R³LIVE++: A Robust, Real-Time, Radiance Reconstruction Package With a Tightly-Coupled LiDAR-Inertial-Visual State Estimator

Jiarong Lin, Member, IEEE, and Fu Zhang, Member, IEEE

Abstract—This work proposed a LiDAR-inertial-visual fusion framework termed R³LIVE++ to achieve robust and accurate state estimation while simultaneously reconstructing the radiance map on the fly. R³LIVE++ consists of a LiDAR-inertial odometry (LIO) and a visual-inertial odometry (VIO), both running in real-time. The LIO subsystem utilizes the measurements from a LiDAR for reconstructing the geometric structure, while the VIO subsystem simultaneously recovers the radiance information of the geometric structure from the input images. R³LIVE++ is developed based on R²LIVE and further improves the accuracy in localization and mapping by accounting for the camera photometric calibration and the online estimation of camera exposure time. We conduct more extensive experiments on public and self-collected datasets to compare our proposed system against other state-of-the-art SLAM systems. Quantitative and qualitative results show that R³LIVE++ has significant improvements over others in both accuracy and robustness. Moreover, to demonstrate the extendability of R³LIVE++, we developed several applications based on our reconstructed maps, such as high dynamic range (HDR) imaging, virtual environment exploration, and 3D video gaming.

Index Terms—3D reconstruction, SLAM, sensor fusion, state estimation.

I. INTRODUCTION

Simultaneous localization and mapping (SLAM) is a technology that utilizes a sequence of sensor (e.g., camera, LiDAR, IMU, etc.) data to estimate the sensor poses and simultaneously reconstruct the 3D map of surrounding environments. Since SLAM can estimate poses in real-time, it has been widely applied in localization and feedback control for autonomous robots (e.g., unmanned aerial vehicles [1], [2], automated ground vehicles [3], [4], [5], and self-driving cars [6], [7], [8]). Meanwhile, with the capacity to reconstruct the map in real-time, SLAM is also crucial in various robot navigation, virtual and augmented reality (VR/AR), surveying, and mapping applications. Different applications usually require a different level of mapping details: sparse feature map, 3D dense point cloud map, and 3D radiance map (i.e., a 3D point cloud map with radiance information). For example, the sparse visual feature map is suitable and has been widely used for camera-based localization, where the sparse features observed in images can be used for calculating the camera’s pose [9], [10]. The 3D dense point cloud can capture the geometrical structure of the environment even for tiny objects. Hence it is widely used in robot navigation and obstacle avoidance [2], [11]. Finally, radiance maps containing both geometry and radiance information are used in mobile mapping, AR/VR, video gaming, 3D simulation, and surveying. These applications require both geometric structures and textures to provide virtual environments alike the real world [12], [13].

Existing SLAM systems can be mainly categorized into two classes based on the used sensor: visual SLAM and LiDAR SLAM. Visual SLAM is based on low-cost and SWaP (size, weight, and power)-efficient camera sensors and has achieved satisfactory results in localization accuracy. The rich colorful information measured by cameras also makes the reconstructed map suitable for human interpretation. However, due to the lack of direct accurate depth measurements, the mapping accuracy and resolution of visual SLAM are usually lower than LiDAR SLAM. To be more specific, visual SLAM maps the environments by triangulating disparities from multi-view images (e.g., structure from motion for mono-camera, stereo-vision for stereo-camera), an extremely computationally expensive process that often requires hardware acceleration or server clusters. Moreover, limited by the measurement noises and the baseline of multi-view images, the computed depth accuracy drops quadratically with the measurement distance, making visual SLAM difficult to reconstruct large-scale outdoor scenes. Furthermore, visual SLAM can only work in scenarios with good illuminations and will degenerate in high-occlusion or texture-less environments.

On the other hand, LiDAR SLAM is based on LiDAR sensors. Benefiting from the high measurement accuracy (a few centimeters) and the long measurement range (hundreds of meters) of LiDAR sensors, LiDAR SLAM can achieve much higher accuracy and efficiency on both localization and map reconstruction than visual SLAM. However, LiDAR SLAM easily fails in scenarios...
It is a LiDAR-Inertial-Visual fusion framework that tightly proposes a lightweight and ground-
It is extensively validated in real-world experiments in which each sensor in localization and produce an accurate, textured, and high-resolution 3D map that suffices the needs of various mapping applications. Motivated by this, we propose \textsuperscript{R}3LIVE++, which has the following features:

- It is a LiDAR-Inertial-Visual fusion framework that tightly couples two subsystems: the LiDAR-inertial odometry (LIO) subsystem and the visual-inertial odometry (VIO) subsystem. The two subsystems jointly and incrementally build a 3D radiance map of the environment in real-time. In particular, the LIO subsystem reconstructs the geometric structure by registering new points in each LiDAR scan to the map, and the VIO subsystem recovers the radiance information by rendering pixel colors in each image to points in the map.
- It has a novel VIO design, which tracks the camera pose (and estimates other system states) by minimizing the radiance difference between points from the radiance map and a sparse set of pixels in the current image. The frame-to-map alignment effectively lowers the odometry drift, and the direct photometric error on a sparse set of individual pixels effectively constrains the computation load. Moreover, based on the photometric errors, the VIO is able to estimate the camera exposure time online, which enables the recovery of environment’s true radiance information.
- It is extensively validated in real-world experiments in terms of localization accuracy, robustness, and radiance map reconstruction accuracy. Benchmark results on 25 sequences from an open dataset (the NCLT-dataset) show that \textsuperscript{R}3LIVE++ achieves the highest overall accuracy among all other state-of-the-art SLAM systems (e.g., LVI-SAM, LIO-SAM, FAST-LIO2, etc.) under comparison. The evaluations on R3LIVE-dataset (self-collected) show that our proposed framework is robust to extremely challenging scenarios that LiDAR and/or camera measurements degenerate (e.g., when the device is facing a single texture-less wall). Finally, compared with other counterparts, \textsuperscript{R}3LIVE++ estimates the camera exposure time more accurately and reconstructs the true radiance information of the environment with significantly smaller errors when compared to the measured pixels in images.
- It is, to our best knowledge, the first radiance map reconstruction framework that can achieve real-time performance on a PC equipped with a standard CPU without any hardware or GPU accelerations. The system is completely open-sourced to ease the reproduction of this work and benefit the follow-up research. Based on a set of offline utilities for mesh reconstruction and texturing further developed, the system shows high potential in a variety of real-world applications, such as 3D HDR imaging, physics simulation, and video gaming.

In this chapter, we review existing works related to our method or system, including LiDAR SLAM, visual SLAM, and LiDAR-visual fused SLAM. Due to the large number of existing works, any attempts to give a full review would be incomplete. Hence, we only select the most relevant ones of each branch for review. For a comprehensive review of the SLAM literature, please refer to the work in [14].

### A. LiDAR(-Inertial) Odometry

In recent years, the rapid development of LiDAR technologies has greatly improved the reliability and performance of LiDAR sensors while significantly reducing the cost. This progress has drawn increasing research attention to LiDAR SLAM [15]. Zhang et al. proposed a real-time LiDAR odometry and mapping framework, LOAM [16], which achieved localization through scan-to-scan point registration and mapping through scan-to-map registration. In both registrations, only edge and plane feature points were considered to lower the computation load. To enable the algorithm to run in real-time on computation-limited platforms, Shan et al. [17] proposed a lightweight and ground-optimized LOAM (LeGO-LOAM), which discarded unreliable features in the step of ground plane segmentation. Chanoh et al. in [18] presented a novel map-centric SLAM framework for improving map quality for 3D LiDAR SLAM, which is achieved with a continuous-time trajectory representation and novel loop closure methods. These works [16], [17], [18] were mainly based on multi-line spinning LiDARs. For emerging solid-state LiDARs with irregular scanning and very small FoV, our previous works [15] used direct scan-to-map registration to achieve localization and mapping.

To further improve the accuracy and robustness of LiDAR odometry, many frameworks that fused LiDAR measurements with inertial sensors were proposed. In LOAM [16], an IMU
could be used to de-skew the LiDAR scan and provide a motion prior for the scan-to-scan registration. It was a loosely-coupled method since the IMU bias (and the full state vector) was not involved in the scan registration process. Compared with loosely-coupled methods, tightly-coupled methods showed higher robustness and accuracy, thus drawn increasing research interest recently. Authors in [19] proposed LIOM, which used a graph optimization based on priors from LiDAR-Inertial odometry and a rotation-constrained refinement method. Compared with the former algorithms, LIO-SAM [20] optimized a sliding window of keyframe poses in a factor graph to achieve higher accuracy. Similarly, Li et al. proposed LiLi-OM [21] for both multi-line and solid-state LiDARs based on a sliding window optimization technique. LINS [22] was the first tightly-coupled LIO that solved the 6 DOF ego-motion via iterated Kalman filtering. To lower the high computation load in calculating the Kalman gain, FAST-LIO [23] proposed a new formula for the Kalman gain computation. The resultant computation complexity depended on the state dimension instead of the measurement dimension. Its successor FAST-LIO2 [24] further improved the computation efficiency by proposing an incremental k-d tree. Such a data structure could significantly reduce the time cost of nearest points search and allow the registration of raw points (instead of feature points, such as planes and edges, in past works). The method using raw points was termed as a "direct" method and could exploit subtle features in the environment, thus increasing the localization accuracy and robustness.

The LIO subsystem of R³LIVE++ is largely based on FAST-LIO2 [24] since it achieves the best overall performance among its counterparts in terms of accuracy, efficiency, and robustness. Moreover, to address the LiDAR degeneration problem and further improve the localization accuracy, we fuse the LIO subsystem with our VIO subsystem in a tightly-coupled manner.

B. Visual(-Inertial) Odometry

Depending on how a camera measurement is formulated, we review the works of visual odometry by categorizing them into two branches based on the criteria proposed in [25]: indirect and direct. These two types of methods have very different pipelines: the former one (indirect method) includes feature extraction, data association, and minimization of feature re-projection error. In contrast, the latter one (direct method) directly minimizes the photometric error (or intensity discrepancy) between consecutive images.

Indirect visual odometry is also called the feature-based method, which had a quite long history. MonoSLAM [26] proposed by Davison et al. was the first monocular visual odometry, which recovered the 3D trajectory of a camera in real-time by creating a sparse but persistent map of natural landmarks within a probabilistic framework. PTAM [27] proposed by Klein and Murray split the tracking and mapping in parallel threads. Visual landmarks in the map were selected from only a few frames to allow efficient bundle-adjustment (BA) optimization that estimated the camera pose and landmark position. Following this idea, a more complete and reliable framework ORB-SLAM [28] was proposed. ORB-SLAM utilized the same feature (i.e., ORB feature) for all the involved tasks, including tracking, mapping, relocalization, and loop closing. Its further work ORB-SLAM2 [29] improved the accuracy by utilizing the metric scale provided by stereo or RGB-D cameras. The scale issue in pure visual SLAM could also be addressed by fusing inertial sensor data, as demonstrated by VINS-Mono [30], Kimera [31], and ORB-SLAM3 [32], which achieve high-accuracy localization by fusing IMU measurements and image features in a sliding window bundle adjustment optimization.

Direct visual odometry is also called photometric-based method, which minimized the intensity differences rather than a geometric error. It was successfully applied in 2D sparse feature tracking (e.g., Lucas–Kanade optical flow [33]) and then extended to visual odometry, LSD-SLAM [34], proposed by Engel et al. was a direct monocular odometry algorithm with both tracking and mapping directly operating on image intensities. It incrementally tracked the camera pose using direct image alignment and simultaneously performed a pose graph optimization to keep the entire camera trajectory globally consistent. In DSO [25], authors proposed a fully direct probabilistic model that integrated a full photometric calibration. By incorporating a photometric bundle adjustment, the system outperformed other state-of-the-art works in terms of both accuracy and robustness. To achieve real-time performance on a standard CPU, the authors also exploited the sparsity structures of the corresponding Hessian matrix. While the photometric model provided accurate pose estimation over short-term tracking without data association, the geometric model gave robustness for a large baseline. Hybrid approaches that used both photometric and geometric errors were proposed, with the most representative work SVO [35], proposed by Forster et al. where the short-term tracking was solved by minimizing the photometric error, while the long-term drift was constrained by a windowed bundle adjustment on visual landmarks.

There have been many discussions in the literature to answer the question: Which is better? While it is difficult to answer this question now, it is true that the direct method often shows better short-term performance in low-textured environments [25], [32]. Besides, the direct method is often more computationally efficient due to the removal of feature extraction [35]. To leverage these advantages, R³LIVE++ uses a photometric-based VIO subsystem. Unlike the pure visual (or visual-inertial) direct odometry systems, which perform bundle adjustment on photometric errors [25] or feature reprojection errors [35] to restrain long-term drift, the VIO in R³LIVE++ makes full use of the geometry structure reconstructed from LiDAR point cloud by minimizing the radiance errors between map points and image pixels. Such a frame-to-map alignment effectively lowers the odometry drift at a low computation cost. Moreover, pure visual (or visual-inertial) direct methods construct photometric errors on dense images [34] or a sparse set of image patches [25], [35], while the photometric errors of R³LIVE++ VIO subsystem are on a sparse set of individual pixels. Furthermore, the VIO in R³LIVE++ accounts for the camera photometric calibration (e.g., non-linear response function and lens vignetting) and estimates the camera exposure time online, which helps improve the odometry accuracy and recovers the true radiance information of the environment.
C. LiDAR-Visual Fused Odometry

On the basis of LiDAR-inertial methods, LiDAR-inertial-visual odometry incorporating measurements from visual sensors shows higher robustness and accuracy. Zhang and Singh in [36] proposed a LiDAR-inertial-visual system that used a loosely-coupled VIO as the motion model to initialize the LiDAR mapping subsystem. Similarly, Shao et al. in [37] proposed a stereo visual-inertial LiDAR SLAM that incorporated the tightly-coupled stereo VIO with LiDAR mapping and LiDAR-enhanced visual loop closure. The overall system was still a loosely-coupled fusion since the LiDAR data were not jointly optimized along with the visual-inertial measurements.

There are also some RGB-D-inertial odometry, such as [37], [38]. Designed for RGB-D cameras, these methods are difficult to be applied on LiDARs due to the significant differences in the measurement pattern, range, and density between RGB-D cameras and LiDAR sensors. In [39], LiDAR measurements were used to provide depth information for camera images at each frame, forming a system similar to RGB-D camera and hence being able to leverage existing visual SLAM works such as ORB-SLAM2 [29]. This was also a loosely-coupled method as it ignored the direct constraints imposed by LiDAR measurements.

For the works mentioned above, the measurement of LiDAR and camera was fused in a loosely-coupled manner. To achieve higher accuracy and robustness, frameworks that fused sensor data in a tightly-coupled way were proposed in recent years. Zuo et al. [40] proposed a LIC-fusion framework combining IMU measurements, sparse visual features, and LiDAR plane and edge features with online spatial and temporal calibration based on the MSCKF framework. The system exhibited higher accuracy and robustness than other state-of-the-art methods in their experiment results. Later on, their further work termed LIC-Fusion 2.0 [41] refined a novel plane-feature tracking algorithm across multiple LiDAR scans within a sliding window to make LiDAR scan matching more robust. David et al. in [42] proposed a multi-sensor fusion framework named VILENS that tightly-coupled the visual, inertial, legged, and LiDAR sensor data using a novel factor graph. The fusion of four different sensor modalities makes the system achieve better accuracy in localization and makes it robust to challenging scenarios even in case of extreme scenarios (e.g., structure-less and/or texture-less environments). Moreover, the above LiDAR-inertial-visual systems mainly focused on the localization part and has very limited consideration on the mapping efficiency and accuracy. Hence, their visual and LiDAR subsystem often maintains two separate maps for the LIO and VIO, preventing the data fusion at a deeper level and the reconstruction of high-accuracy colored 3D maps. R^3LIVE++ is designed to perform both localization and radiance map reconstruction in real-time. The central of these two tasks is a single radiance map shared among and maintained by both LIO and VIO. In particular, the LIO subsystem reconstructs the geometric structure of the map and the VIO subsystem recovers the radiance information of the map.

This paper is an extension of the previously published work R^3LIVE [45]. The extended works of this paper include 1) a full incorporation of the camera photometric calibration, which corrects the camera nonlinear response function and lens vignetting effect; 2) an online estimation of the camera exposure time. The estimated exposure time and the camera photometric calibration enables the system to recover the true radiance information of the environment; 3) a more comprehensive evaluation of the system on both open and self-collected dataset in terms of localization accuracy, robustness and radiance map reconstruction accuracy; and 4) release of the system codes, associated dataset, and the in-house designed hardware devices for collecting this dataset.

III. BASIC MODELS

A. Notations

In this paper, we use notations shown in Table I.

B. System Overview

To simultaneously estimate the sensor pose and reconstruct the environment radiance map, we design a tightly-coupled LiDAR-inertial-visual sensor fusion framework, as shown in
Fig. 2. The overview of our proposed system.

Fig. 3. The image formation process of a color camera.

The radiance is the point radiance consisting of three independent channels (i.e., red, green, and blue channel) accounting for the camera photometric calibration (see Section III-C). Besides, we also record other necessary information about this point, such as the $3 \times 3$ matrix $\Sigma_p$ and $\Sigma_\gamma$, which denote the covariance of the estimation errors of $Gp$ and $\gamma$, respectively, and the timestamps when this point was created and updated.

2) Voxel: For fast and efficient retrieving of points in the radiance map, such as for VIO update in Section V-C and map point radiance recovery in Section V-D, we organize map points inside the fixed-size voxels (e.g., with dimensions of $0.1m \times 0.1m \times 0.1m$). If a voxel has points appended recently (e.g., in recent 1 s), we mark this voxel as deactivated. Otherwise, this voxel is marked as activated.

C. Color Camera Photometric Model

A camera observes the radiance of the real world in the form of images that consists of 2D arrays of pixel intensities. In our work, we model the image formation process of a camera based on [47] and further extend the gray camera model to a color camera. As shown in Fig. 3, for a point $P$ in the world, it reflects the incoming lights emitted from a light source (e.g., the sun). The reflected lights then pass through the camera lens and finally arrive at the CMOS sensor, which records the intensity of the reflected lights and creates a pixel channel in the output image. The recorded intensity is determined by the radiance, a measure of the power reflected at the point $P$.

To model the above imaging process, we denote $\gamma$ the radiance at point $P$. Since a color camera has three channels in its CMOS sensor: red, green, and blue, the radiance $\gamma$ has three components: $\gamma_r, \gamma_g, \gamma_b$, respectively. For each channel $i$, the lights passing through the camera lens have power $O_i(\rho)$ as:

$$O_i(\rho) = V(\rho)\gamma_i$$

where $V(\rho) \in [0, 1]$ is called the vignetting factor accounting for the lens vignetting effect. Since the vignetting effect is different at different areas of the lens, the vignetting factor $V(\rho)$ is a function of the pixel location $\rho$.

$O_i(\rho)$ is the amount of power that can be received by the sensor and is called the irradiance. When taking an image, the captured irradiance $O(\rho)$ is integrated over time (i.e., the exposure time $\tau$). The accumulated irradiance $\theta_i = \tau V(\rho)\gamma_i$
is then converted as the output of pixel intensity $I_i(\rho)$ via the camera response function (CRF) $f_i(\cdot)$:

$$I_i(\rho) = f_i(\tau V(\rho)\gamma_i), \quad I_i \in [0, 255]. \quad (3)$$

Since a real camera sensor has a limited dynamic range and the physical scale of the radiance $\gamma$ can not be recovered anyway, the pixel intensities can be normalized within $[0, 1]$ without loss of generality.

As noted in (3), different channels often have different non-linear response functions (CRF) $f_i(\cdot)$ and they can be calibrated offline along with the vignetting factor $V(\rho)$ based on the method in [47]. The exposure time $\tau$ is estimated online in our work. With the calibration and estimation results, the radiance of point $P$ from the observed pixel value $I(\rho)$ can be computed as:

$$\gamma_i = \frac{f_i^{-1}(I_i(\rho))}{\tau V(\rho)}. \quad (4)$$

Remark: Under the assumption of constant continuous light sources and a Lambertian reflection model, the radiance at point $P$ is a constant physical value that is invariant to the camera pose. Such invariance to time and camera pose enables us to infer the camera ego-motion from the radiance difference between the map and the current image (with photometric calibration), as detailed in Section V-C.

D. State

In our work, we define the full state $x$ as:

$$x = (G^{\vec{R}}_{\vec{f}}, G^{\vec{p}}_{\vec{f}}, G^{\vec{v}}, b_{\vec{g}}, b_{\vec{n}}, G^{\vec{g}}, x_{\vec{C}}, t_{\vec{C}}, \epsilon, t_{\vec{C}}, \phi) \quad (5)$$

where the notations $G^{\vec{R}}_{\vec{f}}, G^{\vec{p}}_{\vec{f}}, G^{\vec{v}}, b_{\vec{g}}, b_{\vec{n}}, G^{\vec{g}}, x_{\vec{C}}, t_{\vec{C}}$ are explained in Table I, $t_{\vec{C}}$ is the time-offset between IMU and camera while LiDAR is assumed to be synced with the IMU already, $\epsilon = 1/\tau$ is the inverse camera exposure time, $\phi = [f_x, f_y, c_x, c_y]^T$ are the camera intrinsics, where $(f_x, f_y)$ denote the camera focal length and $(c_x, c_y)$ the offsets of the principal point from the top-left corner of the image plane. The camera extrinsic $([R_{\vec{C}}, t_{\vec{C}}])$, intrinsic $\phi$ and time-offset $t_{\vec{C}}$ would usually have their rough values available (e.g., offline calibration, CAD model, manufacturer’s manual). To cope with the possible calibration errors (e.g., extrinsic $([R_{\vec{C}}, t_{\vec{C}}]$ and intrinsic $\phi$) or online drifting (e.g., time-offset $t_{\vec{C}}$), we also include them in the state $x$ such that they will be estimated online. Besides, we also estimate the camera exposure time online in order to recover the true radiance value of each map point.

In $R^3$LIVE++, our LiDAR-Inertial odometry (LIO) and visual-Inertial odometry (VIO) take advantage of the measurements data from LiDAR, IMU, and camera sensors, continuously update the full state $x$ for achieving that goal of optimal state estimation.

IV. LiDAR-INERTIAL ODOMETRY (LIO)

Our LIO subsystem reconstructs the geometry structure of the environment by registering each new LiDAR scan to the global map. We use the generalized-iterative closest point (GICP) method [48] to iteratively estimate the LiDAR pose (and other system states) by minimizing the distance of each point in the scan to a plane fitted from the corresponding points in the map. The estimated state estimate is then used to append the new points to the map.

A. LiDAR Point-to-Plane Residual

As shown in Fig. 2, our LIO subsystem constructs the geometric structure of the global map.

For the $k$-th input LiDAR scan, we first compensate the in-frame motion with an IMU backward propagation introduced in [23]. Let $L_k = \{l_1^t, \ldots, l_m^t\}$ be the set of $m$ LiDAR points after motion compensation, we compute the residual of each raw point (or a downsampled subset) of $l_s^t \in L_k$ where $s$ is the index of point and the superscript $L$ denotes that the point is represented in the LiDAR-reference frame.

With $x_k$ being the estimate of $x_k$ at the current iteration, we transform $l_s^t$ from LiDAR frame to the global frame:

$$G_p^s = G_{l_k}(l^t_R L_s^t p_s + l^t_p) + G_{l_k} \quad (6)$$

To register the point to the global map, we search for the nearest five points in the map. To accelerate the nearest neighbor search, map points are organized into an incremental k-d tree (see [24]). If the euclidean distance between $G_p^s$ and the searched five nearest neighbor points is smaller than a threshold (e.g., 0.4 m), these found nearest neighbor points are used to fit a plane with normal $u_s$ and centroid $q_s$. Then, the LiDAR measurement residual $r_l(x_k, l^t_p) =$:

$$r_l(x_k, l^t_p) = u_s^T (G_p^s - q_s). \quad (7)$$

B. LIO ESIKF Update

The residual in (7) should be zero ideally. However, due to the estimation error in $x_k$ and the LiDAR measurement noise, this residual is often not zero and can be used to refine the state estimate $x_k$. Specifically, let $x_n$ be the measurement noise of the point $l^t_p$, we have the relation between the true point location $l^t_p$ and the measured one $l^t_p$ as below:

$$l^t_p = l_p^{gt} + n_s, n_s \sim N(0, \Sigma_{n_s}) \quad (8)$$

This true point location together with the true state $x_k$ should lead to zero residual in (7), i.e.,

$$0 = r_l(x_k, l^t_p^{gt}) \approx r_l(x_k, l^t_p) + H_s^x \delta x_k + \alpha_s \quad (9)$$

where $x_k$ is parameterized by its error $\delta x_k$ in the tangent space of $x_k$ (i.e., $x_k = x_k + \delta x_k$). $\alpha_s \sim N(0, \Sigma_{\alpha_s})$ is the lumped noise due to $n_s$, and $H_s^x$ is the Jacobian of the residual w.r.t. $\delta x_k$.

Equation (9) constitutes an observation distribution for $x_k$ (or equivalently $\delta x_k \equiv \delta x_k$), which is combined with the prior distribution from the IMU propagation:

$$\min_{\delta x_k} \left( \| r_l(x_k, l^t_p) + H_s^x \delta x_k \|_{\Sigma_{\alpha_s}}^2 \right) \quad (10)$$

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
where $\|x\|_2^2 = x^T \Sigma^{-1} x$ is the squared Mahalanobis distance with covariance $\Sigma$, $\hat{x}_k$ is the IMU propagated state estimate, $\Sigma_{\delta \hat{x}_k}$ is the IMU propagated state covariance. The first item essentially represents $\|x \parallel \delta \hat{x}_k\|_2^2$, which incorporates the prior information from the IMU propagation.

Solving (10) leads to the Maximum A-Posteriori (MAP) estimate of $\delta \hat{x}_k^*$ which is then added to $\hat{x}_k$ as below:

$$\hat{x}_k \leftarrow \hat{x}_k + \delta \hat{x}_k^*$$ (11)

The above iteration process is iterated until convergence (i.e., the update $\delta \hat{x}_k^*$ is smaller than a given threshold). The converged state estimate $\hat{x}_k$ is then used as the starting point of the IMU propagation until the reception of the next LiDAR scan or camera image. Furthermore, the converged estimate $\hat{x}_k$ is used to append points in the current LiDAR scan to the global map as follows. For the $s$-th point $p_s \in L_k$, its position in global frame $G p_s$ is first obtained by (6). If $G p_s$ has nearby points in the map with distance 1 cm (see Section III-B-1), $G p_s$ will be discarded to maintain a spatial resolution of 1 cm. Otherwise, a new point structure $P_s$ will be created in the map with:

$$P_s = \left[ G p_s^T, \gamma_s^T \right]^T = \left[ G p_s^T, 0 \right]^T$$ (12)

where the radiance vector $\gamma_s$ is set as zero and will be initialized at the first time it is observed in forthcoming images (see Section V-D). Finally, we mark the voxel containing $G p_s$ as activated such that the radiance of points in this voxel can be updated by the forthcoming images (see Section V-D). This is because, in most LiDAR-camera platforms, such as those used in LVI-SAM [43], FAST-LIVO [49], and our system, LiDAR, and cameras are typically installed closely to achieve a larger field of view overlap. This configuration results in the points within “activated” voxels being unobserved to the current camera. Consequently, we can efficiently identify and retrieve the points that are unoccluded to the current camera since such points would also appear in current LiDAR measurements (hence labeled as “activated” for visual fusion).

V. VISUAL-INERTIAL ODOMETRY (VIO)

While our LIO subsystem reconstructs the geometric structure of the environment, our VIO subsystem recovers the radiance information from the input color images. To be more specific, our VIO subsystem projects a certain number of points (i.e., tracked points) from the global map to the current image, then it iteratively estimates the camera pose (and other system states) by minimizing the radiance error of these points. Only a sparse set of tracked map points is used for the sake of computation efficiency.

Our proposed framework is different from previous photometric-based methods [35], [50], which constitute the residual of a point by considering the photometric error over all its neighborhood pixels (i.e., a patch). These patch-based methods achieve stronger robustness and faster convergence speed than those without. However, the patch-based method is not invariant to either translation or rotation, which requires estimating the relative transform when aligning one patch to another. Plus, the calculation of the residual is not completely precise by assuming the depths of all pixels in the patch are the same as the mid-point. On the other hand, our VIO is operated at an individual pixel, which utilizes the radiance of a single map point to compute the residual. The radiance, which is updated simultaneously in the VIO, is an inherent property of a point in the world and is invariant to both camera translation and rotation. To ensure a robust and fast convergence, we design a two-step pipeline shown in Fig. 2. Specifically, in the first step (i.e., frame-to-frame VIO update), we leverage a frame-to-frame optical flow to track map points observed in the last frame and obtain a rough estimate of the system’s state by minimizing the Perspective-n-Point (PnP) reprojection error of the tracked points (Section V-B). Then, in the second step (i.e., frame-to-map VIO), the state estimate is further refined by minimizing the difference between the radiance of map points and the pixel intensities at their projected location in the current image (Section V-C). With the converged state estimate and the raw input image, we finally update the map points radiance according to the current image measurement (Section V-D).

A. Photometric Correction

For each incoming image $I$, we first correct the image non-linear CRF $f_{\gamma}(\cdot)$ and the vignetting factor $V(\cdot)$, which are calibrated in advance (see Section III-C), to obtain the photometrically corrected image $\Gamma$, whose $i$-th channel at pixel location $p$ is:

$$\Gamma_i(p) = \frac{f_{\gamma}^{-1}(I_i(p))}{V(p)}.$$ (13)

The photometrically corrected image $\Gamma$ is then used in the following VIO pipelines including the frame-to-frame VIO, frame-to-map VIO and radiance recovery.

B. Frame-to-Frame Visual-Inertial Odometry

As depicted in Fig. 4, we leverage Lucas – Kanade optical flow for tracking a certain number (e.g., 400) of map points (i.e., tracked points). These tracked points are initially created by projecting map points onto the first camera images while maintaining a minimum distance (e.g., 50 pixels) between adjacent points. These tracked map points are then employed in both the frame-to-frame and frame-to-map VIO update processes. Following the VIO update, the tracked points are updated accordingly, as will be introduced in Section V-E.

1) Perspective-N-Point Reprojection Error: Assume we have tracked $m$ map points $P = \{p_1, \ldots, p_m\}$ in the last image frame $I_{k-1}$ with their projected location in $I_{k-1}$ being $\{p_{1, k-1}, \ldots, p_{m, k-1}\}$, we leverage the Lucas – Kanade optical flow to find out their corresponding location in the current image $I_k$, denoted as $\{p_1, \ldots, p_m\}$. Then, we iteratively minimize the reprojection errors of $P$ to obtain a rough estimate of the state (see Fig. 5). Specifically, taking the $s$-th point $P_s = \left[ p_s^T, \gamma_s^T \right]^T \in P$ as an example, let $x_k$ be the state estimate at the current iteration, the projection error $r_c(x_k, p_s, G p_s)$ is

$$r_c(x_k, p_s, G p_s) = p_s - \pi(G p_s, x_k)$$ (14)
that is the Jacobian of the residual w.r.t. \( \sim \mathcal{N} \) and Fig. \( \in \mathbf{G} \). Among these tracked points, the solid black dots represent the new points that were added in the last image frame \( \mathbf{I}_{k-1} \).

\[ \Delta \text{ (see and } n \mathbf{G} \text{ and } = r \text{ and the camera measurement } = \text{ moves at a constant velocity of } \text{to the image plane to obtain its } \]

is related to camera pose, extrinsic, intrinsic, and exposure time, we project \( \in \mathbf{G} \) by assuming that the projected position of \( \text{is the standard camera pin-hole model } \) in \( \text{is small (e.g., } \text{is the lumped noise due to } \) can be computed from the \( \in \mathbf{G} \).

\[ \text{is the time interval between the last and current } \text{is the predicted pixel location computed as below:} \]

where \( \pi(G \mathbf{p}_s, x_k) \in \mathbb{R}^2 \) is the predicted pixel location computed as below:

\[ \pi(G \mathbf{p}_s, x_k) = \pi_{\phi h}(G \mathbf{p}_s, x_k) + t_{C_k} \left( \rho_{s_k} - \rho_{s_{k-1}} \right) / \Delta t_{k-1,k} \] (15)

where the first term \( \pi_{\phi h}(G \mathbf{p}_s, x_k) \) is the standard camera pin-hole model, the second one is the temporal correction factor \[51\], and \( \Delta t_{k-1,k} \) is the time interval between the last and current image. Since \( \Delta t_{k-1,k} \) is small (e.g., \( < 50 \text{ ms} \)), we estimated the value of \( t_{C_k} \) by assuming that the projected position of \( G \mathbf{p}_s \) moves at a constant velocity of \( \left( \rho_{s_k} - \rho_{s_{k-1}} \right) / \Delta t_{k-1,k} \) on the image plane during the interval between two consecutive frames.

2) Frame-to-Frame VIO Update: Similar to the LIO update, the state estimation error in \( x_k \) and the camera measurement noise will lead to a certain residual in (14), from which we can update the state estimate \( x_k \) as follows. First, the measurement noise in the residual (14) consists of two sources: one is the pixel tracking error in \( \rho_{s_k} \) and the other lies in the map point location error \( G \mathbf{p}_s \).

\[ G \mathbf{p}_s = G \mathbf{p}_s^* + n_{\mathbf{p}_s}, n_{\mathbf{p}_s} \sim \mathcal{N}(0, \Sigma_{n_{\mathbf{p}_s}}) \] (16)

\[ \rho_{s_k} = \rho_{s_k}^* + n_{\rho_{s_k}}, n_{\rho_{s_k}} \sim \mathcal{N}(0, \Sigma_{n_{\rho_{s_k}}}) \] (17)

where \( G \mathbf{p}_s^* \) and \( \rho_{s_k}^* \) are the true values of \( G \mathbf{p}_s \) and \( \rho_{s_k} \), respectively. Then, correcting such noises and using the true system state should lead to zero residual, i.e.,

\[ 0 = r_c(x_k, \rho_{s_k}^*, G \mathbf{p}_s^*) \approx r_c \left( x_k, \rho_{s_k}, G \mathbf{p}_s \right) + H_\delta \delta x_k + \beta \] (18)

where \( H_\delta \) is the Jacobian of the residual w.r.t. \( \delta x_k \) and \( \beta \sim \mathcal{N}(0, \Sigma_{\beta}) \) is the lumped noise due to \( n_{\mathbf{p}_s} \) and \( n_{\rho_{s_k}} \).

Equation (18) constitutes an observation distribution for \( x_k \), which is combined with the IMU propagation to obtain the MAP estimate of the state in the same way as the LIO update detailed in Section IV-B. The converged state estimate is then refined in the frame-to-map VIO in the next section.

Remark: Since the camera pin-hole model \( \pi_{\phi h}(G \mathbf{p}_s, x_k) \) in (15) is related to camera pose (consisting of the IMU pose \( (G \mathbf{R}_t, G \mathbf{p}_t) \) and camera extrinsic \( (\mathbf{R}_c, \mathbf{p}_c) \)) and intrinsic \( \phi \), so the projection model \( \pi(G \mathbf{p}_s, x_k) \) is also related to these state components. In addition, \( \pi(G \mathbf{p}_s, x_k) \) is also related to the temporal offset \( t_{C} \) due to the temporal correction factor. This will cause \( H_\delta \) to contain nonzero elements corresponding to the IMU pose \( (G \mathbf{R}_t, G \mathbf{p}_t) \), camera extrinsic \( (\mathbf{R}_c, \mathbf{p}_c) \), intrinsic \( \phi \), and temporal offset \( t_{C} \), and hence an update of them in the state estimation.

C. Frame-to-Map Visual-Inertial Odometry

1) Frame-to-Map Radiance Error: The frame-to-frame VIO update can provide a good state estimate \( x_k \), which is further refined by the frame-to-map VIO update by minimizing the radiance error of the tracked map points \( \mathcal{P} \). Let \( \Gamma_k \) the photometrically calibrated image at the \( k \)-th step (see (13)). With the state estimate at the current iteration, \( x_k \), which contains the estimated camera pose, extrinsic, intrinsic, and exposure time, we project a tracked map point \( \mathbf{p}_s \in \mathcal{P} \) to the image plane to obtain its pixel location \( \rho_{s_k} = \pi(G \mathbf{p}_s, x_k) \) (see (15) and Fig. 6). Then, the observed radiance denoted by \( \Phi_s \) can be computed from the exposure time component \( \epsilon_k \) in \( x_k \) as: \( \Phi_s = \epsilon_k \Gamma_k(\rho_{s_k}) \). Finally, the frame-to-map radiance error is the difference between the radiance component \( \gamma_s \) of the point \( \mathbf{p}_s \) and the observed value

Fig. 4. The colored “tadpole”-shaped dots represent the tracked points \( \mathcal{P} \) that are used for VIO updates, which are tracked using the Lucas – Kanade optical flow. The head position of each map point corresponds to its location in the current image frame \( \mathbf{I}_k \), while the tail position corresponds to its location in the previous image frame \( \mathbf{I}_{k-1} \). Among these tracked points, the solid white dots represent historical tracked points that have been tracked in several previous frames, while the solid black dots represent the new points that were added in the last image frame \( \mathbf{I}_{k-1} \).

Fig. 5. Frame-to-frame VIO estimates the system’s state by minimizing the position reprojection error of map points observed in the last frame.

Fig. 6. Frame-to-map VIO refines the state estimate by minimizing the radiance error between the map point and the observed radiance in the current image.
\( \Phi_s \):
\[
\mathbf{r}_c(\mathbf{x}_k, \mathbf{G}_s, \gamma_s) = \Phi_s - \gamma_s, \quad \Phi_s = \bar{\epsilon}_k \Gamma_k(\bar{\rho}_s),
\]
where \( \Phi_s, \Gamma_k(\bar{\rho}_s) \) and \( \gamma_s \) both contain three channels: red, green, and blue.

2) Frame-to-Map VIO Update: The measurement noise in (19) come from both the component \( \gamma_s \) and \( \Phi_s \). For the component \( \gamma_s \), we model it as:
\[
\gamma_s = \gamma_s^{gt} + n_{\gamma_s} + n_{\mathbf{ic}}, \quad n_{\gamma_s} \sim \mathcal{N}(0, \Sigma_{\gamma_s}) \\
\mathbf{n}_{\mathbf{ic}} \sim \mathcal{N}(0, \sigma^2_{\mathbf{ic}} \cdot \Delta t_{\gamma_s})
\]
where \( \gamma_s^{gt} \) represents the ground-truth value of \( \gamma_s \). The terms \( n_{\gamma_s} \) originate from the radiance estimation error as detailed in Section V-D, and \( n_{\mathbf{ic}} \) arises from radiance temporal changes due to variations in illumination. Given that illumination changes usually occur gradually over time, we model this process as a random walk [52], whose covariance is linear to the time interval \( \Delta t_{\gamma_s} \) between current time and last update time of \( \mathbf{P}_s \). Since \( n_{\gamma_s} \) and \( n_{\mathbf{ic}} \) are indeed two independent noise, we have the Gaussian distribution of \( \gamma_s \) as \( \gamma_s \sim \mathcal{N}(\gamma_s^{gt}, \Sigma_{\gamma_s} + \sigma^2_{\mathbf{ic}} \cdot \Delta t_{\gamma_s}) \).

For the second component \( \Phi_s \), in Section IV, it is computed from the state estimate \( \mathbf{x}_k \) and the current image \( \Gamma_k \) as \( \Phi_s = \bar{\epsilon}_k \Gamma_k(\pi(G_p, \mathbf{x}_k)) \), hence its noise consists of two sources: one is the measurement error (from \( \mathbf{x}_k \)) and the other is the image measurement noise (from \( \Gamma_k \)):
\[
\Phi_s = \Phi_s^{gt} + n_{\Phi_s}, \quad \Phi_s^{gt} \sim \mathcal{N}(0, \Sigma_{\Phi_s})
\]
where \( \Sigma_{\Phi_s} \) denotes the covariance due to these two noise sources.

Combining (19), (20) and (21), we obtain the first order Taylor expansion of the true zero residual \( \mathbf{r}_c(\mathbf{x}_k, \mathbf{G}_s, \gamma_s) \):
\[
0 = \mathbf{r}_c(\mathbf{x}_k, \mathbf{G}_s, \gamma_s) \approx \mathbf{r}_c(\mathbf{x}_k, \mathbf{G}_s, \mathbf{c}_s) + \mathbf{H}_s^r \Delta \mathbf{x}_k + \zeta_s
\]
where \( \mathbf{H}_s^r \) is the Jacobian of the residual w.r.t. \( \Delta \mathbf{x}_k \) and \( \zeta_s \sim \mathcal{N}(0, \Sigma_{\zeta}) \) is the lumped noise due to noises in \( \gamma_s \) and \( \Phi_s \).

Similar as before, (22) constitutes an observation distribution for state \( \mathbf{x}_k \), which is combined with the IMU propagation to obtain the MAP estimate of the state.

Remark: Since the \( \Phi_s \) in (19) is related to camera exposure time \( \epsilon \), it will cause \( \mathbf{H}_s^r \) to contain nonzero elements corresponding to the exposure time and hence an update of them in the state estimation.

D. Recovery of Radiance Information

After the frame-to-map VIO update, we have the precise pose of the current image. Then, we perform the Bayesian update to determine the optimal radiance of all map points such that the average radiance error between each point and its viewed images is minimal.

First of all, we retrieve all the points in all activated voxels (activated in Section IV-B). Assume the retrieved point set is \( \mathcal{Q} = \{ \mathbf{q}_1, \ldots, \mathbf{q}_n \} \). For the \( s \)-th point \( \mathbf{q}_s = [G_p_s, \gamma_s] \in \mathcal{Q} \) falling in the current image FoV, we first can obtain the observed radiance vector \( \Phi_s \) by (19) and its covariance \( \Sigma_{\Phi_s} \) by (21). To prevent the underestimation of the radiance vector caused by underexposure or overexposure, we exclude points in \( \mathcal{Q} \) from the update process if their pixel values in any RGB channel are the minimum or maximum value (i.e., 0 or 255).

If \( \mathbf{P}_s \) is a new point appended by the LIO subsystem (see Section IV-B) with \( \gamma_s = 0 \), we set:
\[
\gamma_s = \Phi_s, \quad \Sigma_{\gamma_s} = \Sigma_{\Phi_s}
\]
Otherwise, the radiance vector \( \gamma_s \) saved in the map (see (20)) is fused with newly observed radiance vector \( \Phi_s \) with covariance \( \Sigma_{\Phi_s} \) via Bayesian update (see Fig. 7):
\[
\Sigma_{\gamma_s} = (\Sigma_{\gamma_s} + \sigma^2_{\mathbf{ic}} \cdot \Delta t_{\gamma_s})^{-1} + \Sigma_{\Phi_s}^{-1}
\]
\[
\hat{\gamma}_s = (\Sigma_{\gamma_s} + \sigma^2_{\mathbf{ic}} \cdot \Delta t_{\gamma_s})^{-1} \gamma_s + \Sigma_{\Phi_s}^{-1} \Phi_s
\]
\[
\gamma_s = \hat{\gamma}_s, \quad \Sigma_{\gamma_s} = \Sigma_{\Phi_s}
\]

E. Update of the Tracking Points

After the recovery of radiance information, we update the tracked point set \( \mathcal{P} \) for the next frame of image use. First, we remove points from current \( \mathcal{P} \) if their projection error in (14) or radiance error in (19) are too large, and also remove the points which do not fall into the current image FoV. Second, we project each point in \( \mathcal{Q} \) to the current image and add it to \( \mathcal{P} \) if no other tracked points already existed in a neighborhood of 50 pixels.

VI. EXPERIMENTS

In this chapter, we conduct extensive experiments to validate the advantages of our proposed system against other counterparts in threefold: 1) To verify the accuracy in localization, we quantitatively compare our system against existing state-of-the-art SLAM systems on a public dataset (NCLT-dataset). 2) To evaluate the robustness of our framework, we test it under various challenging scenarios where camera and LiDAR sensor degeneration occurs. 3) To evaluate the accuracy of our system in reconstructing the radiance map, we compare it against existing baselines in estimating the camera exposure time and calculating the average photometric error w.r.t. each image. In the experiments, two datasets are used for evaluation: the NCLT-dataset and the R^3LIVE-dataset.

A. NCLT-Dataset

To compare the accuracy of our proposed method against other state-of-the-art SLAM systems, we perform quantitative
evaluations on NCLT-dataset [53]. NCLT-dataset is a large-scale, long-term autonomy dataset for robotics research that was collected on the University of Michigan’s North Campus. The dataset is comprised of 27 sequences that are collected by exploring the campus, both indoors and outdoors, on varying paths, and at different times of the day across all four seasons. Each sequence includes data from the omnidirectional camera, 3D LiDAR, planar LiDAR, GPS, and wheel encoders on a Segway robot.

We chose NCLT-dataset for three reasons: 1) NCLT-dataset is currently the largest public dataset with ground-truth trajectories of high quality. 2) NCLT-dataset provides all raw data sampled by the sensors, which meet our requirement for the input data. 3) NCLT-dataset has many challenging scenarios, such as moving obstacles (e.g., pedestrians, bicyclists, and cars), illumination changes, varying viewpoints, seasonal and weather changes (e.g., falling leaves and snow), and long-term structural changes caused by construction projects. For details of challenging input (e.g., falling leaves and snow), and long-term structural changes caused by construction projects. For details of challenging input data from both LiDAR and camera sensors, please refer to Section 1 of our Supplementary Material [54].

In the experiments, the front-facing camera data (one of five) and the 3D LiDAR data are used for all systems under evaluation. Moreover, we notice some time synchronization errors in two sequences (i.e., 2012-03-17 and 2012-08-04), where the LiDAR timestamp is 100 ms delayed from the IMU timestamp (about one LiDAR-frame). Therefore, we exclude these two sequences from the evaluation. As a result, 25 sequences are evaluated with total traveling length up to 138 km and duration up to 33h:34m.

B. Self-Collected Dataset: R$^3$LIVE-Dataset

While the large-scale NCLT-dataset is suitable for evaluating the localization accuracy, it didn’t cover any scenarios with sensor degeneration, preventing us from evaluating the system robustness, which is one of the major motivations of this work. Moreover, the camera photometric calibration and the ground-truth exposure time are not available in the NCLT-dataset, which are essential for the reconstruction of the radiance maps and the evaluation of the online exposure time estimation. To fill this gap, we designed a handheld data collection device and made a new dataset named R$^3$LIVE-dataset. The dataset and hardware device are released along with the codes of this work to facilitate the reproduction of our work.

1) Handheld Device for Data Collection: Our handheld device for data collection is shown in Fig. 8(a), which includes a power supply unit, an onboard computer DJI manifold-2c (equipped with an Intel i7-8550u CPU and 8GB RAM), a FLIR Blackfly BFS-u3-13y3c global shutter camera, and a LiVIO AVIA 3D LiDAR. The camera FoV is $82.9^\circ \times 66.5^\circ$ and the LiDAR FoV is $70.4^\circ \times 77.2^\circ$. To quantitatively evaluate the accuracy of our algorithm (Section VI-E) even in GPS-denied environments, we use an ArUco marker [55] as a reference to calculate the sensor pose when returns to the starting point, which enables to evaluate the localization drift. All the mechanical components of this device are designed for compatibility with fused deposition modeling (FDM) 3D printing technology. Their design schematics are open-sourced along with the codes too [56].

To correct the camera’s nonlinear response function and the vignette effect, we perform photometric calibration on the camera based on the method in [47]. The calibrated results are shown in Fig. 9, which are also released on our GitHub repository [56].

2) The R$^3$LIVE-Dataset: The R$^3$LIVE-dataset was collected within the campuses of the University of Hong Kong (HKU) and the Hong Kong University of Science and Technology (HKUST). As summarized in Table II, the dataset includes 13 sequences that are collected by exploring both indoor and outdoor environments, in various scenes (e.g., walkway, park, forest, etc) at different time in a day (i.e., morning, noon, and evening). This allows the dataset to capture both structured urban buildings and cluttered field environments with different lighting conditions. The dataset also includes three sequences (degenerate_seq_00/01/02) where the LiDAR or camera (or both) degenerate by occasionally facing the device to a single and/or texture-less plane (e.g., wall, the ground) or visually. The total traveling length reaches 8.4 km, duration reaching 2.4h. More details of each sequence will be provided in sequel when it is used.

C. System Configurations

For the sake of fair comparison, in the evaluation of our systems and their counterparts, each system uses the same parameters for all sequences in the same dataset. For the counterpart systems (e.g., LIO-SAM [20], LVI-SAM [43], FAST-LIVO [49], FAST-LIO2 [24], etc.), we use
their default configurations on their GitHub repository except for some necessary adjustments to match the hardware setup. For our system, we also make its configuration available on our GitHub repository [56], “https://github.com/hku-mars/r3live/blob/master/config/r3live_config.yaml” for NCLT-dataset and “https://github.com/hku-mars/r3live/blob/master/config/r3live_config.yaml” for R³LIVE-dataset.

D. Experiment-1: Evaluation of Localization Accuracy

In this experiment, we benchmark the localization accuracy of our systems against other state-of-the-art odometry, including LIO-SAM [20], LVI-SAM [43], FAST-LIVO [49], FAST-LIO2 [24], and our previous work R²LIVE [44], on the NCLT-dataset [53]. LIO-SAM and FAST-LIO2 are LiDAR-inertial systems without fusing image data (Section II-A), while LVI-SAM, FAST-LIVO, and R²LIVE are three other state-of-the-art LiDAR-inertial-visual systems. Since our work is a state estimator without any loop detection and correction, we deactivated the loop closure of LIO-SAM and LVI-SAM for the sake of fair comparison. Due to the unavailability of camera photometric calibration for the NCLT-dataset, we disable the photometric calibration modules of our VIO-subsystem by using \( V(\cdot) = 1 \) and \( f_1(\cdot) = 1 \).

Table III shows the absolute position error (APE) [57] of these methods, where Our-LIO is the LIO subsystem of our system. LIO-SAM and LVI-SAM failed in some sequences, and these sequences are excluded from the computation of the average APE. Since our method (see Fig. 2) and R²LIVE [44] output three types of odometry, one at LiDAR input, one at camera input, and one at the IMU input, while the other tested methods (e.g., LIO-SAM, FAST-LIO) outputs odometry only at LiDAR input. To ensure fairness, we compute the APE all using the odometry output at LiDAR input, even for our method and R²LIVE [44]. As can be seen from this table, with the average APE only 8.51m, our proposed system achieves the best overall performance than other LiDAR-inertial-visual systems FAST-LIVO, R²LIVE, and LVI-SAM. The performance improvement mainly comes from the direct method used in the LIO subsystem and the tight-coupling of the LIO and VIO subsystems, the former can be seen by comparing the direct method FAST-LIO2 to the feature-based method LIO-SAM in Table III (and also detailed in [24]), the latter improves the accuracy of the VIO subsystem (hence the complete system) by leveraging the high-accuracy geometry structure reconstructed from the LiDAR. Furthermore, the overall APE of our system is lower than its LIO subsystem Our-LIO and the other LIO systems (i.e., FAST-LIO2 and LIO-SAM), which confirms the effectiveness of fusing camera data. Indeed, we found that in some evaluated sequences, the LiDAR sensor may occasionally face the sun. This can lead to the generation of noisy LiDAR points due to Sunlight contamination [58] as shown in Fig. 1 of our Supplementary Material [54], which adversely affects the accuracy of the LIO. This is due to the input images that are under-exposure (a ∼ d), over-exposure, as well as the presence of moving objects, as detailed in Fig. 2 of our Supplementary Material [54], which may adversely affect the VIO (hence the overall system). In Fig. 11, we overlay all the 25 ground-true trajectories (in the left figure) and ours (the right one). As can be seen, the overlaid trajectories estimated by our system agree with the ground-truth well and each trajectory can still be clearly distinguished without noticeable errors. Note that these 25 trajectories are collected on the same campus area across different times in a day and seasons in a year, still our system can produce consistent and reliable trajectory estimation with these illumination and scene changes, demonstrating the robustness of our system.

Given the fact that the photometric calibration of NCLT-dataset is unavailable, to assess the impact of photometric calibration on localization accuracy of our method, we compared its performance with and without photometric calibration using the R³LIVE-dataset, referred to as “Our” and “Our w/o PC” in Table IV, respectively. We calculated the end-point positioning errors by comparing our results with the ground-truth estimated from an ArUco marker [55]. As shown in Table IV, photometric calibration has a minor impact on our localization outcomes on 5 sequence we tested, where the observations from our LIO and frame-to-frame VIO update primarily contribute to

### Table II
**Overview of the R³LIVE-Dataset**

| Sequence               | Duration (s) | Traveling Length (m) | Sensor Degeneration | Return to origin | ArUco marker | Camera exposure time | Scenarios     |
|------------------------|--------------|----------------------|--------------------|------------------|--------------|----------------------|---------------|
| degenerate_seq_00      | 101          | 74.9                 | Camera, LiDAR      | ✓                | ✓            | Indoor               | Indoor        |
| degenerate_seq_01      | 86           | 53.3                 | LiDAR              | ✓                | ✓            | Outdoor              | Outdoor       |
| degenerate_seq_02      | 85           | 75.2                 | LiDAR              | ✓                | ✓            | Outdoor              | Outdoor       |
| hku-campus_seq_00      | 202          | 190.6                | 304                | ✓                | ✓            | Indoor               | Indoor        |
| hku-campus_seq_01      | 304          | 374.6                | ✓                  | ✓                | ✓            | Outdoor              | Outdoor       |
| hku-campus_seq_02      | 323          | 354.3                | ✓                  | ✓                | ✓            | Indoor, Outdoor      | Indoor, Outdoor |
| hku-campus_seq_03      | 173          | 181.2                | ✓                  | ✓                | ✓            | Indoor, Outdoor      | Indoor, Outdoor |
| hku_main_building      | 1170         | 1036.9               | ✓                  | ✓                | ✓            | Indoor, Outdoor      | Indoor, Outdoor |
| hkuパーク_00            | 351          | 401.8                | ✓                  | ✓                | ✓            | Outdoor, Cluttered   | Outdoor, Cluttered |
| hkuパーク_01            | 226          | 247.3                | ✓                  | ✓                | ✓            | Outdoor, Cluttered   | Outdoor, Cluttered |
| hkusat_campus_00       | 1073         | 1317.2               | ✓                  | ✓                | ✓            | Indoor, Outdoor      | Indoor, Outdoor |
| hkusat_campus_01       | 1162         | 1524.3               | ✓                  | ✓                | ✓            | Indoor, Outdoor      | Indoor, Outdoor |
| hkusat_campus_02       | 478          | 503.8                | ✓                  | ✓                | ✓            | Indoor, Outdoor      | Indoor, Outdoor |
| hkusat_campus_03       | 1618         | 2112.2               | ✓                  | ✓                | ✓            | Outdoor              | Outdoor       |
| **Total**              | **7354**     | **8447.6**           |                    |                  |              |                      |               |

1. Sequences are collected by traveling a loop, with starting from and ending with the same position.
2. Sequences with ArUco marker [55] for providing the ground-truth relative pose.
3. Sequences with ground-truth camera exposure time read from camera’s API.
E. Experiment-2: Evaluation of Robustness

Besides illumination and scene change, we also test the robustness of our system to extreme scenarios where sensor degeneration occurs. We use the R³LIVE-dataset, which contains such extreme scenarios.

1) Evaluation of Robustness in LiDAR Degenerated Scenarios:
In this experiment, we evaluate the robustness of our proposed system by testing our system on the sequence "degenerate_seq_00" and "degenerate_seq_01" of R³LIVE-dataset (see Section VI-B2). These two sequences were collected in front of a stairway with the LiDAR occasionally facing against the ground and a side wall (see Fig. 10(a) and (b)). When facing a wall, the LiDAR only observes a single plane, which is insufficient to determine the LiDAR pose, causing LiDAR degeneration. The device starts from and ends at the same location, enabling the localization accuracy. Yet, photometric calibration is necessary for recovering the accurate radiance map of the environment, as will be illustrated in Section VI-F.

Fig. 10. Robustness evaluation in LiDAR degenerated environments.
Fig. 11. The overlay of ground-truth trajectory and ours on NCLT-dataset. The accompanying video of R$^3$LIVE++ in reconstructing the large-scale radiance map of a sequence from the NCLT dataset is available on YouTube: https://youtu.be/kZ8_7k3Hplk?t=309.

Fig. 12. Tests in simultaneously LiDAR degenerated and visual texture-less environments.

Fig. 13. shows our estimated pose, with the phases of passing through “wall-1” and “wall-2” shaded with purple and yellow, respectively. The estimated covariance is also shown in Fig. 13, which is bounded over the entire estimated trajectory, indicating that our estimation quality is stable over the entire process. The sensor is moved to the starting point, where an ArUco marker board is used to obtain the ground-true relative pose between the starting and end poses. Compared with the ground-true end pose, our algorithm drifts $1.62^\circ$ in rotation and $4.57$ cm in translation. We recommend the readers to the accompanying video on YouTube (https://youtu.be/kZ8_7k3Hplk?t=440) for better visualization of the experiment.

F. Experiment-3: Evaluation of Radiance Map Reconstruction

In this experiment, we evaluate the accuracy of our proposed algorithm in reconstructing the radiance map. Since the ground-true radiance map of the environment can not be measured, we evaluate the accuracy based on two indicators: one is the estimation quality of the camera exposure time and the other is the average photometric error between the reconstructed radiance map and the measured images.

1) Evaluation of Exposure Time Estimation: In this experiment, we evaluate the accuracy of the estimated camera exposure time by comparing it with the ground-true value read from
In this experiment, we evaluate the accuracy of our proposed algorithm in reconstructing the radiance map. Currently, LiDAR point cloud colorization remains one of the most challenging problems in the field of 3D reconstruction. The most common way is using the most recent image frame in time to give the color of each LiDAR frame [39], [49]. The preliminary implementation of our system R³LIVE published previously [45] colorized the point cloud by minimizing the photometric error similar to our current system but does not consider any exposure time estimation or photometric calibration. In this experiment, we compare our system against the previous implementation R³LIVE [45] to show the effectiveness of the exposure time estimation and photometric calibration and against the current baseline [39], [49] to show the advantage of the overall system.

To assess the radiance reconstruction error, after the map reconstruction, we re-project all points in the map with radiance information to each image frame with the estimated camera pose and calibrated photometric parameters. Then, we calculate the photometric error between the map point color and the RGB values of the image at the projected pixel location. The average photometric error of each image frame is calculated through both interior and exterior of a complex building to ensure significant changes in lighting conditions. We compare our estimated results with Tum-cali [47] and DSO [25], which are two existing state-of-the-art baselines that can estimate the camera’s exposure time. All of these methods are initialized by assuming the exposure time of the first image frame is at a default value of 1ms.

The results are shown in Fig. 18, where the estimated exposure time of our method, Tum-cali [47], and DSO [25] are re-scaled to match with the ground-truth for better visualization. The average and maximum error of estimated exposure time w.r.t. the ground-truth is listed in Table V. As shown in Fig. 18 and Table V, our proposed method shows significantly lower estimation error than [47] and [25]. This is mainly because our method estimates the exposure time by minimizing the scan-to-map radiance error, while [47] and [25] recovers the exposure times from consecutive frames. The consequence is that our method can better utilize longer-term temporal intensity changes to restrain the drift of the exposure time estimation.

2) Evaluation of Radiance Map: In this experiment, we evaluate the accuracy of our proposed algorithm in reconstructing the radiance map. Currently, LiDAR point cloud colorization remains one of the most challenging problems in the field of 3D reconstruction. The most common way is using the most recent image frame in time to give the color of each LiDAR

![Image](image.png)

**Table V**

| Sequence               | Our Mean/Max (ms) | Tum-cali Mean/Max (ms) | DSO Mean/Max (ms) |
|------------------------|-------------------|------------------------|-------------------|
| hku campus_seq_02      | 3.460 / 20.311    | 7.182 / 76.217         | 7.505 / 43.198    |
| hku campus_seq_03      | 1.460 / 10.653    | 6.400 / 37.126         | 6.553 / 41.014    |
| hku main_building      | 2.572 / 26.355    | 1.196 / 26.775         | 5.550 / 30.281    |
| hku instasense seq_02  | 0.189 / 1.185     | 0.341 / 1.451          | 2.428 / 17.719    |
| hku instasense seq_03  | 0.302 / 3.514     | 5.225 / 13.361         | 0.381 / 2.040     |

The preliminary implementation of our system R³LIVE published previously [45] colorized the point cloud by minimizing the photometric error similar to our current system but does not consider any exposure time estimation or photometric calibration. In this experiment, we compare our system against the previous implementation R³LIVE [45] to show the effectiveness of the exposure time estimation and photometric calibration and against the current baseline [39], [49] to show the advantage of the overall system.

In this section, we investigate the average time consumption of our proposed system on a CPU-Only PC (equipped with an Intel i7-9700K CPU and 64 GB RAM). Table VII listed the average, maximum, and minimum of sequences’ average time consumption.
Fig. 14. Photometric errors between the reconstructed radiance map and image pixels.

Fig. 15. Closeup of a few scenes in the radiance map reconstructed by the baseline, R³LIVE and R³LIVE++.

Fig. 16. Our reconstructed radiance map of the main building of HKU. (a) The bird’s view of the map, with its details shown in (b ~ n). (b ~ g) closeup of outdoor scenarios and (h ~ n) closeup of indoor scenarios. To see the real-time reconstruction process of the map, please refer to the video on YouTube: https://youtu.be/kZ8_jk3HpIk?t=48.
TABLE VII

| Dataset         | Mean ± STD | Maximum ± STD | Minimum ± STD |
|-----------------|------------|---------------|---------------|
| NCLT-dataset    |            |               |               |
| LiDAR frame (ms)| 34.371 ± 10.551 | 38.153 ± 7.187 | 29.070 ± 8.807 |
| Camera frame (ms)| 16.400 ± 4.168   | 17.613 ± 4.275 | 15.090 ± 4.164 |
| R³LIVE-dataset  |            |               |               |
| LiDAR frame (ms)| 23.453 ± 7.431   | 30.757 ± 5.109 | 8.111 ± 3.622  |
| Camera frame (ms)| 16.244 ± 2.705   | 18.829 ± 3.719 | 12.111 ± 2.502 |

VII. APPLICATIONS WITH R³LIVE

A. High Dynamic Range (HDR) Imaging

After the reconstruction of the radiance map, we are able to render an image by projecting the map to an image plane with a given pose and exposure time with (3). Taking the sequence “hku_main_building” as an example, Fig. 17(a), (b), (c) and (d) are the rendered images with global exposure time of 1 ms, 2 ms, 5 ms and 10 ms, respectively. These images rendered at different exposure times can be merged into an HDR image shown in Fig. 17(e).

B. R³LIVE++ for Various 3D Applications

While R³LIVE++ reconstructs the colored 3D map in real-time, we also develop software utilities to mesh and texture the reconstructed map offline, which are also publicly available on our GitHub repository [56]. With these developed utilities, we can export the reconstructed 3D maps to Unreal Engine [59] for enabling a series of 3D applications such as video games and vehicle simulators. For detailed showcases of our developed applications, we refer our reader to Section 4 of our Supplementary Material [54].
In this paper, we proposed a novel LiDAR-inertial-visual fusion framework termed R^3LIVE++ to achieve robust and accurate state estimation while simultaneously reconstructing the radiance map on the fly. This framework consists of two subsystems (i.e., the LIO and the VIO) that jointly and incrementally build a 3D radiance map of the environment in real-time. By tightly fusing the measurement of three different types of sensors, R^3LIVE++ can achieve higher localization accuracy while being robust enough to scenarios with sensor degenerations.

In our experiments, we extensively validated our proposed algorithm with real-world experiments in terms of localization accuracy, robustness, and radiance map reconstruction accuracy. The benchmark results on 25 sequences from an open dataset (the NCLT-dataset) showed that R^3LIVE++ achieved the highest overall accuracy among all other state-of-the-art SLAM systems under comparison. The evaluation on R3LIVE-dataset (self-collected) showed that the proposed framework was robust to extremely challenging scenarios that LiDAR and/or camera measurements degenerate (e.g., when the device is facing a single texture-less wall). Finally, compared with other counterparts, R^3LIVE++ estimates the camera exposure time more accurately and reconstructs the true radiance information of the environment with significantly smaller errors when compared to the measured values in images.

To demonstrate the extendability of our work, we developed several applications based on our reconstructed radiance maps, such as high dynamic range (HDR) imaging, virtual environment exploration, and 3D video gaming. Finally, to share our findings and make contributions to the community, we made our codes, hardware design, and dataset publicly available on our GitHub repository [56].

B. Future Works

Due to the absence of a loop correction mechanism in R^3LIVE++, there is a possibility of gradual drift caused by the accumulation of localization errors when revisiting the same place. This can potentially result in inconsistent reconstructed outcomes upon revisits. In our forthcoming research, we aim to overcome this limitation by integrating our recent advancements in loop detection, as presented in [60], which is based on LiDAR point clouds. By employing this loop detection mechanism, we could identify loops in real-time and subsequently apply loop corrections to minimize drift and enhance the consistency of the reconstructed results.

Besides, the radiance map in R^3LIVE++ is reconstructed with 3D points that contain radiance information, which prevents us from rendering high-resolution images from the radiance map due to the limited point cloud density of the radiance map (1 cm in our current implementation). While further increasing this point cloud density is possible, it will further increase the processing time. This point density is also limited by the density of the raw points measured by LiDAR sensors. Noticing that images often have much higher resolution, in the future, we could explore how to make full use of such high-resolution images in the fusion framework.

REFERENCES

[1] F. Gao, W. Wu, W. Gao, and S. Shen, “Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments,” J. Field Robot., vol. 36, no. 4, pp. 710–733, 2019.
[2] F. Kong, W. Xu, Y. Cai, and F. Zhang, “Avoiding dynamic small obstacles with onboard sensing and computation on aerial robots,” IEEE Robot. Automat. Lett., vol. 6, no. 4, pp. 7869–7876, Oct. 2021.
[3] H. Latgahin, A. Geiger, and B. Kitt, “Visual SLAM for autonomous ground vehicles,” in Proc. IEEE Int. Conf. Robot. Automat., 2011, pp. 1732–1737.
[4] P. Beinschob and C. Reinke, “Graph slam based mapping for AGV localization in large-scale warehouses,” in Proc. IEEE Int. Conf. Intell. Comput. Commun. Process., 2015, pp. 245–248.
[5] J. Lin, X. Liu, and F. Zhang, “A decentralized framework for simultaneous calibration, localization and mapping with multiple LiDARs,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2020, pp. 4870–4877.
[6] J. Levinson et al., “Towards fully autonomous driving: Systems and algorithms,” in Proc. IEEE Intell. Veh. Symp., 2011, pp. 163–168.
[7] G. Ros, A. Sappa, D. Ponsa, and A. M. Lopez, “Visual slam for driverless cars: A brief survey,” in Proc. Intell. Veh. Symp. Workshops, 2012, pp. 1–6.
[8] A. Singandhupre and H. M. Ha, “A review of slam techniques and security in autonomous driving,” in Proc. 3rd IEEE Int. Conf. Robotic Comput., 2019, pp. 602–607.
[9] O. Pink, “Visual map matching and localization using a global feature map,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops, 2008, pp. 1–7.
[10] M. Bürkl, M. Dymczyk, I. Gilitschenski, C. Cadena, R. Siegwart, and J. Nieto, “Map management for efficient long-term visual localization in outdoor environments,” in Proc. IEEE Intell. Veh. Symp., 2018, pp. 682–688.
[11] B. Nagy and C. Benedek, “Real-time point cloud alignment for vehicle localization in a high resolution 3D map,” in Proc. Eur. Conf. Comput. Vis. Workshops, 2018, pp. 226–239.
[12] W. Li et al., “AADS: Augmented autonomous driving simulation using data-driven algorithms,” Sci. Robot., vol. 4, no. 28, 2019, Art. no. eaaq0863.
[13] Z. Bao, S. Hossain, H. Lang, and X. Lin, “A review of high-definition map creation methods for autonomous driving,” Eng. Appl. Artif. Intell., vol. 122, 2023, Art. no. 106125.
[14] C. Cadena et al., “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” IEEE Trans. Robot., vol. 32, no. 6, pp. 1309–1332, Dec. 2016.
[15] J. Lin and F. Zhang, “Loam_livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small FoV,” in Proc. IEEE Int. Conf. Robot. Automat., 2020, pp. 3126–3131.
[16] J. Zhang and S. Singh, “LOAM: Lidar odometry and mapping in real-time,” in Proc. Robot.: Sci. Syst. Conf., vol. 2, no. 9, 2014, pp. 1–9.
[17] T. Shan and B. Englot, “LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2018, pp. 4758–4765.
[18] C. Park, P. Moughamad, J. L. Williams, S. Kim, S. Sridharan, and C. Fookes, “Elasticity meets continuous-time: Map-centric dense 3D LiDAR SLAM,” IEEE Trans. Robot., vol. 38, no. 2, pp. 978–997, Apr. 2022.
[19] H. Ye, Y. Chen, and M. Liu, “Tightly coupled 3D LiDAR inertial odometry and mapping,” in Proc. Int. Conf. Robot. Automat., 2019, pp. 3144–3150.
[20] 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.
[21] K. Li, M. Li, and U. D. Hanebeck, “Towards high-performance solid-state-LiDAR-inertial odometry and mapping,” IEEE Robot. Automat. Lett., vol. 6, no. 3, pp. 5167–5174, Jul. 2021.

Acknowledgments

The authors would like to thank DJI Company, Ltd. for providing devices and research funds.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Jiarong Lin (Member, IEEE) received the BS degree in optical information science and technology from the University of Electronic Science and Technology of China (UESTC), in 2015. He is currently working toward the PhD degree with the Department of Mechanical Engineering, the University of Hong Kong (HKU), Hong Kong, China. His research interests include light detection and ranging (LiDAR) mapping and sensor fusion.

Fu Zhang (Member, IEEE) received the BE degree in automation from the University of Science and Technology of China (USTC), Hefei, Anhui, China, in 2011, and the PhD degree in controls from the University of California at Berkeley, Berkeley, CA, USA, in 2015. He joined the Department of Mechanical Engineering, the University of Hong Kong (HKU), Hong Kong, as an assistant professor, in August 2018. His current research interests are on robotics and controls, with a focus on unmanned aerial vehicle (UAV) design, navigation, control, and light detection and ranging (LiDAR)-based simultaneous localization and mapping (SLAM).