IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

FAST-LIEO: Fast and Real-Time LiDAR-Inertial-Event-Visual Odometry

Zirui Wang, *Graduate Student Member, IEEE*, Yangtao Ge, *Member, IEEE*, Kewei Dong, I-Ming Chen, *Fellow, IEEE*, and Jing Wu

***Abstract*—Unlike a standard camera that relies on exposure to obtain output frame by frame, an event camera only outputs an event when the change of brightness intensity in a pixel exceeds a threshold, and the outputs of different pixels are independent to each other. Benefited from its bio-inspired design, event camera has the advantages of low latency and high dynamic range. The researches on multi-sensor fusion with event camera are few so far. In this paper, we propose FAST-LIEO, a framework for fast and real-time LiDAR-inertial-event odometry. The framework tightly fuses LiDAR and event camera measurements without any feature extraction or matching. Besides, our system supports both LIEO and LIEVO (extended with RGB camera fusion). We design a novel EIO subsystem for LiDAR-event fusion. The EIO subsystem maintains a semi-dense event map and estimates the state by aligning the event representation to map. The semi-dense event map is built from LiDAR points by utilizing the edge information and temporal information provided by event representations. Besides testing our method on public benchmark dataset, we also collected real-world data by utilizing our sensor suite and conducted experiments on our self-captured dataset. The experiment results show the high robustness and accuracy of our method in challenging conditions with high real-time ability. To the best of our knowledge, our FAST-LIEO is the first system that can tightly fuse LiDAR, IMU, event camera and standard camera measurements in simultaneously localization and mapping.**

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

I. Introduction

THE output image frames of traditional RGB camera need exposure time, and due to this reason overexposure or underexposure in high dynamic range (HDR) scenes and motion blur are unavoidable. An event camera outputs event stream of pixel-level brightness changes with low latency and high temporal resolution (1 μs), and the dynamic range of the event camera (120 dB) is higher than standard RGB camera (60 dB) [1], [2], [3]. Thus there is no exposure issue or motion blur for event stream. The event camera has large potential in state estimation in challenging conditions like weak illumination and aggressive motion.

Due to the unique event stream output, the visual odometry and visual SLAM (simultaneously localization and mapping) systems designed for standard cameras are not compatible with event cameras. Thus, new methods need to be designed for utilizing event cameras in state estimation. Most research on event camera in SLAM is related to event-based VO (visual odometry) [4], [5], [6] or VIO (visual-inertial odometry) [7], [8], [9], [10], [11], and few research is about multi-sensor fusion with event cameras. Some LiDAR-inertial-visual fusion methods [12], [13], [14] have been proposed to fuse standard camera measurements in LIVO (LiDAR-inertial-visual odometry) to overcome the LiDAR degeneration conditions in LiDAR-based SLAM/odometry [15], [16], [17], the visual measurements could fail in challenging conditions like HDR scenes and image motion blur caused by aggressive motion. In this paper, we focus on the multi-sensor fusion with event camera, and propose FAST-LIEO, a fast LiDAR-inertial-event-visual odometry that can integrate the advantages of different sensors. To utilize different sensors with data association, we do not simply use an optimization-base approach to fuse previously existing odometry in the form of the factor graph. A novel EIO (event-inerial odometry) subsystem is designed based on LiDAR-event data association for event measurements in the multi-sensor fusion framework. The contributions of this paper are as follows:

1) We propose a multi-sensor fusion odometry framework that can tightly fuse LiDAR, IMU, event camera and standard camera measurements without any feature extraction or matching. The whole system consists of three subsystem: LIO, EIO and VIO subsystem. Our framework supports both LIEO (LiDAR-inertial-event odometry) and LIEVO (LiDAR-inertial-event-visual odometry). The fusion of standard camera measurements is optional.
2) A novel EIO subsystem is designed for LiDAR-event fusion in the framework. The event map construction process fully utilizes the data association between events and LiDAR points to build a semi-dense map representing the 3D edges in scenes. And the EIO state estimation

Received 19 September 2024; accepted 28 November 2024. Date of publication 26 December 2024; date of current version 9 January 2025. This article was recommended for publication by Associate Editor J. Zhang and Editor J. Civera upon evaluation of the reviewers' comments. This work was supported by the National Science Foundation of China (NSFC) under Grant 62203205 and Grant U1913603. *(Corresponding author: Jing Wu.)*

Zirui Wang, Kewei Dong, and Jing Wu are with the Department of Mechanical and Energy Engineering, Southern University of Science and Technology (SUSTech), Shenzhen 518055, China (e-mail: wuj@sustech.edu.cn).

Yangtao Ge is with the Department of Mechanical and Energy Engineering, Southern University of Science and Technology (SUSTech), Shenzhen 518055, China, and also with the Shenzhen Institute of Artificial Intelligence and Robotics for Society (AIRS), Shenzhen 518055, China.

I-Ming Chen is with the School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore 639798.

The source code of FAST-LIEO and our dataset are available at: https://github.com/wsjpla/FAST-LIEO.

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3522843, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3522843

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

is conducted by the alignment of event representation to map.

3) To evaluate our proposed method, we utilize both public benchmark dataset and our self-captured LIE dataset. The experiment results show that our method can achieve high robustness and accuracy across different platforms and challenging conditions.

The following paper is organized as follows: Section II discusses the related work of our FAST-LIEO. Section III introduces the proposed framework in detail. Section IV demonstrates the experiment results on both public dataset and our LIE dataset. Section V gives conclusions.

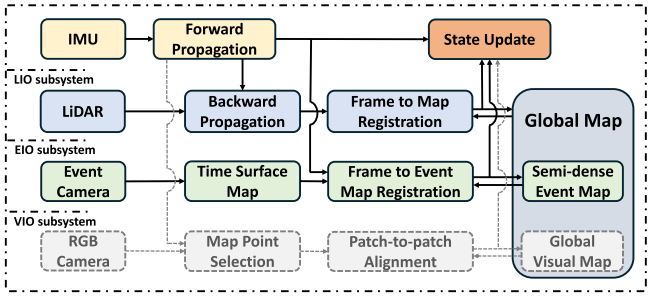

Fig. 1. The framework of FAST-LIEO. The system supports both LIEO and LIEVO. The fusion of standard RGB camera measurements is optional.

II. Related Work

Considering that currently researches on multi-sensor fusion with event camera are few, the related works are introduced in two parts: LiDAR-inertial-visual fusion SLAM/odometry and event-based SLAM/odometry.

A. LiDAR-Inertial-Visual Fusion SLAM/odometry

In recent years, several successful LiDAR-Inertial-Visual fusion methods are proposed to overcome the sensor degeneration conditions and combine the advantages of different sensors. [18] achieve LiDAR, IMU, standard camera measurements fusion in a coarse-to-fine approach. IMU perdition is refined by visual measurements and LiDAR scan matching. LIC-Fusion [19] is a feature-based method that tightly fuses LiDAR-inertial-camera measurements in a multi-state constraint Kalman filter. LIC-Fusion 2.0 [20] extends the previous version with point cloud plane feature detection and tracking in a sliding-window filter. LIV-SAM [14] is also a feature-based fusion method. The tightly-coupled LVIO maintains a factor graph and has two subsystems: LIS(LiDAR-inertial system) and VIS(visual-inertial system), which are respectively adopted from LIO-SAM [16] and VINS-Mono [21]. It also introduces failure detection in its two subsystems to achieve higher robustness. R2LIVE [22] builds its two subsystems, LIO and VIO, based on FAST-LIO [23] and VINS-Mono [21], and the state estimation is achieved by ESIKF (error state iterated Kalman filter) and factor graph optimization. R3LIVE [12] designs a new VIO subsystem that renders map points and estimates the image frame state by frame-to-map photometric update. It builds a dense 3D colored map as the output of global map. FAST-LIVO [13] also has two subsystems: LIO and VIO, and the LIO part is built based on FAST-LIO2 [15]. The VIO utilizes a patch-to-patch coarse-to-fine update approach to achieve high efficiency.

B. Event-Based SLAM/odometry

The research on event-based SLAM/odometry has been enlarged since different event cameras (e.g. DAVIS346 and DVXplorer) became commercially available. EVO [4] achieves parallel tracking and mapping utilizing only event camera. It obtains event frames by directly accumulating a fixed number of events, and the inverse compositional LucasKanade (LK) method is used for pose tracking. EVO initializes the system through a bootstrap method by assuming that the camera moves in a plane scene parallel to the image plane during the first few seconds of system initialization. ESVO [6] is the first event-based stereo odometry, which achieves stereo event-based depth estimation for mapping. The estimated inverse depth maps are refined by fusion. The camera pose tracking is calculated by registering the time surfaces to 3D reconstructed map. DEVO [5] detects Canny edge information in time surfaces and utilizes a depth camera to recover the depths of event camera pixels. Its camera tracking strategy is similar to ESVO. EVIO [11] is a feature-based method using extended Kalman filter to fuse feature tracking and IMU. Ultimate SLAM [7] is the first SLAM system that tightly fuses event camera, standard camera and IMU. It is a feature-based method using nonlinear optimization. The features are extracted and tracked in event frames with motion compensation. [8] fuses event measurements and IMU using pose graph optimization with sliding window. The features are extracted in raw events and then tracked in time surfaces with polarity. ESVIO [9] is a stereo visual-inertial odometry using sliding window optimization based on feature detection and tracking. The framework also supports the fusion of stereo standard camera images. PL-EVIO [10] utilizes both point and line features in event-based VIO. Its pipeline also supports RGB images as optional input.

In FAST-LIEO, our EIO subsystem does not rely on any feature detection or matching. We utilize the temporal and spatial information in event representations (time surfaces) with the aid of LiDAR points to achieve high efficient EIO estimation.

III. System Description

Fig. 1 shows the overview of our FAST-LIEO framework. It can support both LiDAR-Inertial-Event fusion (LIEO) and LiDAR-Inertial-Event-Visual fusion (LIEVO). The system contains three subsystems: the LIO, the EIO and the VIO subsystem. The LIO subsystem builds a dense map and estimates the state by minimizing scan-to-map residual. The EIO subsystem utilizes time surface maps to select proper map points from LiDAR points and build a semi-dense event map. The state estimation of EIO is computed by the alignment of event representation to map. The VIO subsystem updates the state by patch-to-patch sparse-direct alignment [13] and maintain a sparse VIO map.

The above subsystems are tightly coupled by an ESIKF (error state iterated Kalman filter). The state vector $\mathbf{x} \in \mathrm{R}^{18}$ is defined

as:

$$\mathbf{x} \triangleq \begin{bmatrix} {}^{G}\mathbf{R}_I^T & {}^{G}\mathbf{p}_I^T & {}^{G}\mathbf{v}_I^T & \mathbf{b}_\mathbf{g}^T & \mathbf{b}_\mathbf{a}^T & {}^{G}\mathbf{g}^T \end{bmatrix}^T \tag{1}$$

where $({}^{G}\mathbf{R}_I^T, {}^{G}\mathbf{p}_I^T)$ is the pose of IMU in global frame, ${}^{G}\mathbf{R}_I^T$ is the rotation part, ${}^{G}\mathbf{p}_I^T$ is the translation part, ${}^{G}\mathbf{v}_I^T$ is the IMU linear velocity in global frame, $\mathbf{b}_\mathbf{g}^T$ is gyroscope bias, $\mathbf{b}_\mathbf{a}^T$ is accelerometer bias and ${}^{G}\mathbf{g}^T$ is the gravitational acceleration in global frame.

To express the process of the state update, encapsulated "box-plus" $\boxplus$ and "boxminus" $\boxminus$ operations on manifold [23], [24] are used. For $\mathcal{M} = SO(3) \times \mathsf{R}^n$, we have:

$$\begin{bmatrix} \mathbf{R} \\ \mathbf{a} \end{bmatrix} \boxplus \begin{bmatrix} \mathbf{r} \\ \mathbf{b} \end{bmatrix} \triangleq \begin{bmatrix} \mathbf{R} \cdot \mathrm{Exp}(\mathbf{r}) \\ \mathbf{a} + \mathbf{b} \end{bmatrix} \tag{2}$$

$$\begin{bmatrix} \mathbf{R}_1 \\ \mathbf{a} \end{bmatrix} \boxminus \begin{bmatrix} \mathbf{R}_2 \\ \mathbf{b} \end{bmatrix} \triangleq \begin{bmatrix} \mathrm{Log}\left(\mathbf{R}_2^T \mathbf{R}_1\right) \\ \mathbf{a} - \mathbf{b} \end{bmatrix} \tag{3}$$

in which $\mathbf{r} \in \mathsf{R}^3$ and $\mathbf{a}, \mathbf{b} \in \mathsf{R}^n$. $\mathrm{Exp}(\cdot)$ is the exponential map from $so(3)$ to $SO(3)$, namely from rotation vector to rotation matrix, while $\log(\cdot)$ is the inverse map.

A. LiDAR-Inertial Odometry Subsystem

Once receiving a new LiDAR scan with enough IMU measurements, the in-scan motion blur of LiDAR points is compensated by backward propagation [15]. The accumulated LiDAR points ${}^{L}\mathbf{p}_i$ in the new scan are all compensated to the scan-end time t_k. The undistorted LiDAR points $\{{}^{L}\mathbf{p}_i^u\}$ are then registered to the map by point-to-plane registration. For each LiDAR point ${}^{L}\mathbf{p}_i^u$, five nearest map points are searched to form a plane patch with center point ${}^{G}\mathbf{q}_i$ and normal vector ${}^{G}\mathbf{u}_i^T$. The LiDAR measurements residual is defined as:

$$\mathbf{r}_{L_k}\left({}^{L}\mathbf{p}_i^u\right) = {}^{G}\mathbf{u}_i^T \left({}^{G}\mathbf{T}_{I_k}\, {}^{I}\mathbf{T}_L\, {}^{L}\mathbf{p}_i^u - {}^{G}\mathbf{q}_i\right) \tag{4}$$

where ${}^{G}\mathbf{T}_{I_k}$ is the transformation from IMU frame to global frame at time t_k, and ${}^{I}\mathbf{T}_L$ is the extrinsic between LiDAR and camera. The maximum a posteriori (MAP) estimation for this LiDAR scan is:

$$\min_{\mathbf{x}_k \in \mathcal{M}} \left(\|\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k\|^2_{\hat{\mathbf{P}}_k} + \sum_{i=1}^{m_l} \left\|\mathbf{r}_{L_k}\left({}^{L}\mathbf{p}_i^u\right)\right\|^2_{\mathbf{\Sigma}_l} \right) \tag{5}$$

where $\|\mathbf{x}\|^2_{\mathbf{\Sigma}} = \mathbf{x}^T \mathbf{\Sigma}^{-1} \mathbf{x}$, $\mathbf{\Sigma}_l$ is the LiDAR measurement covariance, $\mathbf{x}_k$ is the true state vector at t_k, $\hat{\mathbf{x}}_k$ and $\hat{\mathbf{P}}_k$ are the predicted state and covariance by IMU forward propagation. The detailed propagation derivation is in [15]. The optimization is solved by ESIKF, which is equivalent to Gauss-Newton approach [25].

After residual computation convergence, the LiDAR points are fed to the global map on *ikd-tree* [26]. Benefit from the high efficient data structure, the above nearest point search and map update process can achieve high real-time ability.

B. Event-Inertial Odometry Subsystem

1) Event Representation: We build time surface (TS) maps for event representation. Time surface map is a 2D map that can be considered as a single channel image whose pixel value is related to the most recent event trigger time. We use an exponential decay kernel [27] to build time surface maps without polarity. The TS $\mathcal{T}$ without polarity at time t_k is defined as:

$$\mathcal{T}_k\left(\mathbf{u}, t_k\right) \triangleq \exp\left(-\frac{t_k - t_{\text{last}}\left(\mathbf{u}\right)}{\eta}\right) \tag{6}$$

where $\mathbf{u}$ is the 2D pixel coordinate, $t_{\text{last}}(\mathbf{u})$ is the the most recent event trigger time at this pixel before t_k, and η is the decay time. In this paper we choose $\eta = 30$ ms as a compromise between accumulated noise and temporal information preservation. For image processing, we build negative TS without polarity and remap the pixel value to [0, 255] as TS frame. The negative TS without polarity is defined as:

$$\mathcal{T}_k^n\left(\mathbf{u}, t_k\right) \triangleq 255.0 \cdot \left(1 - \mathcal{T}_k\left(\mathbf{u}\right)\right) \tag{7}$$

The pixel value, which indicates the event trigger time, is used in the event map update process. Our system generates TS frames at a fixed rate. Before generating a new TS frame, if the number of newly coming events is lower than a threshold, which indicates zero velocity or low-speed motion, the TS frame generation and following calculation will be skipped. In this case, the state estimation relies only on the LIO subsystem so that the noise events will not affect the EIO estimation and the event map maintenance.

2) EIO Estimation and Semi-Dense Event Map: We maintain a semi-dense event map representing the 3D edges in scenes and the semi-dense map is a subset of the dense LIO map. For a newly built TS frame $\mathcal{T}_k^n$ at time t_k, we project the event map onto TS plane and choose the map points within event camera FoV as valid local map points. In the local map point selection process, we use the newest LiDAR scan to check the map point depth values and exclude map points with abnormal depth values. If the pixel depth provided by event map point has large difference with the depth given by the newest LiDAR scan, the map point will not be added in the local event map.

The EIO estimation is achieved by frame-to-map alignment after building local event map. The TS provides rich edge information, so the frame-to-map alignment can be conducted directly without any feature extraction or matching. The TS also provides information about the history of past triggered event within decay time, therefore the TS frame is not only related to the camera pose and surrounding environments, but also related to the camera motion, which helps the convergence of frame-to-map alignment. For each valid map point ${}^{G}\mathbf{p}_j^e$ in local map, the residual is:

$$\mathbf{r}_{E_k}\left({}^{G}\mathbf{p}_j^e\right) = \mathcal{T}_k^n\left(\boldsymbol{\pi}_e\left({}^{E}\mathbf{T}_I \cdot {}^{G}\mathbf{T}_{I_k}^{-1} \cdot {}^{G}\mathbf{p}_j^e\right), t_k\right) \tag{8}$$

where ${}^{G}\mathbf{T}_{I_k}$ is the transformation from IMU frame to global frame at current TS frame time t_k, ${}^{E}\mathbf{T}_I$ is the extrinsic between IMU and event camera, $\pi_e(\cdot)$ projects the local event map point onto TS image plane. The maximum a posteriori (MAP) estimation for the event measurement is:

$$\min_{\mathbf{x}_k \in \mathcal{M}} \left(\|\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k\|^2_{\hat{\mathbf{P}}_k} + \sum_{j=1}^{m_e} \left\|\mathbf{r}_{E_k}\left({}^{G}\mathbf{p}_j^e\right)\right\|^2_{\mathbf{\Sigma}_e} \right) \tag{9}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

where $\mathbf{\Sigma}_e$ is the event measurement weight. For the ESIKF process, let $\hat{\mathbf{x}}_k^\kappa$ be the estimated state at the κ-th iteration, $\mathbf{P}$ be the state covariance and

$$\delta\hat{\mathbf{x}}_k^\kappa = \begin{bmatrix}\delta\boldsymbol{\theta}^T & \delta{}^G\mathbf{p}_I^T & \delta{}^G\mathbf{v}_I^T & \delta\mathbf{b}_{\boldsymbol{\omega}}^T & \delta\mathbf{b}_{\mathbf{a}}^T & \delta{}^G\mathbf{g}^T\end{bmatrix}^T \tag{10}$$

$$\mathbf{z}_k^\kappa = \left[\mathbf{r}_{E_k}({}^G\mathbf{p}_1^e), \ldots, \mathbf{r}_{E_k}({}^G\mathbf{p}_{m_e}^e)\right]^T \tag{11}$$

$$\mathbf{H} = \begin{bmatrix}\mathbf{H}_1^{\kappa^T}, \ldots, \mathbf{H}_{m_e}^{\kappa^T}\end{bmatrix}^T. \tag{12}$$

$\mathbf{H}_j^\kappa$ is defined as:

$$\begin{aligned}\mathbf{H}_j^\kappa &= \left.\frac{\partial\mathbf{r}_{E_k}({}^G\mathbf{p}_j^e)}{\partial\delta\hat{\mathbf{x}}_k^\kappa}\right|_{\delta\hat{\mathbf{x}}_k=\mathbf{0}} \\ &= \frac{\partial\mathcal{T}_k^n}{\partial\mathbf{u}}\cdot\frac{\partial\mathbf{u}}{\partial{}^E\mathbf{p}_j^e}\cdot\frac{\partial{}^E\mathbf{p}_j^e}{\partial\delta\hat{\mathbf{x}}_k^\kappa}\end{aligned} \tag{13}$$

where ${}^E\mathbf{p}_j^e$ is the event map point in event camera frame, $\frac{\partial\mathcal{T}_k^n}{\partial\mathbf{u}}$ is the TS image gradient at the pixel corresponding to the map point. Let ${}^E\mathbf{p}_j^e = \begin{bmatrix}X_j & Y_j & Z_j\end{bmatrix}^T$, then

$$\frac{\partial\mathbf{u}}{\partial{}^E\mathbf{p}_j^e} = \begin{bmatrix}\frac{f_x}{Z_j} & 0 & -\frac{f_x X_j}{Z_j^2} \\ 0 & \frac{f_y}{Z_j} & -\frac{f_y Y_j}{Z_j^2}\end{bmatrix} \tag{14}$$

$$\frac{\partial{}^E\mathbf{p}_j^e}{\partial\delta\hat{\mathbf{x}}_k^\kappa} = \begin{bmatrix}\frac{\partial{}^E\mathbf{p}_j^e}{\partial\delta\theta} & \frac{\partial{}^E\mathbf{p}_j^e}{\partial\delta{}^Gp_I} & \mathbf{0}_{3\times12}\end{bmatrix} \tag{15}$$

and

$$\frac{\partial{}^E\mathbf{p}_j^e}{\partial\delta\theta} = \left[{}^E\mathbf{p}_j^e\right]_\times\cdot{}^E\mathbf{R}_I + {}^E\mathbf{R}_I\cdot\left[{}^I\mathbf{t}_E\right]_\times \tag{16}$$

$$\frac{\partial{}^E\mathbf{p}_j^e}{\partial\delta\boldsymbol{\theta}} = -\left({}^G\mathbf{R}_E\right)^T \tag{17}$$

where ${}^E\mathbf{R}_I$ is the rotation part of the extrinsic from IMU to event camera, ${}^I\mathbf{t}_E$ is the translation part of the extrinsic from event camera to IMU, and ${}^G\mathbf{R}_E$ is the rotation part of the event camera pose in global frame.

Assuming that for each event map point, we have $-e_j = \mathbf{r}_{E_j} + \mathbf{H}_j^\kappa\cdot(\mathbf{x}_k\boxminus\hat{\mathbf{x}}_k^\kappa)\sim\mathcal{N}(\mathbf{0},\mathbf{\Sigma}_e)$. Then the iterated Kalman filter can be simplified as:

$$\mathbf{K} = \left(\mathbf{H}^T\mathbf{H} + \mathbf{\Sigma}_e\cdot\mathbf{P}^{-1}\right)^{-1}\cdot\mathbf{H}^T \tag{18}$$

$$\hat{\mathbf{x}}_k^{\kappa+1} = \hat{\mathbf{x}}_k^\kappa\boxplus\left(-\mathbf{K}\mathbf{z}_k^\kappa + (\mathbf{I}-\mathbf{K}\mathbf{H})\left(\hat{\mathbf{x}}_k\boxminus\hat{\mathbf{x}}_k^\kappa\right)\right) \tag{19}$$

Once the iteration converge, the state covariance update should be:

$$\bar{\mathbf{P}} = (\mathbf{I}-\mathbf{K}\mathbf{H})\,\mathbf{P} \tag{20}$$

To reduce the influence of noise and make it easier to converge in the optimization process, we utilize a median filtering and a Gaussian blur to the negative TS. We do not use a coarse-to-fine approach or a feature-based method because as mentioned before, the TS frame provides information about the history of past triggered events that can help the convergence of aligning negative TS to the semi-dense event map. The examples of event map points tracking is shown in Fig. 2. To improve efficiency

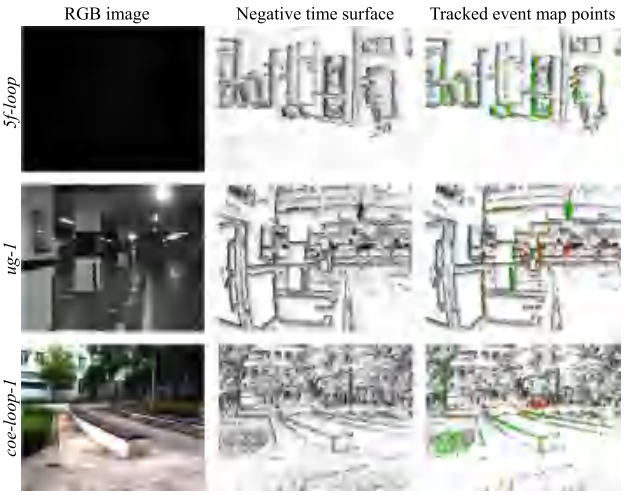

Fig. 2. Event map point tracking in different scenes of our LIE dataset sequences. The three columns are respectively the RGB images from standard camera, negative time surface maps, and tracked event map points projected on median filtered negative time surfaces. Redder map point means higher depth value. In the dark environment in *5f-loop* sequence, the event camera can still have stable output due to its high dynamic range.

without affecting accuracy, when too many points are added in local event map, we randomly select only a set of map points in each iteration instead of computing all local map points. The size of stochastic sampling subset is around 400 points in our experiments. In residual computation process, we randomly choose 400 points from local map, and in each iteration before convergence, the choice of map points is arbitrary.

After residual computation convergence, the state is updated, and new event map point are selected from LiDAR points. The newest LiDAR scan is project onto the TS frame and the LiDAR points will be added to global event map only if the LiDAR points are projected on TS pixels in which recent events are triggered. According to equation (6) and (7), the TS pixel where event is triggered in the last 1 ms has value within [246, 255], which corresponds to negative TS pixel value within [0, 9]. Therefore, only the LiDAR points with projected pixel value lower than 9 will be picked as map points. Considering that the resolution of event cameras are usually low (e.g., the resolution of DAVIS346 camera is 346 × 260), there could be more than one map points that are projected onto the same pixel. We only select the map point with lowest depth as valid map points in each TS pixel.

Besides, after state estimation, a map point removal process is conducted to reduce noise and remove points that could lie on dynamic objects. The semi-dense EIO map represents the 3D edges in scenes, and low pixel values in negative TS correspond to the most recent edge locations. The local map points are projected on the median filtered TS frame with estimated event camera pose, and the map points with high pixel values will be considered as outliers and removed from the global event map.

C. Visual-Inertial Odometry Subsystem

The VIO system is directly adopted from FAST-LIVO [13].

A patch-to-patch coarse-to-fine approach is utilized in visual measurements. The visual map is a sparse map, and each map

TABLE I
THE ABSOLUTE TRANSLATION ERRORS (ATE) AND ABSOLUTE ROTATION ERRORS (ARE) IN THE TUNNEL_E_C SEQUENCE OF ECMD DATASET

	FAST-LIEO 10Hz	FAST-LIEVO 10Hz	FAST-LIEO 60Hz	FAST-LIEVO 60Hz	FAST-LIVO	R3LIVE	FAST-LIO2	ESVIO	Ultimate SLAM
ATE (m)	3.760039	35.84195	0.540759	**0.514881**	29.22779	fail	36.746454	fail	fail
ARE (°)	7.518733	7.997148	5.62893	**4.723980**	8.926026	fail	10.858896	fail	fail

point ${}^{G}\mathbf{p}_n^v$ is associated with image patches from different image frames, and the patches are stored in multi-level pyramid structure. The image plane is divided into grids and the LiDAR scan points projected on image plane with the highest image gradient in each grid are added in visual map with their image patches. In VIO estimation process, the map point patch with the smallest observation angle to current image is selected as reference patch $\mathbf{Q}_n$. Also, an affine transformation $\mathbf{A}_n$ is conducted to project the patch to current image pyramid. The visual measurements residual is defined as:

$$\mathbf{r}_{V_k}\left({}^{G}\mathbf{p}_n^v\right) = \mathbf{I}_k\left(\boldsymbol{\pi}_c\left({}^{I}\mathbf{T}_C^{-1}{}^{G}\mathbf{T}_{I_k}^{-1}{}^{G}\mathbf{p}_n^v\right)\right) - \mathbf{A}_n\mathbf{Q}_n \tag{21}$$

where $\mathbf{I}_k(\cdot)$ is the patch pyramid in image frame at time t_k, $\boldsymbol{\pi}_c(\cdot)$ projects the point in camera frame onto image plane, ${}^{I}\mathbf{T}_C$ is the extrinsic from camera to IMU. The maximum a posteriori (MAP) estimation for this visual measurement is:

$$\min_{\mathbf{x}_k\in\mathcal{M}}\left(\|\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k\|^2_{\hat{\mathbf{P}}_k} + \sum_{n=1}^{m_v}\left\|\mathbf{r}_{V_k}\left({}^{G}\mathbf{p}_n^v\right)\right\|^2_{\boldsymbol{\Sigma}_v}\right) \tag{22}$$

where Σ_v is the visual measurement weight. It is noticed that we put the global map points addition process after VIO state estimation for accuracy consideration.

IV. EXPERIMENTS AND RESULTS

To verify our method, we compare our method with several state-of-the-art (SOTA) multi-sensor fusion SLAM methods on both open source public dataset and our self-captured dataset. Our method and other compared methods are run on a laptop with CPU i7-13700H and 16 GB RAM in Windows Subsystem for Linux (WSL) with Ubuntu 20.04.

A. Benchmark on Public Dataset

We select ECMD [28] as benchmark dataset. The ECMD dataset was collected by a car. We use the left RGB camera (resolution 1920 × 1200), left DVXplore camera (resolution 640 × 480) and the middle Velodyne HDL-32E LiDAR for testing. The frequency of TS is set to two different values (10 Hz and 60 Hz) in experiments. We compare our methods with FAST-LIVO [13], R3LIVE [12] (modified LiDAR front-end to support Velodyne HDL-32E), FAST-LIO2 [15], ESVIO [9] and Ultimate SLAM [7]. For FAST-LIEVO and FAST-LIVO, the images from the RGB camera are resized to 960 × 600. Absolute translation errors (ATE) and absolute rotation errors (ARE) are calculated by using EVO package[1] with the given groundtruth poses from GNSS-RTK/INS suite. The evaluation results of *tunnel_e_c* sequence is shown in Table I, and our FAST-LIEVO and FAST-LIEO outperform among different methods.

[1][Online]. Available: https://github.com/MichaelGrupp/evo

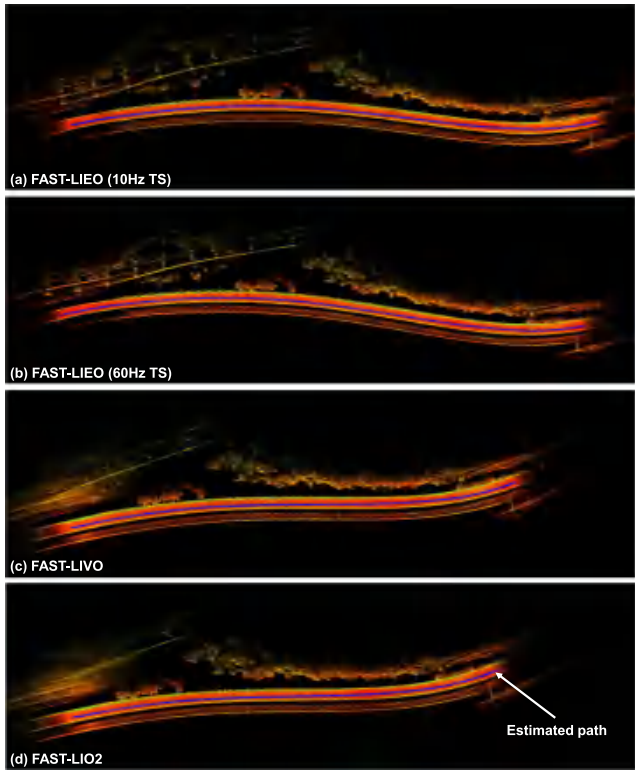

Fig. 3. Mapping results of different methods on ECMD *tunnel_e_c* sequence. The blue lines are the estimated paths. This sequence has an initial velocity larger than 20 m/s.

The *tunnel_e_c* sequence has an initial velocity larger than 20 m/s instead of starting the movement from rest with zero velocity, which is a challenge for initialization and state estimation. As shown in Fig. 3, the mapping results of FAST-LIO2 and FAST-LIVO have obvious drifts at the beginning part while our LIEO can rapidly converge to proper state with event measurements. R3LIVE, ESVIO and Ultimate SLAM fail in this sequence because of bad initialization. Besides, from the comparison results, in bad initialization condition, higher TS rate (60 Hz than 10 Hz) can lead to higher accuracy, but the real-time efficiency could be affected for running LIEO with 60 Hz 640 × 480 TS because of the large time consumption of computing TS. More detailed information about time consumption is provided in Section IV-B6.

It is noticed that we only use one sequence from ECMD because of the time-desynchronization issues. The filter-based methods are sensitive to data timestamps. There are obvious time-offsets between different sensors, especially event cameras, and the time-offsets varies in different sequences. We roughly give a time-offset value (1.5 s) between the left DVXplore camera and the IMU for the *tunnel_e_c* sequence when testing.

TABLE II
END-TO-END TRANSLATION ERRORS (m) IN SEQUENCES OF OUR LIEO DATASET

Sequences	FAST-LIEO (ours)	FAST-LIEVO (ours)	FAST-LIVO	R3LIVE	FAST-LIO2	Ultimate SLAM EIO	Ultimate SLAM EVIO
5f-loop	0.0562	**0.0305**	0.9450	162.8568	0.0613	20.8744	12.0343
ug-loop-1	**0.0972**	0.1066	0.1001	0.1596	0.1236	12.3030	30.6494
ug-loop-2	0.3814	0.7135	0.8001	**0.1209**	0.6663	fail	10.9716
coe-loop-1	0.0377	**0.0052**	0.0285	5.0592	0.0375	71.7176	4.7939
coe-loop-2	**1.2513**	2.2603	2.5081	2.0972	3.2076	126.7693	72.5493
coe-loop-3	**0.0077**	0.0743	0.0785	2.0450	0.0285	fail	36.3994

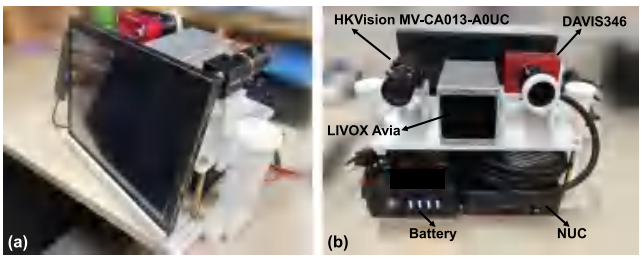

Fig. 4. The platform we used for data collection.

Fig. 5. For *loop-* sequences, the start pose and the end pose are are the same.

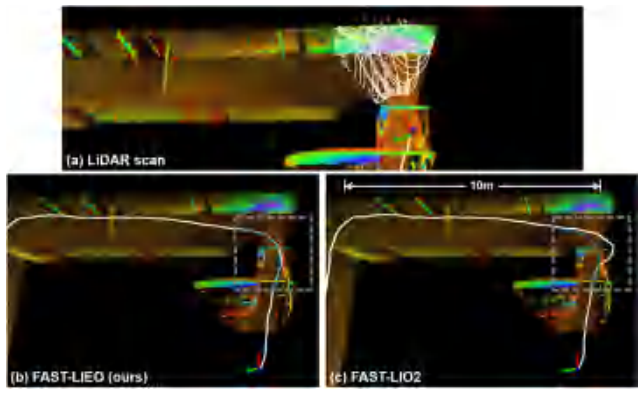

Fig. 6. Test in LiDAR degeneration condition. The blue dotted line is the reference path showing the correction effect of our LIEO.

B. Self-Captured Dataset Experiments

1) LIE Dataset: The handheld platform we used for data collection shown in Fig. 4 includes one event camera (DAVIS346, resolution 346 × 260), one standard RGB camera (HKVision MV-CA013-A0UC, resolution 1280 × 1024) and one LiDAR (Livox Avia). We provide 6 sequences in LIE dataset. Three sequneces were captured in indoor environments and other three were captured in outdoor environments. *5f-* sequences were captured at the 5th floor of the COE (college of engineer) building. *ug-* sequences were captured at an underground parking. *coe-* sequences were collected outside the COE building. *loop-* means the start and the end of the sequence are at the same position and pose.

As shown in Fig. 5, we use a mental rack fixed at ground for our platform to ensure that the start pose and the end pose are the same so that the end-to-end errors can be calculated.

2) Evaluation Results: All compared methods, including FAST-LIEO, R3LIVE, FAST-LIVO, FAST-LIO2 and Ultimate SLAM (including EIO and EVIO), do not have loop closure so that the end-to-end errors can be used for evaluation. The RGB images are resized to 640 × 512 when running FAST-LIEVO and FAST-LIVO while R3LIVE uses the original image resolution. For the different event camera resolution compared with DVxplore used in ECMD, the relative parameter settings, including TS generation threshold and event measurements covariance, are different. To ensure high real-time ability, the TS rate is set to 10 Hz among all sequences, which is the same as the RGB camera frame rate. The evaluation results are shown in Table II. Our methods outperform in most sequences. In *ug-loop-2* sequence, R3LIVE has better performance than any other method. In R3LIVE, the VIO subsystem renders and updates the texture of global point cloud map utilizing RGB images. Benefiting from the rendered map texture and higher resolution RGB image, its visual update calculated by minimizing the photometric error between rendered map points and RGB image has lower drift in stable illumination scenes.

In some sequences, the LIEVO has better performance than LIEO. Our EIO update relies on the edge information provided by event representations. In some texture-less conditions, the edges in time surfaces are relatively less or poorly distributed, which could affect the EIO estimation. The patch-to-patch visual measurement approach does not rely on edge features and can overcome the negative influence of texture-less scenes, resulting in lower errors of LIEVO in some sequences.

3) Experiment in Degeneration Scene: Fig. 6 shows the localization and mapping results in case of LiDAR degeneration. The white lines are the estimated paths. FAST-LIO2 has an obvious drift due to the LiDAR degeneration when facing a wall in corridor (Fig. 6(a)). Our FAST-LIEO successfully pass through the door with event measurements (Fig. 6(b)).

4) Experiments in Weak Light and HDR Environments: The *5f-loop* sequence provides weak illumination and HDR scenes for experiments and the evaluation results is shown in the above Table II. The average speed of this sequence is around 1.4 m/s. R3LIVE has a large drift in this sequence caused by the dark area shown in Fig. 8(d). The drift of FAST-LIVO is due to the weak illumination environment before the end of the sequence

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

Fig. 7. Experiments in large-scale environments. (a) shows the trajectories of the *coe-loop* sequences in satellite map. (b)–(d) are the colored point cloud maps of FAST-LIEVO.

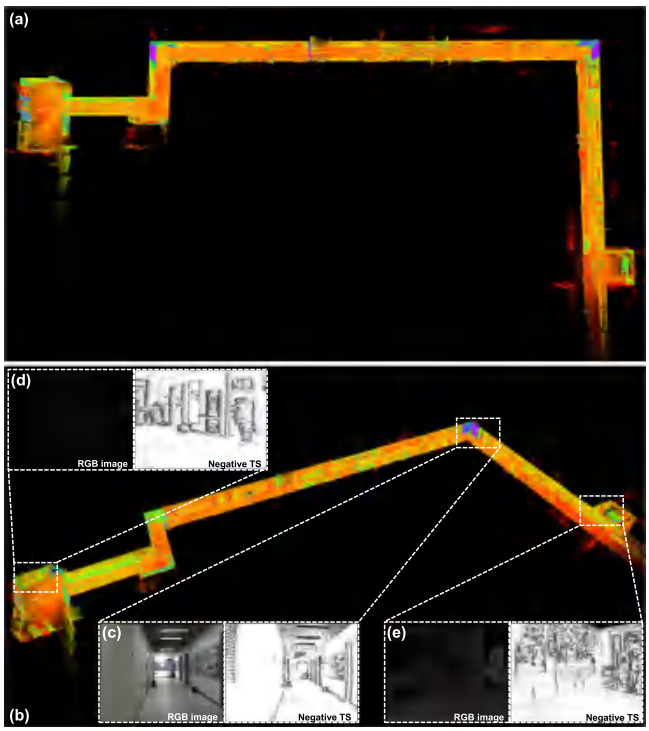

Fig. 8. Test in HDR environments. (a) and (b) show the mapping result of *5f-loop* sequence. (c)–(e) are scenes with different illumination conditions.

shown in Fig. 8(e). Compared to other methods, our LIEO and LIEVO have lower translation drifts within 6 cm.

We also compare the results of LIEO with and without the map point stochastic sampling process described in Section III-B2. The drift of LIEO without the stochastic sampling is 5.40 cm in translation and 2.066° in rotation, which is close to the result with the sampling process (5.62 cm, 2.050°). The overall accuracy is not affected by the map point sampling process.

TABLE III
AVERAGE TIME CONSUMPTION PER FRAME (MS)

EIO subsystem		LIO subsystem	VIO subsystem
Building local map	2.88 (2.55)		
State estimation	1.61 (3.43)		
Global map update	1.43 (1.46)		
Map points removal	0.13 (0.18)		
Total time	6.05 (7.61)	22.03	10.6

5) Experiments in Outdoor Large-Scale Environments: The *coe-loop* sequences, with total length over 1.7 km, were recorded outside the COE building. The evaluation results with end-to-end errors are also shown in Table II. Our methods achieve lower end-to-end drifts in experiments. R3LIVE has larger drifts due to the standard camera exposure issue across bright and shadow areas. The mapping results of FAST-LIEVO are shown in Fig. 7, and the point clouds are colored by RGB images. The results indicate that our method can maintain robustness and accuracy in both indoor and outdoor environments.

6) Real-Time Ability Analysis: Table III shows the average time consumptions of the total system per frame. The results show the high real-time ability of our system. Besides, creating TS with resolution 346 × 260 costs around 5 ms per frame.

In Table III, the numbers in brackets shows the time consumption without the stochastic sampling process described in Section III-B2. The running time of state estimation is reduced by half with the local map point stochastic sampling and the running efficiency is improved without compromising the accuracy.

Considering that the DAVIS346 (346 × 260) we used in our LIE dataset and the DVXplore (640 × 480) in ECMD [28] have different resolutions, we also compare the EIO subsystem running time per TS frame under different resolutions in Table IV. Higher TS resolution requires more time consumption, and the delay of higher resolution EIO is mainly caused by the large

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

TABLE IV
EIO SUBSYSTEM TIME CONSUMPTION PER TS FRAME (MS)

Processes	346×260	640×480
Building TS frame	5.12	15.75
EIO estimation	6.05	12.61

time consumption of creating TS frame. According to the time consumption per frame shown in Table IV, when using higher resolution DVXplore in ECMD dataset, the 60 Hz TS version can not maintain high real-time performance compared to the 10 Hz version.

7) Limitations: The LIEO system could fail when LiDAR degeneration and texture-less conditions occur simultaneously because the two subsystems both fail in this case.

The EIO estimation method in our LIEO is directly aligning negative TS to semi-dense map, which is more dense and direct than feature-based methods like Ultimate SLAM. The alignment relies on initial estimation by IMU propagation and past measurements. In some edge cases the alignment process could have bad initial values so that the EIO estimation could not converge to a proper state.

V. CONCLUSION

In this paper we present FAST-LIEO, a fast and real-time multi-sensor fusion odometry that can tightly fuse LiDAR, IMU, event camera and RGB camera measurements. Our FAST-LIEO supports both LIEO and LIEVO. For LiDAR-inertial-event fusion purpose, we design a novel EIO subsystem in which EIO measurements are calculated with semi-dense event map that are built from LiDAR points. The EIO estimation is done by the alignment of negative time surface maps to the semi-dense event map. A map points removal process is conducted to reduce the noise of event map after EIO state estimation. The experiments were conducted on both public dataset and our self-captured LIE dataset to verify our method in challenging conditions.

ACKNOWLEDGMENT

The authors would like to thank Mengzheng Zhang for helping the LIE dataset experiments.

REFERENCES

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 × 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008.

[2] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240 × 180 130 dB 3 μs latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, Oct. 2014.

[3] G. Gallego et al., "Event-based vision: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 1, pp. 154–180, Jan. 2022.

[4] H. Rebecq, T. Horstschaefer, G. Gallego, and D. Scaramuzza, "EVO: A geometric approach to event-based 6-DOF parallel tracking and mapping in real time," *IEEE Robot. Automat. Lett.*, vol. 2, no. 2, pp. 593–600, Apr. 2017.

[5] Y. Zuo, J. Yang, J. Chen, X. Wang, Y. Wang, and L. Kneip, "DEVO: Depth-event camera visual odometry in challenging conditions," in *Proc. 2022 Int. Conf. Robot. Automat.*, 2022, pp. 2179–2185.

[6] Y. Zhou, G. Gallego, and S. Shen, "Event-based stereo visual odometry," *IEEE Trans. Robot.*, vol. 37, no. 5, pp. 1433–1450, Oct. 2021.

[7] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate SLAM? combining events, images, and IMU for robust visual SLAM in HDR and high-speed scenarios," *IEEE Robot. Automat. Lett.*, vol. 3, no. 2, pp. 994–1001, Apr. 2018.

[8] W. Guan and P. Lu, "Monocular event visual inertial odometry based on event-corner using sliding windows graph-based optimization," in *Proc. 2022 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Kyoto, Japan, 2022, pp. 2438–2445. [Online]. Available: https://ieeexplore.ieee.org/document/9981970/

[9] P. Chen, W. Guan, and P. Lu, "Esvio: Event-based stereo visual inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 8, no. 6, pp. 3661–3668, Jun. 2023.

[10] W. Guan, P. Chen, Y. Xie, and P. Lu, "Pl-EVIO: Robust monocular event-based visual inertial odometry with point and line features," *IEEE Trans. Automat. Sci. Eng.*, vol. 21, no. 4, pp. 6277–6293, Oct. 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10287884/

[11] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry," in *Proc. 2017 IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5816–5824.

[12] 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. 2022 Int. Conf. Robot. Automat.*, 2022, pp. 10672–10678.

[13] 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. 2022 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 4003–4009.

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

[15] 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.

[16] 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. 2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.

[17] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst.*, 2014, vol. 2, no. 9, pp. 1–9.

[18] J. Zhang and S. Singh, "Laser–visual–inertial odometry and mapping with high robustness and low drift," *J. Field Robot.*, vol. 35, no. 8, pp. 1242–1264, 2018.

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

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

[21] 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, Apr. 2018.

[22] 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.

[23] 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.

[24] C. Hertzberg, R. Wagner, U. Frese, and L. Schröder, "Integrating generic sensor fusion algorithms with sound state representations through encapsulation of Manifolds," *Inf. Fusion*, vol. 14, no. 1, pp. 57–77, 2013.

[25] B. Bell and F. Cathey, "The iterated Kalman filter update as a Gauss-Newton method," *IEEE Trans. Autom. Control*, vol. 38, no. 2, pp. 294–297, Feb. 1993.

[26] Y. Cai, W. Xu, and F. Zhang, "IKD-tree: An incremental K-D tree for robotic applications," 2021, *arXiv:2102.10808*.

[27] X. Lagorce, G. Orchard, F. Galluppi, B. E. Shi, and R. B. Benosman, "HOTS: A hierarchy of event-based time-surfaces for pattern recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1346–1359, Jul. 2017.

[28] P. Chen et al., "ECMD: An event-centric multisensory driving dataset for SLAM," *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 407–416, Jan. 2024.