Fig. 4 shows the satellite image of the atrium where we recorded the dataset.

To better analyze the improvement achieved by adding the LiDAR residual in a bundle adjustment module, we compare the proposed systems with our baseline ORB-SLAM3. Fig. 5



(a) (b)

Fig. 3. (a) The computer-aided design (CAD) model of our sensor unit, with the body coordinate system of each sensor plotted on them. (b) The real appearance of the sensor unit.

Fig. 4. Satellite image of the atrium where the self-recorded dataset sequences are collected. The white line is the projection of the frame trajectory from TC$^2$LI-SLAM in sequence 2.

TABLE III
LOOP ERROR OF SLAM SYSTEMS ON SELF-RECORDED DATASET

| Seq. | 1 | 2 |
|---|---|---|
| ORB-SLAM3-V | 0.61% | 0.44% |
| TC$^2$L-SLAM | **0.51%** | **0.33%** |
| ORB-SLAM3-VI | 0.87% | 0.52% |
| TC$^2$LI-SLAM | **0.66%** | **0.31%** |

shows the bird-eye-viewed trajectory of our TC$^2$LI-SLAM and TC$^2$L-SLAM compared to ORB-SLAM3, where 5(a)–(b) are the results on the single-looped sequence and 5(c)–(d) on the longer sequence. Here we calculate the loop error as an evaluation criterion, which is the ratio of the start-to-end Euclidean distance to the length of the trajectory. The length of the trajectory is the shortest trajectory length in each set. The results are listed in Table III.

As shown in Fig. 5 and Table III, our systems loop back better and reduce the loop error by 25% on average, which indicates that the accumulated pose error is extensively reduced with the additional LiDAR residual in the local bundle adjustment module. Notably, on both sequences, TC$^2$L-SLAM performs overall the best, being equally well as TC$^2$LI-SLAM on seq. 2, and ORB-SLAM3-VI performs slightly worse than ORB-SLAM3-V. These are caused by the fact that our vehicle for data collection moves too slowly with a maximum speed of 1.5 m/s, which creates unreliable inertial constraints in the pose optimization. This further proves the necessity of implementing TC$^2$L-SLAM.

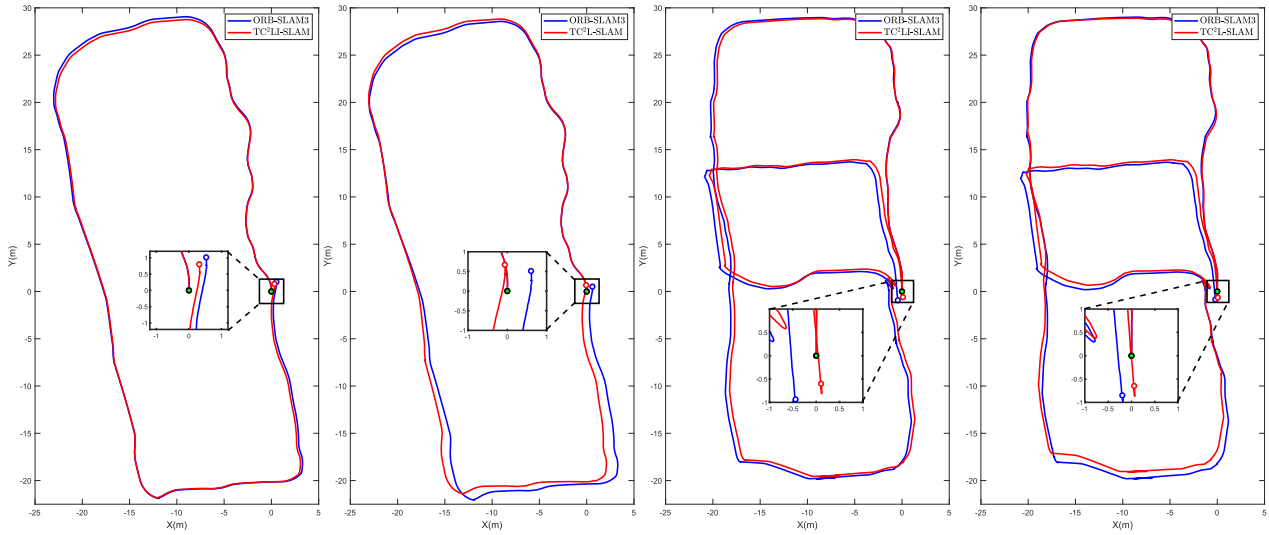



Fig. 5. Trajectories of TC$^2$L(I)-SLAM and ORB-SLAM3-V(I) on our self-recorded dataset sequences. (a) and (b) are from sequence 1, showing the optimized trajectory of TC$^2$LI-SLAM and TC$^2$L-SLAM (red) compared to ORB-SLAM3-V(I) (blue). (c) and (d) are from the longer sequence 2. The solid green circles mark the start points, while the hollow blue and red circles mark the end points of the optimized trajectories.

TABLE IV
APE OF SLAM SYSTEMS FOR ABLATION STUDY IN METERS

| Baseline | | TC$^2$LI-SLAM | | TC$^2$L-SLAM | |
|---|---|---|---|---|---|
| Seq. | | 07 | 09 | 07 | 09 |
| B | Mean | 2.43 | 4.60 | 2.63 | 6.59 |
| | RMSE | 2.96 | 5.28 | 3.26 | 8.77 |
| B+C | Mean | 2.45 | 4.92 | 2.47 | 5.43 |
| | RMSE | 3.00 | 5.45 | 3.06 | 6.18 |
| B+S | Mean | 2.38 | 4.42 | 2.41 | 5.01 |
| | RMSE | 2.89 | 4.96 | 2.96 | 5.78 |
| B+C+S | Mean | **2.26** | **3.88** | **1.68** | **4.70** |
| | RMSE | **2.75** | **4.15** | **1.98** | **5.37** |

TABLE V
AVERAGE PER-FRAME TIME CONSUMPTION OF FRONT END

| | KITTI sequences | Self-recorded sequences |
|---|---|---|
| TC$^2$LI-SLAM | 41.1 ms | 28.4 ms |
| TC$^2$L-SLAM | 35.1 ms | 25.5 ms |

### C. Ablation Study

Here we conduct an ablation study to explore the effects of the standardization of coordinate systems (C) and standardization of state update methods (S) in the cross-modal residual standardization method. We denote our systems without all the standardization methods as the Baseline (B), We evaluate the effect of standardization methods by adding C and S respectively and simultaneously to B (B + C, B + S, B + C + S) under the same sensor configuration. The experiment is conducted on seq. 07 and 09 of the KITTI dataset. Table IV shows the APE results, which suggest that both processes in our cross-modal residual standardization method play essential roles in performance improvement.

### D. Evaluation of Runtime Performance

To prove the real-time efficiency of our proposed method, we conduct a runtime evaluation for both TC$^2$LI-SLAM and TC$^2$L-SLAM on a desktop PC (with Intel i7-12700H CPU and 16 GB RAM). Here we focus on the time consumption of the front end, which includes the data synchronization process. We record the average time that starts with every image input and ends after stage 2 of data synchronization. Note that the time consumption of each stage of data synchronization is roughly 0.07 milliseconds on average, depending on the number of planar feature points.

We run our proposed systems on all the available KITTI sequences and self-recorded sequences, the resulting average time usage is shown in Table V. The time consumption on the sef-recorded dataset is less than that on KITTI because we use a sparser LiDAR. Due to ikd-Tree's periodical rebuilding, the actual time consumption may spike sometimes, but it won't affect the overall efficiency. In conclusion, our system can be considered real-time based on the fact that KITTI sequences have the input frequency of both camera images and LiDAR scans at 10 Hz, while in self-recorded sequences, the input frequency of the camera images is around 30 Hz.

## VIII. CONCLUSION AND FUTURE WORKS

In this letter, we propose two novel and accurate real-time SLAM systems, TC$^2$LI-SLAM and TC$^2$L-SLAM, that fuse multi-sensor information tightly in a keyframe-based bundle adjustment module. A keyframe-oriented data synchronization algorithm is designed, correlating LiDAR and camera measurements while enabling real-time efficiency of the front end. Furthermore, a cross-modal residual standardization method is proposed, bridging the gap in residual definitions between different sensors. The proposed systems are tested on both public and self-recorded datasets, illustrating their real-time efficiency

and improved accuracy. In future works, we will investigate adaptive constraint weight adjustment and feature selection methods to further improve the performance and robustness of our systems, and incorporate online system switching based on sensor degradation judgment to cope with intricate or noisy environments.

## REFERENCES


[1] M. Zaffar, S. Ehsan, R. Stolkin, and K. M. Maier, "Sensors, SLAM and long-term autonomy: A review," in *Proc. IEEE NASA/ESA Conf. Adapt. Hardw. Syst.*, Edinburgh, U.K., 2018, pp. 285–290.

[2] Y. Chen, Y. Zhou, Q. Lv, and K. K. Deveerasetty, "A review of V-SLAM," in *Proc. IEEE Int. Conf. Inf. Automat.*, 2018, pp. 603–608.

[3] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, Dec. 2021.

[4] R. Gomez-Ojeda, F.-A. Moreno, D. Zuñiga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "PL-SLAM: A stereo SLAM system through the combination of points and line segments," *IEEE Trans. Robot.*, vol. 35, no. 3, pp. 734–746, Jun. 2019.

[5] J. Zhang and S. Singh, "Low-drift and real-time LiDAR odometry and mapping," *Auton. Robots*, vol. 41, no. 2, pp. 401–416, 2017.

[6] Q. Zou, Q. Sun, L. Chen, B. Nie, and Q. Li, "A comparative analysis of LiDAR SLAM-based indoor navigation for autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6907–6921, Jul. 2022.

[7] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[8] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 834–849.

[9] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[10] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, Oct. 2017.

[11] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, Mar. 2018.

[12] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO :Fast semi-direct monocular visual odometry," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 15–22.

[13] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robot.*, vol. 33, no. 2, pp. 249–265, Apr. 2017.

[14] M. Li and A. I. Mourikis, "High-precision, consistent EKF-based visual-inertial odometry," *Int. J. Robot. Res.*, vol. 32, no. 6, pp. 690–711, 2013.

[15] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, Feb. 2017.

[16] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.

[17] T. Schneider et al., "Maplab: An open framework for research in visual-inertial mapping and localization," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1418–1425, Jul. 2018.

[18] J. Jiang, J. Yuan, X. Zhang, and X. Zhang, "DVIO: An optimization-based tightly coupled direct visual-inertial odometry," *IEEE Trans. Ind. Electron.*, vol. 68, no. 11, pp. 11212–11222, Nov. 2021.

[19] Z. Song, X. Zhang, T. Li, S. Zhang, Y. Wang, and J. Yuan, "IR-VIO: Illumination-robust visual-inertial odometry based on adaptive weighting algorithm with two-layer confidence maximization," *IEEE/ASME Trans. Mechatronics*, vol. 28, no. 4, pp. 1920–1929, Aug. 2023.

[20] R. Li, X. Zhang, S. Zhang, J. Yuan, H. Liu, and S. Wu, "BA-LIOM: Tightly coupled laser-inertial odometry and mapping with bundle adjustment," *Robotica*, vol. 42, no. 3, pp. 684–700, 2024.

[21] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.

[22] W. Xu and F. Zhang, "FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.

[23] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.

[24] J. Zhang and S. Singh, "Visual-LiDAR odometry and mapping: Low-drift, robust, and fast," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2174–2181.

[25] J. Graeter, A. Wilczynski, and M. Lauer, "LIMO: LiDAR-monocular visual odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018,pp., 7872–7879.

[26] J. Zhang, M. Kaess, and S. Singh, "Real-time depth enhanced monocular odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 4973–4980.

[27] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "CamVox: A low-cost and accurate LiDAR-assisted visual SLAM system," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5049–5055.

[28] Z. Hu, J. Yuan, Y. Gao, B. Wang, and X. Zhang, "NALO-VOM: Navigation oriented LiDAR-guided monocular visual odometry and mapping for unmanned ground vehicles," *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 2612–2623, Jan. 2024.

[29] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled LiDAR-visual-inertial odometry via smoothing and mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 5692–5698.

[30] Y. Jia et al., "Lvio-fusion: A self-adaptive multi-sensor fusion SLAM framework using actor-critic method," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 286–293.

[31] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "LIC-Fusion: LiDAR-inertial-camera odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 5848–5854.

[32] X. Zuo et al., "LIC-Fusion 2.0: LiDAR-inertial-camera odometry with sliding-window plane-feature tracking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5112–5119.

[33] C.-C. Chou and C.-F. Chou, "Efficient and accurate tightly-coupled visual-LiDAR SLAM," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14509–14523, Sep. 2022.

[34] J. Lin, C. Zheng, W. Xu, and F. Zhang, "R$^2$LIVE: A robust, real-time, LiDAR-inertial-visual tightly-coupled state estimator and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7469–7476, Oct. 2021.

[35] J. Lin and F. Zhang, "R$^3$LIVE: A robust, real-time, RGB-colored, LiDAR-inertial-visual tightly-coupled state estimation and mapping package," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 10672–10678.

[36] C. Zheng, Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang, "FAST-LIVO: Fast and tightly-coupled sparse-direct LiDAR-inertial-visual odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 4003–4009.

[37] Y. Wang and H. Ma, "mVIL-fusion: Monocular visual-inertial-LiDAR simultaneous localization and mapping in challenging environments," *IEEE Robot. Automat. Lett.*, vol. 8, no. 2, pp. 504–511, Feb. 2023.

[38] S. Isaacson, P.-C. Kung, M. Ramanagopal, R. Vasudevan, and K. A. Skinner, "LONER: LiDAR only neural representations for real-time SLAM," *IEEE Robot. Automat. Lett.*, vol. 8, no. 12, pp. 8042–8049, Dec. 2023.

[39] T. Fu, S. Su, Y. Lu, and C. Wang, "iSLAM: Imperative SLAM," *IEEE Robot. Automat. Lett.*, vol. 9, no. 5, pp. 4607–4614, May 2024.

[40] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G $^2$ o: A general framework for graph optimization," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 3607–3613.

[41] Z. Liu and F. Zhang, "BALM: Bundle adjustment for LiDAR mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3184–3191, Apr. 2021.

[42] M. Grupp, "EVO: Python package for the evaluation of odometry and SLAM," 2017. [Online]. Available: https://github.com/MichaelGrupp/evo