SR-LIVO: LiDAR-Inertial-Visual Odometry and Mapping with Sweep Reconstruction

Zikang Yuan[1], Jie Deng[2], Ruiye Ming[2], Fengtian Lang[2] and Xin Yang[2]*

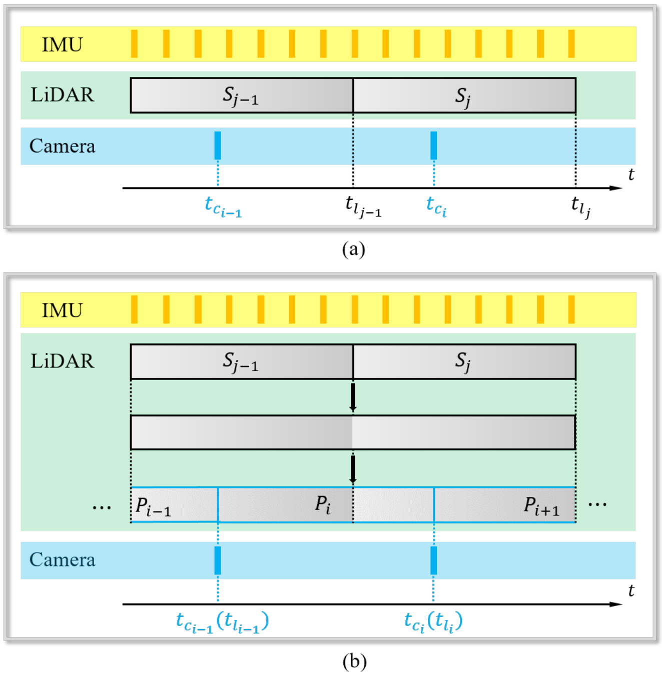
***Abstract*—Existing LiDAR-inertial-visual odometry and mapping (LIV-OAM) systems mainly utilize the LiDAR-inertial odometry (LIO) module for structure reconstruction and the LiDAR-assisted visual-inertial odometry (VIO) module for color rendering. However, the performance of existing LiDAR-assisted VIO module doesn't match the accuracy delivered by LIO systems in the scenarios containing rich textures and geometric structures (i.e., without failure mode for both camera and LiDAR). This paper introduces SR-LIVO, an advanced and novel LIV-OAM system employing sweep reconstruction to align reconstructed sweeps with image timestamps. This allows the LIO module to accurately determine states at all imaging moments, enhancing pose accuracy and processing efficiency. Experimental results on two public datasets demonstrate that: 1) our SR-LIVO outperforms the existing state-of-the-art LIV-OAM systems in both pose accuracy, rendering performance and runtime efficiency; 2) In scenarios with rich textures and geometric structures, the LIO framework can provide more accurate pose than existing LiDAR-assisted VIO framework, and thus helps rendering. We have released our source code to contribute to the community development in this field.**

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

I. Introduction

In robotic applications like autonomous vehicles [4] and drones [2], [3], cameras, 3D light detection and ranging (LiDAR) and inertial measurement unit (IMU) are key sensors. The integration of IMU measurements can provide motion prior to ensure accurate and quick state estimation. While LiDAR excels in capturing 3D structures, it lacks color information which cameras compensate for. This synergy has led to the rise of LiDAR-inertial-visual odometry and mapping (LIV-OAM) as a leading method for accurate state estimation and dense colored map reconstruction.

Existing state-of-the-art LIV-OAM systems [5], [6], [19] mainly consist of a LIO module and a LiDAR-assisted VIO module. The LIO module reconstructs 3D structures while the LiDAR-assisted VIO module colors them. Both LIO and LiDAR-assisted VIO modules perform state estimation. The LIO module obtains the estimated state at the end timestamp of a LiDAR sweep (e.g., $t_{l_{j-1}}$ and t_{l_j} in Fig. 1 (a)), and the LiDAR-assisted VIO module solves the state at the timestamp of each captured image (e.g., $t_{c_{i-1}}$ and t_{c_i} in Fig. 1 (a)). The experimental results in our previous work [17] illustrate that existing LO can achieve more accurate pose estimation than LiDAR-assisted VO in scenarios where failure modes of camera and LiDAR are not a factor. Given that the geometric observations from LiDAR and the visual observations provided by camera are much stronger than those from IMU, the inclusion of an IMU does not alter the comparative performance trend between LIO and LiDAR-assisted VIO systems.



(a)

(b)

Fig. 1. Illustration of (a) the raw sensor measurements from IMU, LiDAR and camera, (b) the raw sensor measurements from IMU, camera and the reconstructed LiDAR sweep data, where the end timestamp of reconstructed sweep is aligned with the timestamp of captured image.

In this paper, we propose SR-LIVO, a novel and advanced LIV-OAM framework that enhances both accuracy and reliability. We shift state estimation entirely to the LIO module, known for its better precision. A key challenge brought by this design is that: the timestamp of the captured images and the end timestamp of LiDAR sweeps are often not aligned.

Manuscript received: December, 27, 2023; Revised: March, 17, 2024; Accepted: April, 4, 2024.

This paper was recommended for publication by Editor Pascal Vasseur upon evaluation of the Associate Editor and Reviewers' comments.

This work is supported by the National Natural Science Foundation of China (62122029, 62061160490, U20B200007).

[1] Zikang Yuan is with Institute of Artificial Intelligence, Huazhong University of Science and Technology, Wuhan, 430074, China. (E-mail: yzk2020@hust.edu.cn)

[2] Jie Deng, Ruiye Ming, Fengtian Lang and Xin Yang* are with the Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan, 430074, China. (* represents the corresponding author. E-mail: u202013949@hust.edu.cn; M202272555@hust.edu.cn; M202372913@hust.edu.cn; xinyang2014@hust.edu.cn)

Digital Object Identifier (DOI): see top of this page.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2024.3389415

Without the state of the image acquisition moment, we can hardly use image data to render color for the restored 3D structures. To address this issue, we highlight an additional important function of sweep reconstruction [17], i.e., data synchronization (e.g., $t_{l_{i-1}}$ and t_{l_i} in Fig. 1 (b)) with the timestamp of captured images (e.g., $t_{c_{i-1}}$ and t_{c_i} in Fig. 1 (b)). In this way, the LIO module can directly estimate states at image capture moments. Consequently, the vision module's role is simplified to optimizing camera-parameters (e.g., camera intrinsics, extrinsics and time-offset) and completing the color rendering task.

Experimental results on the public datasets NTU_VIRAL [8], $R3Live$ [5] demonstrate the following key findings: 1) Our system outperforms existing state-of-the-art LIV-OAM systems in terms of the smaller absolute trajectory error (ATE) than [5], [6] and higher Peak Signal-to-Noise Ratio (PSNR) and Structure Similarity Index Measure (SSIM) than [5], and is much more efficient than [5]; 2) In scenario with rich textures and geometric structures, compared with existing LiDAR-assisted VIO framework, the LIO framework can provide more accurate pose on the testing dataset, which also helps rendering. Meanwhile, the visualization results demonstrate that our SR-LIVO can achieve comparable reconstruction results to [5] on their self-collected dataset (i.e., $R3Live$), and can achieve much superior results to [5] on the NTU_VIRAL dataset. We have released the source code of our system to benefit the development of the community[1].

To summarize, the main contributions of this work are two folds: 1) We identify an important function of sweep reconstruction, i.e., aligning LiDAR sweep and image timestamps; 2) We design a new LIV-OAM system based on sweep reconstruction, where the vision module is no longer needed to perform state estimation but only for coloring the reconstructed map.

The rest of this paper is structured as follows. Sec. II reviews the relevant literature. Sec. III provides preliminaries. Secs. IV and V presents system overview and details, followed by experimental evaluation in Sec. VI. Sec. VI concludes the paper.

II. Related Work

In recent years, various LiDAR-visual and LiDAR-inertial-visual fusion frameworks have been proposed. V-LOAM [18] is the first cascaded LIV-OAM framework that provides motion priori for LiDAR odometry via a loosely coupled VIO. [12] cascades a tightly-coupled stereo VIO, a LiDAR odometry and a LiDAR-based loop closing module together. Compared with [18] and [12], the vision module of DV-LOAM [13] utilizes the direct method to perform pose estimation and multi-frame joint optimization in turn to make the vision module provide more accurate motion priori for the subsequent LiDAR module. Lic-Fusion [20] combines the IMU measurements, sparse visual features, LiDAR features with online spatial and temporal calibration within the multi state constrained Kalman filter (MSCKF) framework. To further enhance the accuracy and the robustness of the LiDAR points registration, LIC-Fusion 2.0 [21] proposes a plane-feature tracking algorithm across multiple LiDAR sweeps in a sliding-window and refines the pose of sweep within the window. LVI-SAM [11] integrates the data from camera, LiDAR and IMU into a tightly-coupled graph-based optimization framework. The vision and LiDAR module of LVI-SAM can run independently when each other fails, or jointly when both visual and LiDAR features are sufficient. R2Live [6] firstly proposes to run a LIO module and a LiDAR-assisted VIO module in parallel, where the LIO module provides geometric structure information for the LiDAR-assisted VIO module. The back end utilizes the visual landmarks to perform graph-based optimization. Based on R2Live, R3Live [5] omits the graph-based optimization module and adds a color rendering module for dense color map reconstruction. Compared to R3Live, Fast-LIVO [19] combines LiDAR, camera and IMU measurements into a single error state iterated Kalman filter (ESIKF), which can be updated by both LiDAR and visual observations. mVIL-Fusion [14] proposes a three-staged cascading LIV-SLAM framework, which consists of a LiDAR-assisted VIO module, a multi sweep-to-sweep joint optimization module, a sweep-to-map optimization module and a loop closing module. VILO-SLAM [9] fuses 2D LiDAR residual factor, IMU residual factor and visual reprojection residual factor into an optimization-based framework. In [15], a weight function is designed based on geometric structure and reflectivity to improve the performance of solid-state LIO under severe linear acceleration and angular velocity changes.

III. Preliminary

A. Coordinate Systems

We denote $(\cdot)^w$, $(\cdot)^l$, $(\cdot)^o$ and $(\cdot)^c$ as a 3D point in the world coordinate, the LiDAR coordinate, the IMU coordinate and the camera coordinate respectively. The world coordinate is coinciding with $(\cdot)^o$ at the starting position.

We denote the IMU coordinate for taking the i_{th} IMU measurement at time t_i as o_i and the corresponding camera coordinate at t_i as c_i. The transformation matrix (i.e., extrinsics) from c_i to o_i is denoted as $\mathbf{T}_{c_i}^{o_i} \in SE(3)$, where $\mathbf{T}_{c_i}^{o_i}$ consists of a rotation matrix $\mathbf{R}_{c_i}^{o_i} \in SO(3)$ and a translation vector $\mathbf{t}_{c_i}^{o_i} \in \mathbb{R}^3$. Similarly, we can also obtain the extrinsics form LiDAR to IMU, i.e., $\mathbf{T}_{l_i}^{o_i}$. For the datasets involved in this work, we use the internal IMU of LiDAR, therefore, we treat $\mathbf{T}_{l_i}^{o_i}$ as absolutely accurate and do not require online correction. For $\mathbf{T}_{c_i}^{o_i}$, we optimize it online because the camera is a separate external sensor.

B. Distortion Correction

For each camera image, we utilized the offline-calibrated distortion parameters to correct the image distortion. For LiDAR points, we utilize the IMU-integrated pose to compensate the motion distortion.

IV. System Overview

Fig. 2 illustrates the framework of our system which consists of three main modules: a sweep reconstruction module, a

[1] https://github.com/ZikangYuan/sr_livo

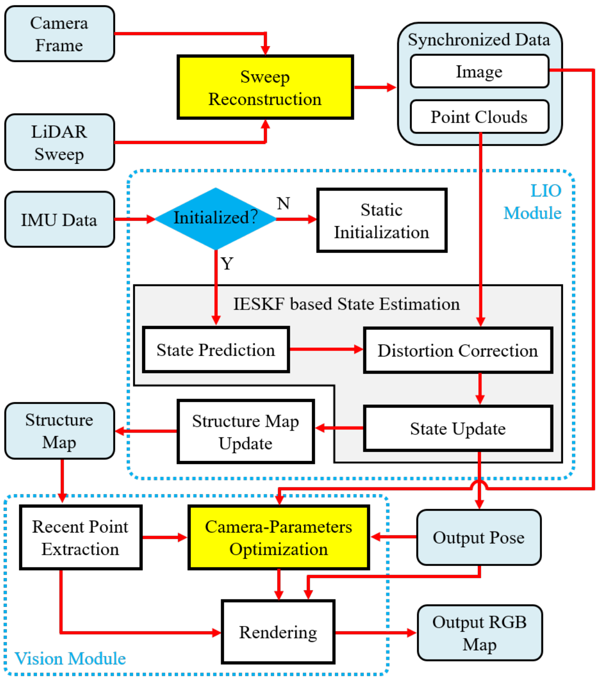


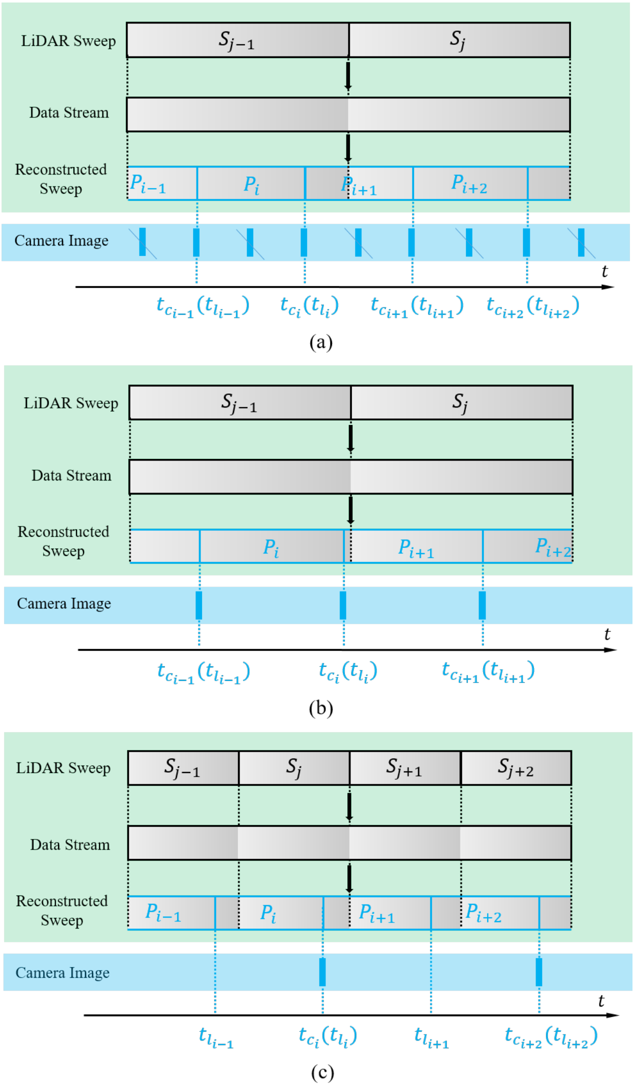
Fig. 2. Overview of our SR-LIVO which consists of three main modules: a sweep reconstruction module for timestamp alignment, a LIO module for state estimation and structure restoring, and a vision module for camera-parameters optimization and color rendering. The yellow rectangles indicate the operations of our system that are different from that of existing state-of-the-art frameworks.

LIO state estimation module and a vision module. The sweep reconstruction module aligns the end timestamp of reconstructed sweep with the timestamp of captured image. The LIO module estimates the state of the hardware platform, and restores the structure in real time. The vision module optimizes the camera-parameters including camera intrinsics, extrinsics, time-offset, and renders color to the restored structure map in real time. For map management, we utilized the Hash voxel map, which is the same as CT-ICP [1]. The implementation details of various parts of the LIO module are exactly the same as our previous work SR-LIO [16], therefore, we omit the introduction of this module, and only introduce the details of sweep reconstruction and our vision module in Sec. V. It is necessary to emphasize that the state estimation of SR-LIVO is completely done by the LIO module, and the vision module focuses on camera-parameter optimization and rendering.

V. System Details

A. Sweep Reconstruction

In our previous work SDV-LOAM [17], we firstly propose the sweep reconstruction idea, which increases the frequency of LiDAR sweeps to the same frequency as camera images. In this work, we point out another important function of this idea: aligning the end timestamp of reconstructed sweep to the timestamp of captured image. According to the frequency of LiDAR sweeps and camera images, the processing method is also different in practice. There are three specific cases as following:



(a)

(b)

(c)

Fig. 3. Illustration of the sweep reconstruction method under three situations: (a) The frequency of captured images is more than twice that of raw LiDAR sweeps. (b) The frequency of captured images is less than twice but more than that of raw LiDAR sweeps. (c) The frequency of captured images is less than that of raw LiDAR sweeps.

The frequency of captured images is more than twice that of raw LiDAR sweeps (as shown in Fig. 3 (a)). In this case, we firstly down-sample the frequency of camera images to twice the frequency of raw LiDAR sweeps. Then we disassemble the raw LiDAR sweep (i.e., S_{j-1} and S_j in Fig. 3 (a)) into continuous point cloud data stream, and recombine the point cloud data stream with aligning the end timestamp (e.g., t_{l_i} in Fig. 3 (a)) of reconstructed sweep (i.e., P_i in Fig. 3 (a)) to the timestamp of down-sampled captured image (e.g., t_{c_i} in Fig. 3 (a)). In this way, the number of LiDAR points in reconstructed sweep is only half that in raw input sweep. For spinning LiDAR, not only the number, but also the horizontal distribution range is reduced from 360° to 180°. If the frequency of reconstructed sweeps is more than twice that of raw sweeps, the number and horizontal distribution range of points will decrease further, and eventually the LIO module may fail to perform properly. Therefore, we need to down-sample the frequency of camera images to at most twice the frequency of raw LiDAR sweeps.

The frequency of captured images is less than twice but more than that of raw LiDAR sweeps (as shown in Fig. 3 (b)). In this case, we directly disassemble the raw LiDAR sweep (i.e., S_{j-1} and S_j in Fig. 3 (b)) into continuous point cloud data stream, and recombine the point cloud data stream with aligning the end timestamp (e.g., t_{l_i} in Fig. 3 (b)) of reconstructed sweep (e.g., P_i in Fig. 3 (b)) to the timestamp of captured image (e.g., t_{c_i} in Fig. 3 (b)). In this way, the point cloud number and the horizontal distribution range of reconstructed sweeps are less than that of raw LiDAR sweeps, but also enough to support the LIO module run properly.

The frequency of captured images is less than that of raw LiDAR sweeps (as shown in Fig. 3 (c)). In this case, we directly disassemble the raw LiDAR sweep (i.e., S_{j-1}, S_j, S_{j+1} and S_{j+2} in Fig. 3 (c)) into continuous point cloud data stream, and recombine the point cloud data stream according to the following two situations: 1) We assume the current reconstructed sweep begins at t_{l_i} but the end timestamp is not yet determined. When the time interval from current moment to t_{l_i} reaches the time period of a raw LiDAR sweep and there are no images around current moment, we set the current moment as the end timestamp (e.g., $t_{l_{i+1}}$ in Fig. 3 (c)) of this reconstructed sweep (e.g., P_{i+1} in Fig. 3 (c)). 2) We assume the current reconstructed sweep begins at $t_{l_{i+1}}$ but the end timestamp is not yet determined. When the current moment reaches the timestamp of captured image (e.g., $t_{c_{i+2}}$ in Fig. 3 (c)) and there is a sufficient time interval from $t_{c_{i+2}}$ to $t_{l_{i+1}}$, we set the current moment as the end timestamp (e.g., $t_{c_{i+2}}(t_{l_{i+2}})$ in Fig. 3 (c)) of reconstructed sweep (e.g., P_{i+2} in Fig. 3 (c)). Under the situation 2), not all synchronized data includes image data (i.e., P_{i-1} and P_{i+1} in Fig. 3 (c)). For synchronized data without camera images, we just utilize the LIO module to estimate state without the vision module running. For synchronized data with both point cloud and image data, the LIO module and the vision module execute in turn.

B. Vision Module

1) Recent Point Extraction: Similar as R3Live [5], firstly we record all recently visited voxels $\{V_1, V_2, \cdots, V_m\}$ when performing structure map update. Then, we select the newest added point from each visited voxel V_i, to obtain the recent point set $P_r = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_m\}$.

2) Camera-Parameters Optimization: Different from existing state-of-the-art frameworks (e.g., R3Live and Fast-LIVO), we have estimated the state at the timestamp (e.g., t_k) of captured images (e.g., c_k) in the LIO module. Therefore, we no longer need to solve state (i.e., pose, velocity and IMU bias) in the vision module, but only need to utilize the image data to optimize some camera-parameters:

$$\boldsymbol{x}_k = \left[t_{c_k}^{o_k}, \mathbf{R}_{c_k}^{o_k}, \mathbf{t}_{c_k}^{o_k}, \phi_k\right]^T \tag{1}$$

where $t_{c_k}^{o_k}$ is the time-offset between IMU and camera while LiDAR is assumed to be synced with the IMU already. $\mathbf{R}_{c_k}^{o_k}$ and $\mathbf{t}_{c_k}^{o_k}$ are the extrinsics between camera and IMU. $\phi_k = \left[f_{x_k}, f_{y_k}, c_{x_k}, c_{y_k}\right]^T$ are the camera intrinsics, where (f_{x_k}, f_{y_k}) denote the camera focal length, (c_{x_k}, c_{y_k}) denote the offsets of the principal point from the top-left corner of the image plane.

To balance the effects of previous estimates and the current image observation on camera-parameters, we utilize an ESIKF to optimize the camera-parameters $\boldsymbol{x}_k$, where the error state $\delta\boldsymbol{x}_k$ is defined as:

$$\delta\boldsymbol{x}_k = \left[\delta t_{c_k}^{o_k}, \delta\mathbf{R}_{c_k}^{o_k}, \delta\mathbf{t}_{c_k}^{o_k}, \delta\phi_k\right]^T \tag{2}$$

For the state prediction, the error state $\delta\boldsymbol{x}_k$ and covariance $\mathbf{P}_k$ is propagated as:

$$\begin{aligned} \delta\boldsymbol{x}_k &= \mathbf{0} \\ \mathbf{P}_k &= \mathbf{P}_{k-1} \end{aligned} \tag{3}$$

For the state update, the minimizing PnP projection error and the minimizing photometric error are used to update $\delta\boldsymbol{x}$ in turn.

The minimizing PnP projection error. Assuming that we have tracked n map points $P_t = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_n\}$, and their projection on image c_{k-1} is $\left\{\boldsymbol{\rho}_{1_{k-1}}, \boldsymbol{\rho}_{2_{k-1}}, \cdots, \boldsymbol{\rho}_{n_{k-1}}\right\}$, we leverage the Lucas-Kanade optical flow to find out their locations in the current image c_k: $\left\{\boldsymbol{\rho}_{1_k}, \boldsymbol{\rho}_{2_k}, \cdots, \boldsymbol{\rho}_{n_k}\right\}$. P_t is the set of 3D landmarks which correspond to 2D pixels of last image frame, where the 2D pixels are tracked from previous image frames or newly selected evenly from the projection of dense map at regular intervals on last image. For the exemplar point $\mathbf{p}_i \in P_t$, we calculate the re-projection error by:

$$\begin{gathered} \mathbf{p}_i^{c_k} = \left(\mathbf{R}_{o_k}^{w} \cdot \mathbf{R}_{c_k}^{o_k}\right)^T \cdot \mathbf{p}_i^{w} - \mathbf{R}_{c_k}^{o_k\,T} \cdot \mathbf{t}_{c_k}^{o_k} - \left(\mathbf{R}_{o_k}^{w} \cdot \mathbf{R}_{c_k}^{o_k}\right)^T \cdot \mathbf{t}_{o_k}^{w} \\ \mathbf{r}^{\mathbf{p}_i} = \boldsymbol{\rho}_{i_k} - \pi\left(\mathbf{p}_i^{c_k}, \boldsymbol{x}_k\right) \end{gathered} \tag{4}$$

where $\pi\left(\mathbf{p}_i^{c_k}, \boldsymbol{x}_k\right)$ is computed as below:

$$\begin{aligned} \pi\left(\mathbf{p}_i^{c_k}, \boldsymbol{x}_k\right) = &\left[f_{x_k} \cdot \frac{[\mathbf{p}_i^{c_k}]_x}{[\mathbf{p}_i^{c_k}]_z} + c_{x_k}, f_{y_k} \cdot \frac{[\mathbf{p}_i^{c_k}]_y}{[\mathbf{p}_i^{c_k}]_z} + c_{y_k}\right]^T \\ &+ \frac{t_{c_k}^{o_k}}{\Delta t_{k-1,k}}\left(\boldsymbol{\rho}_{i_k} - \boldsymbol{\rho}_{i_{k-1}}\right) \end{aligned} \tag{5}$$

where $\Delta t_{k-1,k}$ is the time interval between the last image c_{k-1} and the current image c_k. In Eq. 5, the first item is the pin-hole projection function and the second one is the online-temporal correction factor [10]. We can express the observation matrix $\mathbf{h}$ as:

$$\mathbf{h} = \left[\mathbf{r}^{\mathbf{p}_1 T}, \mathbf{r}^{\mathbf{p}_2 T}, \cdots, \mathbf{r}^{\mathbf{p}_n^T}\right]^T \tag{6}$$

The corresponding Jacobian matrix of observation constraint $\mathbf{H}$ is calculated as:

$$\begin{aligned} \mathbf{H} &= \left[\mathbf{H}_1^T, \mathbf{H}_2^T, \cdots, \mathbf{H}_n^T\right]^T \\ \mathbf{H}_i &= \left[\begin{array}{cccc} \frac{\partial \mathbf{r}^{\mathbf{p}_i}}{\partial t_{c_k}^{o_k}}^T & \frac{\partial \mathbf{r}^{\mathbf{p}_i}}{\partial \mathbf{R}_{c_k}^{o_k}}^T & \frac{\partial \mathbf{r}^{\mathbf{p}_i}}{\partial \mathbf{t}_{c_k}^{o_k}}^T & \frac{\partial \mathbf{r}^{\mathbf{p}_i}}{\partial \phi_k}^T \end{array}\right]^T \end{aligned} \tag{7}$$

The minimizing photometric error. For the exemplar point $\mathbf{p}_i \in P_r$, if $\mathbf{p}_i$ has been rendered the color intensity $\boldsymbol{\gamma}_i$, we firstly project $\mathbf{p}_i^w$ to $(\cdot)^{c_k}$ by Eq. 4 and 5, and then calculate the photometric error by:

$$\mathbf{r}^{\mathbf{p}_i} = \boldsymbol{\gamma}_i - \boldsymbol{I}\left(\pi\left(\mathbf{p}_i^{c_k}, \boldsymbol{x}_k\right)\right) \tag{8}$$

where $\boldsymbol{I}(\cdot)$ represents the color intensity of image pixel. The observation matrix and the corresponding Jacobin matrix have the similar formula as Eq. 6-7.

For each image c_k, we firstly utilize **the minimizing PnP projection error** to update the ESIKF, and then utilize **the minimizing photometric error** to update the ESIKF. We only optimize camera-parameters of c_k, while the state of c_k has been solved in our LIO module.

3) Rendering: After the camera-parameters have been optimized, we perform the rendering function to update the color of map points. To ensure the density of colored point cloud map, we not only render points in the recent point set P_r, but also render all points in recently visited voxels $\{V_1, V_2, \cdots, V_m\}$. The rendering function we utilize is the same as R3Live [5].

VI. Experiments

We evaluate our system on the drone-collected dataset NTU_VIRAL [8] and the handheld device-collected dataset $R3Live$ [5]. The sensors used in NTU_VIRAL dataset are the left camera, the horizontal 16-channel OS1 gen1[4] LiDAR and its internal IMU, while the high-accuracy laser tracking methods are employed to provide position ground truth. The sensors used in $R3Live$ dataset are the camera, the LiVOX AVAI LiDAR and its internal IMU, while no position ground truth data are provided. We utilize all sequences of NTU_VIRAL [8] and 6 sequences (i.e., $r3live$_01: hku_campus_seq_00, $r3live$_02: hku_campus_seq_01, $r3live$_03: hku_campus_seq_02, $r3live$_04: hku_park_00, $r3live$_05: hku_park_01 and $r3live$_06: $hkust_campus_seq$_02) of $R3Live$ for evaluation. A consumer-level computer equipped with an Intel Core i7-11700 and 32 GB RAM is used for all experiments.

For our testing on the $R3Live$ dataset, we selectively utilize only 6 sequences due to specific considerations. Some of the excluded sequences feature degenerate scenarios that compromise the performance of LIO, making them suitable for our evaluation. The development of SR-LIVO is grounded in our belief that the existing LIO framework can provide more accurate pose than current LiDAR-assisted VIO frameworks in scenarios without degeneration affecting both camera and LiDAR. Therefore, sequences that exhibit failure modes for LiDAR were intentionally omitted from our testing regimen. Additionally, we exclude certain sequences on account of their length, which resulted in the creation of a global colored map that surpassed the 32GB RAM capacity of our testing device, rendering them impractical for our testing purposes.

A. Comparison of the State-of-the-Arts

We compare our system with two state-of-the-art LIVOAM systems, i.e., R3Live [5] and Fast-LIVO [19], on NTU_VIRAL dataset [8] and $R3Live$ dataset [5]. For the NTU_VIRAL dataset, we utilize the universal evaluation metrics - absolute translational error (ATE) as the evaluation metrics. The $R3Live$ dataset does not provide the position groundtruth, so we only evaluate ATE on NTU_VIRAL dataset. For a fair comparison, we obtain the results of above systems based on the source code provided by the authors.

TABLE I
RMSE of ATE on NTU_VIRAL dataset (unit: m)

	R3Live [5]	Fast-LIVO [19]	Ours
eee_01	1.69	0.28	**0.21**
eee_02	×	**0.18**	0.23
eee_03	0.64	0.26	**0.22**
nya_01	0.63	0.34	**0.18**
nya_02	0.35	0.29	**0.19**
nya_03	0.23	0.29	**0.20**
sbs_01	0.40	0.73	**0.12**
sbs_02	0.27	0.25	**0.22**
sbs_03	**0.21**	0.24	**0.21**

Denotations: ”×” means the system drifts halfway through the corresponding sequence.

TABLE II
Rendering Comparison with R3Live

	PSNR ↑		SSIM ↑	
	R3Live	Ours	R3Live	Ours
$r3live$_01	16.76	**18.02**	0.73	**0.79**
$r3live$_02	17.31	**17.73**	0.76	**0.78**
$r3live$_03	14.75	**16.15**	0.69	**0.76**
$r3live$_04	15.55	**15.91**	0.72	**0.73**
$r3live$_05	15.48	**16.04**	0.71	**0.72**
$r3live$_06	14.28	**15.22**	0.68	**0.73**
eee_01	12.11	**18.92**	0.37	**0.82**
eee_02	×	**20.83**	×	**0.85**
eee_03	17.34	**22.84**	0.51	**0.82**
nya_01	14.07	**19.80**	0.49	**0.84**
nya_02	13.50	**17.75**	0.68	**0.86**
nya_03	15.53	**18.42**	0.79	**0.88**
sbs_01	15.87	**20.28**	0.60	**0.83**
sbs_02	17.42	**22.45**	0.52	**0.79**
sbs_03	15.56	**21.83**	0.51	**0.79**

Denotations: ”↑” means the higher value is better, and ”×” means the system drifts halfway through the corresponding sequence.

Results in Table I demonstrate that our system outperforms R3Live and Fast-LIVO for almost all sequences in terms of smaller ATE. ”×” means the system drifts halfway through the run, where our SR-LIVO has better robustness than R3Live on NTU_VIRAL dataset.

In addition, we also compare the rendering effects of R3Live and our SR-LIVO. Following Nerf [7], we conduct our evaluation by first projecting the colorized map onto each image. We then calculate two metrics, i.e., Peak Signal-to-Noise Ratio (PSNR) and Structure Similarity Index Measure (SSIM), using the colors of map points and corresponding projected pixels to assess rendering effectiveness. A higher PSNR indicates better quality, while a SSIM value close to 1 (with 1 being the maximum) signifies a more accurate rendering. Results in Table II show that our SR-LIVO can achieve higher PSNR and SSIM than R3Live on all testing sequences, demonstrating the better rendering performance of our system.

TABLE III
RMSE OF ATE COMPARISON ON LIDAR-ASSISTED VIO MODULE AND LIO MODULE (UNIT: M)

	R3Live	R3Live (V)	Fast-LIVO	Fast-LIVO(V)	Ours	Ours(V)
eee_01	**1.69**	1.71	**0.28**	0.30	**0.21**	0.24
eee_02	×	×	**0.18**	0.26	**0.23**	**0.23**
eee_03	**0.64**	0.81	**0.26**	0.27	**0.22**	0.27
nya_01	**0.63**	**0.63**	**0.33**	**0.33**	**0.18**	0.19
nya_02	**0.35**	0.41	**0.29**	0.30	**0.19**	0.20
nya_03	**0.23**	0.32	**0.22**	0.29	**0.20**	0.24
sbs_01	**0.40**	0.70	0.73	**0.42**	**0.12**	0.38
sbs_02	**0.27**	0.33	**0.25**	0.31	**0.22**	**0.22**
sbs_03	**0.21**	0.23	**0.24**	0.27	**0.21**	0.22

Denotations: "×" means the system drifts halfway through the corresponding sequence.

TABLE IV
RENDERING EFFECT WITH POSE FROM LIDAR-ASSISTED VIO AND LIO MODULE

	PSNR ↑		SSIM ↑	
	Ours(V)	Ours	Ours(V)	Ours
$r3live$_01	16.05	**18.02**	0.71	**0.79**
$r3live$_02	15.95	**17.73**	0.71	**0.78**
$r3live$_03	14.36	**16.15**	0.68	**0.76**
$r3live$_04	14.45	**15.91**	0.63	**0.73**
$r3live$_05	15.18	**16.04**	0.70	**0.72**
$r3live$_06	13.69	**15.22**	0.64	**0.73**
eee_01	15.07	**18.92**	0.64	**0.82**
eee_02	16.99	**20.83**	0.71	**0.85**
eee_03	18.13	**22.84**	0.63	**0.82**
nya_01	16.27	**19.80**	0.70	**0.84**
nya_02	14.25	**17.75**	0.74	**0.86**
nya_03	14.63	**18.42**	0.75	**0.88**
sbs_01	16.72	**20.28**	0.69	**0.83**
sbs_02	18.93	**22.45**	0.63	**0.79**
sbs_03	17.90	**21.83**	0.64	**0.79**

Denotations: "↑" means the higher value is better.

B. Comparison of LIO module and LiDAR-assisted VIO module

Our system is designed based on the logic that: the state estimation performance of existing LIO is more accurate than that of LiDAR assisted VIO on scenario without failure modes of camera and LiDAR. Therefore, we provide quantitative data in this section to prove that our logical base is correct. We compare the ATE results of the LiDAR-assisted VIO module with the LiDAR module on R3Live, Fast-LIVO and our SR-LIVO. Our proposed SR-LIVO does not include the LiDAR-assisted VIO module, and we implement a LiDAR-assisted VIO module modeled similar as R3Live to complete this ablation study.

The results in Table III demonstrate that the accuracy of LIO modules is superior to that of LiDAR-assisted VIO modules, whether the framework is based on R3Live, Fast-LIVO or our SR-LIVO on NTU_VIRAL dataset. In addition, we also compare the rendering performance of our SR-LIVO and ours(V), where SR-LIVO(i.e., "ours" in Table III) utilizes the pose estimates from the LIO module, and ours(V) utilizes the pose estimates from the implemented LiDAR-assisted VIO module for rendering. Results in Table IV show that our SR-LIVO can achieve higher PSNR and SSIM than ours(V) on all testing sequences. This result demonstrates that the proposed sweep reconstruction with time alignment can greatly improve the rendering performance by allowing the system to provide a more accurate pose at the exact moments of image capture.

TABLE V
RENDERING EFFECT WITH/WITHOUT CAMERA-PARAMETER OPTIMIZATION

	PSNR ↑		SSIM ↑	
	don't optimize	optimize	don't optimize	optimize
$r3live$_01	17.19	**18.02**	0.75	**0.79**
$r3live$_02	17.39	**17.73**	0.76	**0.78**
$r3live$_03	15.35	**16.15**	0.71	**0.76**
$r3live$_04	15.50	**15.91**	0.70	**0.73**
$r3live$_05	15.83	**16.04**	0.71	**0.72**
$r3live$_06	14.94	**15.22**	0.70	**0.73**
eee_01	18.73	**18.92**	0.81	**0.82**
eee_02	20.60	**20.83**	0.84	**0.85**
eee_03	22.74	**22.84**	0.81	**0.82**
nya_01	19.14	**19.80**	0.80	**0.84**
nya_02	17.54	**17.75**	0.84	**0.86**
nya_03	18.09	**18.42**	0.87	**0.88**
sbs_01	19.86	**20.28**	0.81	**0.83**
sbs_02	22.09	**22.45**	0.77	**0.79**
sbs_03	21.61	**21.83**	0.77	**0.79**

Denotations: "↑" means the higher value is better.

TABLE VI
TIME CONSUMPTION PER SWEEP (UNIT: MS)

	Vision		LiDAR		Total	
	R3Live	Ours	R3Live	Ours	R3Live	Ours
$r3live$_01	38.90	**20.86**	17.51	**9.67**	50.57	**30.53**
$r3live$_02	33.80	**19.95**	24.23	**10.83**	49.95	**30.78**
$r3live$_03	37.16	**21.51**	18.16	**9.55**	49.27	**31.06**
$r3live$_04	43.25	**20.33**	19.33	**10.54**	56.14	**30.87**
$r3live$_05	40.23	**19.64**	20.95	**11.78**	54.20	**31.42**
$r3live$_06	38.82	**20.80**	27.92	**13.08**	57.43	**33.88**
eee_01	**5.23**	5.43	38.66	**11.15**	31.02	**16.58**
eee_02	×	**5.41**	30.04	**9.69**	24.60	**15.10**
eee_03	**5.00**	6.93	27.43	**10.00**	23.29	**16.93**
nya_01	6.58	**6.02**	19.62	**7.94**	19.67	**13.96**
nya_02	**6.49**	8.68	20.07	**7.81**	19.87	**16.49**
nya_03	**6.98**	7.22	20.04	**8.02**	20.34	**15.24**
sbs_01	**5.10**	5.69	24.30	**8.97**	21.31	**14.66**
sbs_02	5.52	**5.48**	26.74	**9.31**	23.36	**14.79**
sbs_03	5.26	**5.11**	26.47	**10.24**	22.92	**15.35**

Denotations: "×" means the system drifts halfway through the corresponding sequence.

C. Ablation Study of Camera-Parameter Optimization

In our system, pose estimation and map reconstruction tasks are conducted by the LIO module, and the camera-parameter optimization module's impact is exclusive to the rendering performance. In this section, we evaluate the effectiveness of the camera-parameter optimization module by comparing the PSNR and SSIM value of SR-LIVO with and without the optimization of camera-parameters. Results in Table V demonstrate that optimizing camera-parameters can further improve the rendering performance slightly. Given that the offline-calibrated camera-parameters from the NTU_VIRAL and $R3Live$ datasets are already generally precise, the online optimization of camera-parameters does not yield substantial improvements in the rendering metrics.

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2024.3389415

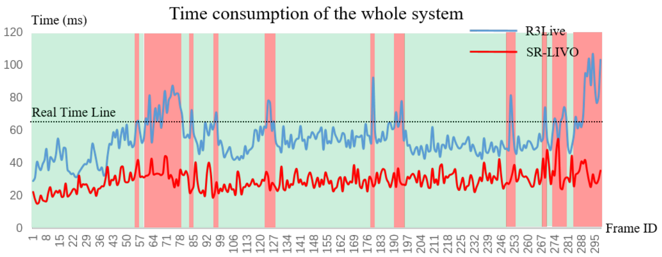


Fig. 4. The curve of time consumption as the frame ID changes on sequence $r3live$_01. R3Live cannot ensure the real-time performance for a considerable amount of time, while our SR-LIVO can run in real time stably.

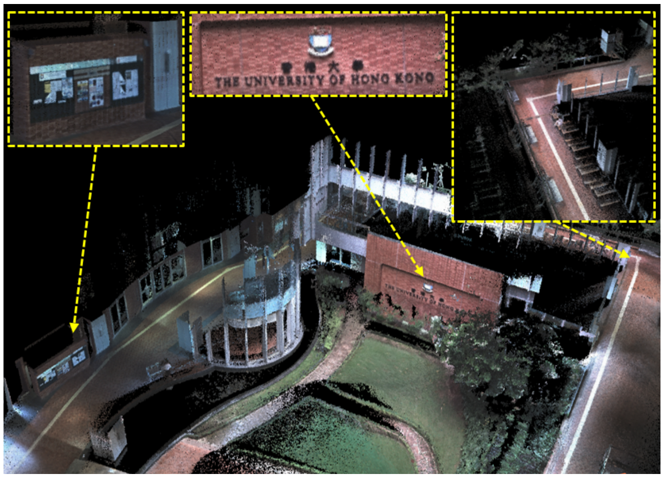
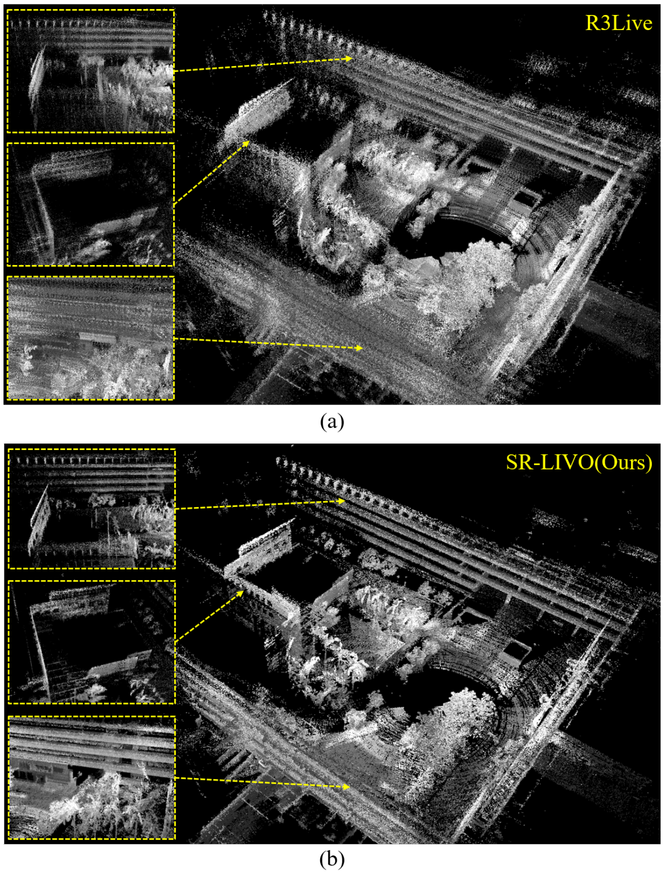


Fig. 5. Our SR-LIVO is able to reconstruct a dense, 3D, RGB-colored point cloud map, which is comparable to the reconstruction result of R3Live (Fig. 1 in [5]).



(a)

(b)

Fig. 6. (a) and (b) are the reconstructed grayscale map on eee_01 by R3Live and SR-LIVO respectively, while our system achieves significantly better reconstruction result.

D. Time Consumption

We evaluate the runtime breakdown (unit: ms) of our system and R3Live for all testing sequences. In general, the whole system framework consists of the vision module and the LiDAR module. For each sequence, we test the time consumption of the above two modules, and the total time for handling a sweep. Results in Table VI show that our system takes 30∼34ms to handle a sweep on $R3Live$ dataset and 14∼17ms to handle a sweep on NTU_VIRAL dataset, while R3Live takes 49∼58ms to handle a sweep on $R3Live$ dataset and 10∼31ms to handle a sweep on NTU_VIRAL dataset. That means our system can run around 1.6X faster than R3Live.

E. Real-Time Performance Evaluation

We take $r3live$_01 as the exemplar sequence, and plot the curve of time consumption as the frame ID changes. Fig. 4 demonstrates that R3Live cannot ensure the real-time performance for a considerable amount of time, while our SR-LIVO can run in real time stably.

F. Visualization for Map

Fig. 5 shows the ability of our SR-LIVO to reconstruct a dense, 3D, RGB-colored point cloud map on the exemplar sequence (e.g., $r3live$_01), which is comparable to the reconstruction result of R3Live (Fig. 1 in [5]). Under the premise that the visualization result is equal, our method can run stably in real time while R3Live cannot, demonstrating the strength of our approach.

The camera images of NTU_VIRAL dataset are grayscale images, leading to the grayscale map. Fig. 6 compares the visualizations of our reconstructed grayscale map with R3Live on the exemplar sequence (e.g., eee_01) of NTU_VIRAL dataset, on which our system achieves significantly better reconstruction result.

VII. Conclusion

This paper proposes a novel LIV-OAM system, named SR-LIVO, which adapts the sweep reconstruction method to align the end timestamp of reconstructed sweep to the timestamp of captured image. Thus, the state of all image-captured moments can be solved by the LIO module, which is more accurate than existing LiDAR-assisted VIO on scenarios without failure modes of camera and LiDAR. In SR-LIVO, we utilize an ESIKF to solve state in LIO module, and utilize an ESIKF to optimize camera-parameters in vision module respectively for optimal state estimation and colored point cloud map reconstruction.

Our system achieves state-of-the-art pose accuracy on two public datasets, and achieves much lower time consumption than R3Live while keeping the reconstruction result comparable or better. Future work includes adding loop closing module to this framework.

References

[1] P. Dellenbach, J.-E. Deschaud, B. Jacquet, and F. Goulette, "Ct-icp: Real-time elastic lidar odometry with loop closure," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 5580–5586.

[2] F. Gao, W. Wu, W. Gao, and S. Shen, "Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments," *Journal of Field Robotics*, vol. 36, no. 4, pp. 710–733, 2019.

[3] F. Kong, W. Xu, Y. Cai, and F. Zhang, "Avoiding dynamic small obstacles with onboard sensing and computation on aerial robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7869–7876, 2021.

[4] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE intelligent vehicles symposium (IV)*. IEEE, 2011, pp. 163–168.

[5] 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 *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10 672–10 678.

[6] 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 Robotics and Automation Letters*, vol. 6, no. 4, pp. 7469–7476, 2021.

[7] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[8] T.-M. Nguyen, S. Yuan, M. Cao, Y. Lyu, T. H. Nguyen, and L. Xie, "Ntu viral: A visual-inertial-ranging-lidar dataset, from an aerial vehicle viewpoint," *The International Journal of Robotics Research*, vol. 41, no. 3, pp. 270–280, 2022.

[9] G. Peng, Y. Zhou, L. Hu, L. Xiao, Z. Sun, Z. Wu, and X. Zhu, "Vilo slam: Tightly coupled binocular vision–inertia slam combined with lidar," *Sensors*, vol. 23, no. 10, p. 4588, 2023.

[10] T. Qin and S. Shen, "Online temporal calibration for monocular visual-inertial systems. in 2018 ieee," in *RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3662–3669.

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

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

[13] W. Wang, J. Liu, C. Wang, B. Luo, and C. Zhang, "Dv-loam: Direct visual lidar odometry and mapping," *Remote Sensing*, vol. 13, no. 16, p. 3340, 2021.

[14] Y. Wang and H. Ma, "mvil-fusion: Monocular visual-inertial-lidar simultaneous localization and mapping in challenging environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 504–511, 2022.

[15] T. Yin, J. Yao, Y. Lu, and C. Na, "Solid-state-lidar-inertial-visual odometry and mapping via quadratic motion model and reflectivity information," *Electronics*, vol. 12, no. 17, p. 3633, 2023.

[16] Z. Yuan, F. Lang, T. Xu, and X. Yang, "Sr-lio: Lidar-inertial odometry with sweep reconstruction," *arXiv preprint arXiv:2210.10424*, 2022.

[17] Z. Yuan, Q. Wang, K. Cheng, T. Hao, and X. Yang, "Sdv-loam: Semi-direct visual-lidar odometry and mapping," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

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

[19] 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 *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4003–4009.

[20] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "Lic-fusion: Lidar-inertial-camera odometry," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 5848–5854.

[21] X. Zuo, Y. Yang, P. Geneva, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, "Lic-fusion 2.0: Lidar-inertial-camera odometry with sliding-window plane-feature tracking," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5112–5119.

Zikang Yuan received the B.E. degree from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2018. He is currently a 4th year PhD student of HUST, School of Institute of Artificial Intelligence. He has published two papers on ACM MM, three papers on TMM, one paper on TPAMI and one paper on IROS. His research interests include monocular dense mapping, RGB-D simultaneous localization and mapping, visual-inertial state estimation, LiDAR-inertial state estimation, LiDAR-inertial-wheel state estimation and visual-LiDAR pose estimation and mapping.

Jie Deng is currently a 4th year B.E. student of HUST, School of Electronic Information and Communications. Her research interests include LiDAR-inertial state estimation, LiDAR-inertial-wheel state estimation and visual-LiDAR pose estimation and mapping.

Ruiye Ming received the B.E. degree from Zhengzhou University(ZZU), Zhengzhou, China, in 2022. He is currently a 2nd year graduate student of HUST, School of Electronic Information and Communications. His research interests include visual-LiDAR odometry, LiDAR point based loop closing, object reconstruction and deep-learning based depth estimation.

Fengtian Lang received the B.E. degree from Huazhong University of Science and Technology (HUST), Wuhan, China, in 2023. He is currently a 1st year graduate student of HUST, School of Electronic Information and Communications. He has published one paper on IROS. His research interests include LiDAR-inertial state estimation, LiDAR-inertial-wheel state estimation and LiDAR point based loop closing.

Xin Yang received her PhD degree in University of California, Santa Barbara in 2013. She worked as a Post-doc in Learning-based Multimedia Lab at UCSB (2013-2014). She is current Professor of Huazhong University of Science and Technology School of Electronic Information and Communications. Her research interests include simultaneous localization and mapping, augmented reality, and medical image analysis. She has published over 90 technical papers, including TPAMI, IJCV, TMI, MedIA, CVPR, ECCV, MM, etc., co-authored two books and holds 3 U.S. Patents. Prof. Yang is a member of IEEE and a member of ACM.