HVL-SLAM: Hybrid Vision and LiDAR Fusion for SLAM

Wei Wang, Chenjie Wang, Jun Liu, Xin Su, Bin Luo, and Cheng Zhang

***Abstract*— In the field of simultaneous localization and mapping (SLAM), map-based localization has been widely used in autonomous driving, particularly for all-speed and all-road adaptive cruise, automatic parking, and other high-level functions. As a result, LiDAR sensors are frequently used in visual-based SLAM to improve the overall accuracy of ego-motion estimation and environment reconstruction. In this article, a novel tightly coupled monocular hybrid visual LiDAR SLAM (HVL-SLAM), which utilizes both visual and LiDAR measurements in tracking and mapping. First, the proposed method reduces the 3-D uncertainty of features by employing object segmentation and Delaunay triangulation. The motion between adjacent frames is then estimated using a hybrid tracking module that minimizes photometric and reprojection error. Finally, a joint optimization method for refining the pose is proposed, which incorporates visual and LiDAR measurements into optimization with dynamic weights, resulting in higher positioning accuracy and robustness. The experiments on the public KITTI odometry benchmark and real-world outdoor datasets demonstrate that HVL-SLAM outperforms state-of-the-art approaches in terms of pose estimation and mapping performance. The code is released to the community. Code available at https://github.com/kinggreat24/hvl_slam.**

***Index Terms*— Feature depth extraction, hybrid pose estimation, joint visual LiDAR pose optimization, visual LiDAR simultaneous localization and mapping (SLAM).**

I. Introduction

DUE to its wide applications in areas such as autonomous driving, intelligent robotics, and virtual reality, simultaneous localization and mapping (SLAM) has been an active research topic. State-of-the-art SLAM systems can be divided into three categories based on the sensors, they contain: visual SLAM, LiDAR SLAM, and multisensor fusion SLAM. The fusion of multiple sensors in SLAM, such as LiDAR and vision fused SLAM, can compensate for the inherent shortcomings of a single sensor. This article focuses on the integration of visual and LiDAR measurements in SLAM systems.

The standard visual LiDAR SLAM methods, which are broadly classified as depth-enhanced visual approaches and LiDAR registration-based approaches, combine complementary information from another sensor to improve the single-sensor VO/SLAM system. Depth-enhanced visual methods use accurate depth information from LiDAR to augment the visual odometry pipeline, resulting in improved ego-motion estimation accuracy and less reliance on light illumination. Visual feature-based methods, such as DEMO [1], LIMO [2], CamVox [3], and RGB-L [4], assign LiDAR-determined depth values to corresponding visual feature points. Unlike visual feature-based methods, direct-based visual methods, such as DVL-SLAM [5] and the method proposed in [6], use depth value from LiDAR in direct camera motion tracking. LiDAR registration-based approaches [7], [8] compensate LiDAR SLAM using image information in specific dimensions, which can improve robustness in failure-prone scenarios like LiDAR-degenerate, sharp turns, and aggressive motions.

These visual-LiDAR SLAM methods primarily provide additional information to a single LiDAR or visual SLAM framework in order to improve its performance, but they do not fully utilize all sensor measurements. For example, visual-enhanced approaches select only LiDAR point clouds corresponding to visual features, discarding the majority of LiDAR information, whereas LiDAR registration-based approaches either do not include visual measurements in loop closing or do not use LiDAR measurements in visual SLAM.

To maximize the benefits of multisensor information fusion, tightly coupled fusion methods are proposed to improve the overall SLAM's accuracy and robustness. TVLO [9] creates maps from two different sensors independently, calculates the residuals of LiDAR geometric and stereo reprojection separately, and then combines them to estimate ego-motion. Some researches [10], [11] combine visual and LiDAR measurements in motion estimation and loop closing, yielding impressive pose estimation results. However, stereo cameras in these stereo-based fusion methods, such as TVL-SLAM [11], require a long baseline for accurate long-depth estimation, which is frequently limited in real-world scenarios. Furthermore, the calibration between the two cameras is susceptible to mechanical variations, which can reduce depth estimation accuracy. As a result, we intend to perform ego-motion estimation by tightly integrating a monocular camera and a 3-D LiDAR.

Manuscript received 25 September 2023; revised 30 January 2024 and 2 June 2024; accepted 8 July 2024. Date of publication 22 July 2024; date of current version 26 September 2024. This work was supported in part by Wuhan University, Wuhan, China, through the National Key Research and Development Program of China under 2022YFB3903404; and in part by the National Natural Science Foundation of China under Grant 62371348 and Grant 42230108. *(Wei Wang and Chenjie Wang contributed equally to this work.) (Corresponding author: Jun Liu.)*

Wei Wang, Jun Liu, Bin Luo, and Cheng Zhang are with the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430072, China (e-mail: kinggreat24@whu.edu.cn; luob@whu.edu.cn; liujunand@whu.edu.cn; zhangchengzc@whu.edu.cn).

Chenjie Wang is with the Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230026, China (e-mail: wangchenjie@iai.ustc.edu.cn).

Xin Su is with the School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430072, China (e-mail: xinsu.rs@whu.edu.cn).

Digital Object Identifier 10.1109/TGRS.2024.3432336

As a crucial element in tightly coupled methodologies, numerous depth completion techniques [2], [4], [10], [12] have been developed to accurately determine the depth values of visual features. In these methods, the depth prior for each feature point is estimated separately using a neighboring patch. Specifically, LiDAR points within a patch are first segmented into foreground and background points. The foreground points are then used to fit a plane, and the depth of the feature point is obtained by intersecting the point ray with the plane. However, such methods are constrained by the patch size. Furthermore, the density of the laser point cloud has a significant impact on the depth fitting results. If the point cloud is sparse, the results of foreground and background segmentation and spatial plane fitting will be highly uncertain.

Beyond the mapping issues, the positioning based on the generated map is a critical component of the SLAM system, particularly in GPS-denied environments or situations where GPS cannot provide precise localization results. Most localizations [13], [14], [15] using only LiDAR maps are easily diverged in repeated structural scenes, such as urban canyons, tunnels, underground parking lots, and so on. Furthermore, those methods require initial global localization, which is still difficult to obtain due to the lack of salient features in LiDAR range data.

In contrast, the vision-based approaches show better efficiency and performance in initial global localization because of the rich angular resolution and informative color information, such as NetVlad [16], DBoW [17], and HF-Net [18]. Furthermore, even in the most difficult conditions, HF-Net [18] achieves incomparably fast yet robust and accurate localization performance with the help of a global visual map. The visual map, on the other hand, is generated offline, which takes time, and its scale may not be referenced to an absolute scale because an absolute depth scale is unknown to a monocular camera.

To that end, we propose in this article a fusion mapping system that tightly integrates visual and LiDAR sensors for accurate pose estimation and efficient robot localization without the need for GPS or a geotagged database. The following are the proposed system's main contributions.

1) An efficient and robust depth extraction method for feature points (see Section III-B) is performed based on the point cloud object segmentation and Delaunay triangles. It does not necessitate complex parameter settings and can function well even when the LiDAR range data is limited.
2) A hybrid visual LiDAR fusion tracking module (see Section III-C) is proposed to improve the robustness of ego-motion estimation, particularly in degenerate scenes.
3) A visual LiDAR joint optimization approach (see Section III-D) is devised with dynamic weights derived from both visual and LiDAR measurements. Nonlinear least-square error is used to maintain local accuracy while dynamically adjusting the weights of residuals from different factors.
4) We devise a method for rapidly generating a visual-LiDAR fusion map with feature points from both sensors, and we validate its superior performance with relocalization tasks. Fig. 1 depicts an example of fusion map construction using hybrid visual LiDAR SLAM (HVL-SLAM).

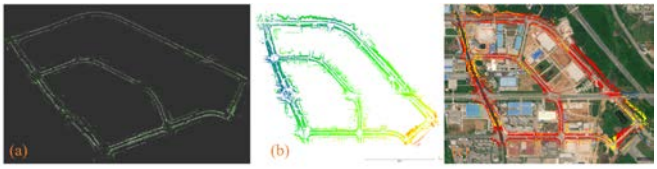

Fig. 1. Mapping results on large outdoor environment. (a) Fusion map. (b) LiDAR feature map, where colors are rendered by the altitude of points. (c) Map aligned with Google map. More details can be seen in *https://youtu.be/hWXjsMqWPy4*.

II. Related Work

The state-of-the-art works of visual LiDAR odometry can be divided into three categories: depth-enhanced visual approaches, LiDAR registration-based approaches, and tightly coupled fusion methods.

A. Depth-Enhanced Visual Approaches

The visual-enhanced method tries to use a visual odometry pipeline with known feature depth data from laser scans. The DEMO [1], for example, was proposed to improve visual odometry by assigning depth measurements from a LiDAR to visual features. Following works, such as LIMO [2], CamVox [3], and PLVL-SLAM [12], use bundle adjustment techniques to achieve significantly higher motion estimation accuracy. In contrast to these methods, which combine feature-based visual SLAM with camera tracking, DVL-SLAM [5] employs direct visual SLAM. They use projected laser points as feature points instead of salient gradient points extracted from images. Then, using the feature points' known depth values, a multiframe photometric optimization similar to the DSO [19] is performed to estimate the poses of the keyframes. The method in [6] proposed a direct laser-visual odometry approach based on the photometric image alignment method. This method forecasts occlusions using planar information and uses two-stage registration to obtain precise frame-by-frame self-motion estimates. However, one common problem with depth-enhanced visual approaches is that they do not account for laser spots that are outside the camera's field of view. Most range readings will be ignored with sensors like Velodyne LiDAR, which can provide a 360° sweep. In this case, it may make the system less accurate and more susceptible to the effects of texture-less situations.

B. LiDAR Registration Based Approaches

In contrast to depth-enhanced visual methods, LiDAR registration-based approaches attempt to align the entire point cloud in various aspects using image information. For example, Zhang et al. proposed V-LOAM [7], which simply uses the visual odometry result as an initial guess to the iterative closest point (ICP) process, reducing the likelihood of the ICP being trapped in local minima. Recent research indicates that redundant odometry can significantly improve the robustness of ego-motion estimation. Reinke et al. [8] proposed

an integrated visual-LiDAR odometry system that employs multiple odometry algorithms in parallel. Sanity checks and a Chamfer distance-based score are used to choose the most promising estimation. Besides, images can be used for loop closure detection as well as assisting the laser in initial motion calculations. For example, Park et al. [20] presented a novel framework, named ElasticLiDAR++, for multimodal map-centric SLAM. The loop closure detection module keeps generating 2-D features from the visual sensor and compares them to the previously generated keyframes.

C. Tightly Coupled Fusion Methods

Vision-driven approaches use only a small percentage of LiDAR point clouds to enhance visual features, ignoring the majority of LiDAR measurements. To investigate the benefits of each sensor and compensate for the disadvantages of the other, TVLO [9] collected both visual and LiDAR measurements, stacking and minimizing the residuals of both modalities during the pose optimization phase, and avoiding the potential problems of assigning LiDAR depths to noncorresponding visual features. Following that, TVL-SLAM [11] extends this method to a SLAM system that incorporates visual and LiDAR measurements in motion estimation, loop detection, loop closing, and extrinsic calibration, resulting in impressive pose estimation results. TVL-SLAM is a stereo-based approach, whereas ours is monocular. The visual and LiDAR sensors are run independently in the frontend, while all of the visual and LiDAR measurements are incorporated in the backend optimizations. The above systems usually have high accuracy when estimating the sensor poses. However, the real-time performance of these systems in real-world mobile robot applications is not satisfactory, especially when mobile robot computation resources are limited.

He et al. [10] proposed a collaborative mapping system that combined visual LiDAR odometry for robust and accurate pose estimation with a pose graph-based collaborative SLAM pipeline to produce high-quality mapping results. More recently, Wang et al. [21] proposed DV-LOAM, which is a direct Visual-LiDAR fusion framework. The system begins with a two-staged direct visual odometry module for efficient coarse state estimation, then refines the coarse pose with the LiDAR mapping module, and finally uses the loop-closure module to correct the accumulated drift. Yuan et al. [22] proposed SDV-LOAM, a system that integrates semi-direct LiDAR-assisted depth-enhanced visual odometry with LiDAR odometry for precise and robust pose estimation and mapping. In addition to point features, some tightly coupled fusion methods [23], [24] improve accuracy by fusing multiple types of features. However, most existing tightly coupled fusion methods use LiDAR point cloud projection to establish point or surface correspondences to assign LiDAR depth to visual feature points. Due to the sparsity of point clouds, fitting errors, and significant depth differences between foreground and background, the effectiveness and efficiency of depth-enhanced visual features in current methods are limited. Furthermore, in the backend optimization process, most existing methods apply the fixed weights for visual and LiDAR points in a least-squares optimization. They do not consider adaptively adjusting the weights of these observations during computation to dynamically adapt to different scenes, thereby achieving better performance. Considering the above issues, we propose a depth fitting method based on object segmentation and Delaunay triangles to achieve fast and robust depth extraction of feature points. Additionally, during joint optimization, an adaptive weighting module is introduced to adjust the weights of visual and LiDAR joint optimization, which not only improves pose accuracy but also optimizes the visual feature map.

III. Method

A. Overview

Each frame $\mathcal{F}_k$ is composed of its corresponding image $\mathcal{I}_k$, LiDAR point cloud $\mathbf{P}_k$, and visual point features $\mathbf{F}_k$. We assume that the image and the LiDAR point cloud are time synchronized, and the external parameters $\mathbf{T}_{\text{LiDAR}}^{\text{Cam}}$ between them are known. Therefore, we can use (1) to project the LiDAR points into the image space coordinate system

$$\mathbf{P}_k^C = \mathbf{T}_{\text{LiDAR}}^{\text{Cam}} \mathbf{P}_k^L. \tag{1}$$

In addition, for a point $P_i \in \mathbf{P}_k^C$, we can use the camera projection function $\pi(\cdot)$ to project it into image plane

$$\begin{aligned} p_i &= \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \pi(P_i) = \frac{1}{Z_i} \times K \times P_i \\ &= \frac{1}{Z_i} \begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} \end{aligned} \tag{2}$$

where $\{f_x, f_y, c_x, c_y\}$ are the intrinsic parameters of the camera.

The pose of each frame $\mathcal{F}$ is represented by a matrix $\mathbf{T}_{\text{c(amera)w(orld)}} \in \text{SE}(3)$. We set the local coordinate of the camera as the body coordinate, and the world coordinate as the beginning of the body coordinate. Thus, the relative pose transformation between $\mathcal{F}_m$ and $\mathcal{F}_n$ can be denoted as

$$\mathbf{T}_m^n = \mathbf{T}_{cw}^n \mathbf{T}_{cw}^{m\ -1} \tag{3}$$

and it also can be used for coordinate transformation of points between the coordinates as follows:

$$\begin{aligned} P_n &= \mathbf{T}_m^n P_m = \exp\left(\hat{\xi}\right) P_m \\ &= \begin{bmatrix} \mathbf{R}_m^n & \boldsymbol{t}_m^n \\ \mathbf{0}^T & 1 \end{bmatrix} P_m \end{aligned} \tag{4}$$

where P_m and P_n are points in each frame, $R_m^n \in \text{SO}(3)$ and $t_m^n \in \mathbb{R}^3$ are a rotation matrix and translation vector, respectively. Lie algebra $\xi = \begin{bmatrix} \phi \\ \rho \end{bmatrix} \in \mathbb{R}^6$, ϕ and ρ are angular and linear velocity vectors after dividing by the time interval, respectively.

Fig. 2 shows the framework of our approach, which is developed based on the ORB-SLAM2. Our system contains three running threads: a frame-to-frame tracking thread, a visual-LiDAR jointly optimization thread, and a loop closure detection thread. The tracking module first extracts the visual point features every frame. Then, a hybrid visual LiDAR

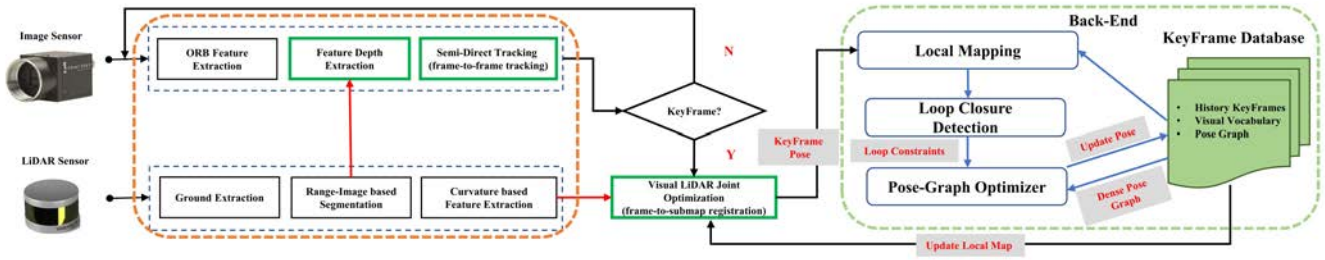

Fig. 2. Overview of the proposed HVL-SLAM method. The components highlighted in green are our main contributions in this work.

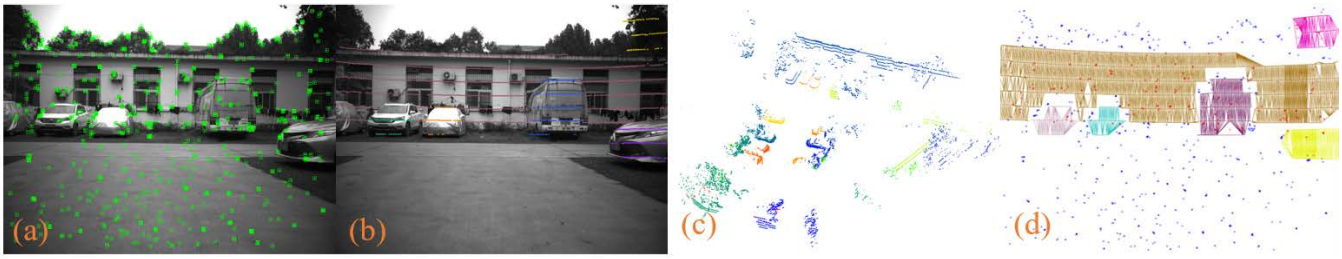

Fig. 3. Illustration of point feature depth extraction. (a) Extracted ORB features. (c) Is LiDAR segmentation result, each color represents an object cluster, while (b) shows the image projection result of object clusters. (d) Object Delaunay triangles generated by our method. Note that ground laser points are not drawn for better visualization.

tracking approach (Section III-C) is used to estimate the ego-motion between adjacent frames. Once a new keyframe is generated, the feature's depth prior is obtained using the proposed LSDT depth fitting algorithm (Section III-B), then the pose is refined via a visual LiDAR joint optimization method (Section III-D). After that, a bag-of-word (BoW)-based loop closure detection is performed to detect loop closure. Finally, a pose graph optimization process is performed to maintain global consistency of the global map.

B. LSDT Depth Fitting

In this section, we describe the depth fitting method to extract the point depth from the LiDAR data based on object segmentation and Delaunay triangulation, as shown in Fig. 3.

The extraction of oriented FAST and rotated BRIEF (ORB) features is initially performed, following the methodology delineated in ORB-SLAM2 [25]. Subsequently, a ground extraction method [26] segments the original LiDAR point cloud into ground and nonground components. The nonground point cloud is further processed through a range-image-based segmentation approach [27], resulting in the formation of distinct object point clusters. Postelimination of smaller clusters, points within the same cluster are assigned a unique object label. The final stage involves the construction of Delaunay triangulation for the points corresponding to each object label as well as for the ground point cloud, as illustrated in Fig. 3(d). The triangulation results are integral in fitting the depth to the feature points, wherein the depth is determined at the intersection of the point ray and the respective triangle plane. Algorithm 1 summarizes the entire depth fitting algorithm.

In contrast to the above methods [2], [12], we assume that the depth on the same object is continuous. We build a Delaunay triangulation of each object based on this assumption to fit the depth value of feature points located on the object. Since no foreground and background segmentation or plane fitting is required, our proposed feature depth fitting is very efficient. The typical running time is 10 ms. In addition, range-image-based point cloud segmentation takes into account the continuity of objects in space, making it more robust than local foreground and background segmentation. Thus, even when using sparse laser sensors like the VLP-16, our depth fitting

Algorithm 1 LSDT Feature Depth Extraction Algorithm

Require: LiDAR point cloud $\mathbf{P}$; ORB feature points set $\mathbf{F} = \{p_1, p_2, \ldots, p_n\}$; Camera intrinsic parameters $\mathbf{K}$; The external parameters $\mathbf{T}_{LiDAR}^{Cam}$ between camera and LiDAR.
Ensure: The depth $\{d_1, d_2, \ldots, d_n\}$ of ORB features.
// (1) LiDAR segmentation
$\mathbf{P}$ is segmented into ground point cloud and nonground point cloud using [28].
The nonground point cloud is segmented into clusters $\{\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_m\}$ using [27].
// (2) Generate objects based 2.5D Delaunay triangles (2.5D refers to pixel coordinates (u, v) and depth value d).
for i = 1 to m **do**
For each LiDAR point $P_k \in \mathbf{S}_i$, project P_k into image plane using (1) and (2);
Construct Delaunay triangle vertexes $\mathbf{v}_k = \{u_k, v_k, d_k, label = i\}$, and insert into vertex set $\mathscr{V}$;
end for
Generate Delaunay triangles **DT** using vertexes $\mathscr{V}$
// (3) Extract the depth via the generated **DT**.
for each ORB feature point $p_i \in \mathbf{P}^V$ **do**
if locate(DT, p_i**) then**
Interpolate the depth d_i of feature p_i using the triangle via (5).
end if
end for

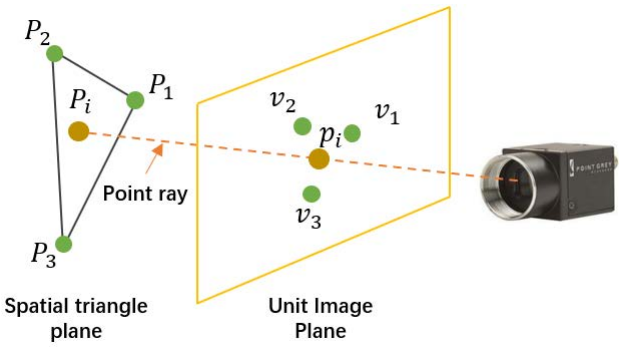

Fig. 4. Illustration of feature depth fitting process.

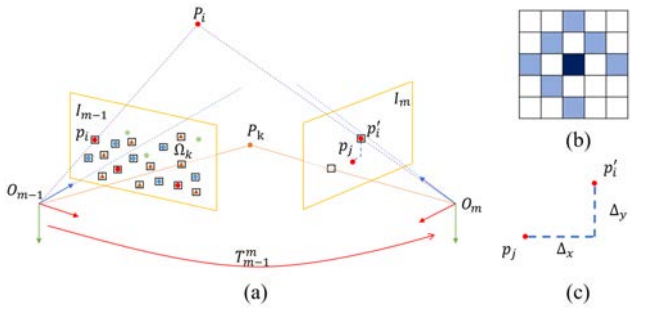

Fig. 5. (a) Illustration of the hybrid frame-to-frame tracking process, red points, blue points, and green points represent visual feature points, and brown triangle points are salient laser points. (b) Patch pattern that is used to calculate photometric residual. (c) Geometric reprojection residual.

method can provide more accurate depth values, especially at the edges of objects.

The depth d_i of the corresponding feature point $p_i = [\,u_i\ v_i\,]^T$ can be calculated via the following equation:

$$d_i = \frac{n_1X_1 + n_2Y_1 + n_3Z_1}{n_1(u_i - c_x)/f_x + n_2(v_i - c_y)/f_y + n_3} \tag{5}$$

where $\mathbf{n} = [\,n_1\ n_2\ n_3\,]^T$ is the normal vector of the space plane and calculated using the three vertices of the spatial triangle that the feature point ray intersects. $P_1 = [\,X_1\ Y_1\ Z_1\,]^T$ is the vertex of the spatial triangle as shown in Fig. 4.

C. Hybrid Visual LiDAR Frame-to-Frame Tracking

We created a hybrid visual LiDAR frame-to-frame tracking module to ensure accurate and fast ego-motion estimation. It combines important LiDAR points with visual feature points to reduce photometric and reprojection errors. It hybridizes salient LiDAR points and visual feature points to optimize both photometric error and reprojection error. Visual features are affected by textures, and LiDAR points are affected by degraded scenes. The combination of these two types of sensor points can mitigate their respective drawbacks and achieve a robust tracking effect. Meanwhile, our method benefits from the advantages of both direct and feature-based methods due to the combination of photometric error and geometric reprojection error. The specifics of two types of cost residuals, photometric residual and geometric residual are detailed in the following.

1) Photometric Residual: We minimize the photometric errors only for those pixels with depth which are composed of two types: the selected salient LiDAR points, and the ORB feature points. Note that in this step, LiDAR points outside the camera's field of view are discarded. However, during the subsequent visual-LiDAR joint optimization, all laser points from the current frame are employed to optimize the pose. Every point in a patch is assigned the same depth of the feature in the center of this patch. Examples of two types of pixels are shown by the brown triangle and red/blue circle, respectively, in Fig. 5(a). For a point p_k with depth d_k in frame $\mathcal{F}_{m-1}$, the photometric error is defined as follows:

$$r(p_k) = \mathcal{I}_{m-1}(p_k) - a\mathcal{I}_m\left(\pi\left(\mathbf{T}_{m-1}^m\pi^{-1}(p_k, d_k)\right)\right) + b \tag{6}$$

where a and b are the gain and brightness of the current image. Finally, our photometric objective function for tracking is defined as follows:

$$\mathbb{E}_{\text{pho}} \overset{\text{def}}{=} \sum_{i=1}^{N}\sum_{p_j\in\Omega(p_i)} w\big(r(p_j)\big)\big(r(p_j)\big)^2 \tag{7}$$

where $\Omega(\cdot)$ is the patch of the sparse pattern shown in Fig. 5(b), and the t-distribution-based weight function $w(\cdot)$ is defined as follows:

$$w(r) \overset{\text{def}}{=} \frac{\nu + 1}{\nu + \left(\frac{r}{\delta r}\right)^2}. \tag{8}$$

Here, the degree of freedom ν is set to 5, and the variance of residual δr is obtained in each iteration while performing the Gauss–Newton optimization.

2) Geometric Residual: For each incoming frame $\mathcal{F}_m$, the image features are first associated with the last frame $\mathcal{F}_{m-1}$. Denote p_i and p_j one of the matched corresponding features within the last frame and the current frame, respectively. Thus, the reprojection error can be defined as follows:

$$r(p_i) = p_j - \pi\left(\mathbf{T}_{m-1}^m\pi^{-1}(p_i, d_i)\right) \tag{9}$$

where d_i is the depth of p_i, and will be optimized in bundle adjustment module. Finally, our feature-based reprojection geometric error objective function for tracking is defined as follows:

$$\mathbb{E}_{\text{geo}} \overset{\text{def}}{=} \sum_{i=1}^{N} w(r(p_i))(r(p_i))^2. \tag{10}$$

The weight function $w(\cdot)$ we use is Huber kernel which is used in ORB_SLAM2 [29]

$$w(r) \overset{\text{def}}{=} \begin{cases} 1, & \text{if } r < d \\ \dfrac{d}{r}, & \text{else.} \end{cases} \tag{11}$$

Combing (7) and (10), we obtain a minimization problem of the form

$$\{\xi^*, a^*, b^*\} \overset{\text{def}}{=} \arg\min_{\xi} \frac{1}{\delta_{\text{pho}}^2}\sum_{i=1}^{N}\mathbb{E}_{\text{pho},i} + \frac{1}{\delta_{\text{geo}}^2}\sum_{i=1}^{M}\mathbb{E}_{\text{geo},i} \tag{12}$$

where δ_{pho}^2 and δ_{geo}^2 are the standard deviations of the residuals of (6) and (9), respectively.

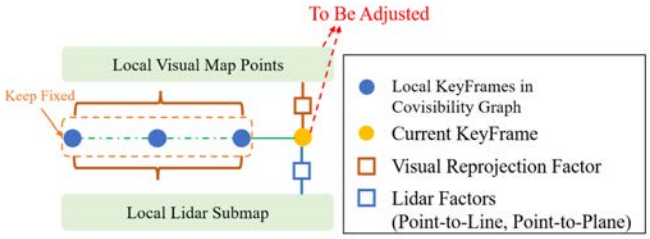

Fig. 6. Graph structure for visual-LiDAR joint optimization.

After solving the nonlinear optimization problem 12, the camera pose $\mathbf{T}_m$ of the current frame is updated immediately. Then, the 3-D map points are reprojected to the current frame to check whether the 3-D–2-D matching is an outlier or not. Outliers will be removed once detected.

D. Visual-LiDAR Jointly Local Mapping

Since scan-to-map-based pose refinement is an important method to reduce cumulative error and provide a more accurate estimated pose, it has been widely used in state-of-art LiDAR systems [30], [31], [32], [33]. However, in degeneration environments with less visible structures such as tunnels or hallways, these methods are still prone to failure. In comparison to the previously mentioned LiDAR-based methods [2], [3], [12], depth enhanced visual feature-based approaches show advantages, particularly in rotation estimation when relatively high-definition range data are available. The reasons can be divided into the following categories. To begin, the depth of a visual feature is generated directly from the precise LiDAR scan rather than being triangulated using previously estimated motion, reducing the uncertainty of depth estimation. The positions of the visual map points are then refined using bundle adjustment techniques. Finally, because of the rich texture information provided by the image sensor, feature extraction can reach subpixel levels, and feature matching is more accurate and robust than LiDAR feature association.

Therefore, our key idea is to fuse the visual reprojection factors and LiDAR scan-to-map factors to refine the pose of the current keyframe and local visual map points. For each incoming keyframe $\mathcal{KF}_k$, the image features $\mathcal{F}_k$ of the current keyframe are matched with the local visual map. Let $\mathcal{X}_k$ denote all matched features at time k. In addition, the edge points $\mathcal{E}_k$, planar points $\mathcal{P}_k$, and ground points $\mathcal{G}_k$ are extracted according to the computed local curvature at the same time, and the details of feature extraction can be found in [34]. Note that, the LiDAR feature points $\{\mathcal{E}_k, \mathcal{P}_k, \mathcal{G}_k\}$ have been undistorted using the relative transform T^m_{m-1} obtained in the last section. We linearly interpolate points within one scan and correct distortion. The details about distortion can be found in [33].

An illustration of the joint visual LiDAR optimization is given in Fig. 6. Finally, the pose of the current keyframe can be estimated by solving the following nonlinear least-square problem:

$$\min_{R,t} \frac{1}{\delta_v^2} \sum_{X_k^i \in \mathcal{F}_k} \rho \underbrace{\left(\| d_{\mathcal{R}}\left(X_k^i\right)\|^2\right)}_{\text{visual constraints}} + \frac{1}{\delta_e^2} \sum_{E_k^j \in \mathcal{E}_k} \rho \underbrace{\left(\| w_j \cdot d_{\mathcal{E}}\left(E_k^j\right)\|^2\right)}_{\text{lidar edge constraints}}$$

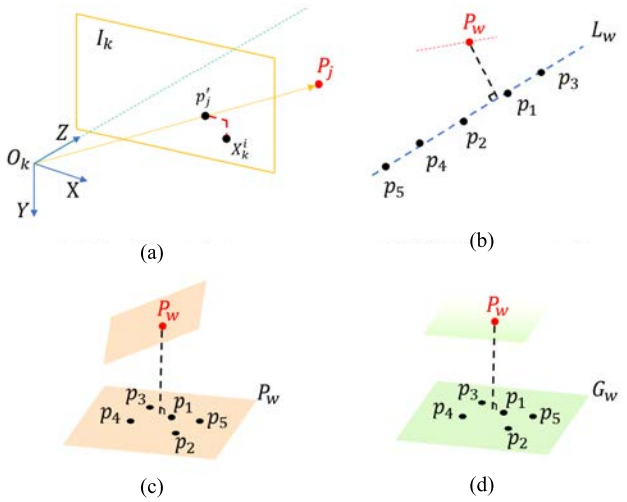

Fig. 7. Illustration of visual-LiDAR joint optimization error terms. (a) Visual constraints. (b) LiDAR edge constraints. (c) LiDAR plan constraints. (d) LiDAR ground constraints.

$$+ \frac{1}{\delta_p^2} \sum_{P_k^m \in \mathcal{P}_k} \rho \underbrace{\left(\| w_m \cdot d_{\mathcal{P}}\left(P_k^m\right)\|^2\right)}_{\text{lidar planar constraints}}$$

$$+ \frac{1}{\delta_g^2} \sum_{G_k^n \in \mathcal{G}_k} \rho \underbrace{\left(\| w_n \cdot d_{\mathcal{P}}\left(G_k^n\right)\|^2\right)}_{\text{lidar ground constraints}}$$

where $\rho(\cdot)$ is the Huber norm, $d_{\mathcal{R}}(\cdot)$ represents the visual reprojection error, $d_{\mathcal{E}}(\cdot)$ and $d_{\mathcal{P}}(\cdot)$ are point-to-line error and point-to-plane error, respectively.

1) Visual Constraint: For each feature point $X_k^i \in \mathcal{F}_k$ of current KeyFrame, the visual constraint is defined as (13), illustrated in Fig. 7(a)

$$d_{\mathcal{R}}\left(X_k^i\right) = X_k^i - \pi\left(\mathbf{T}_W^k \times P_j\right) \tag{13}$$

where $P_j \in \mathbb{R}^3$ is a 3-D feature map point belonging to local visual map, and $\mathbf{T}_W^k$ is the initial pose obtained from Section III-C.

2) Lidar Edge Constraints: For each lidar edge point $E_k^j \in \mathcal{E}_k$, we find five nearest points from $\mathbb{E}_m$ as illustrated in Fig. 7(b). Noticing that the point E_k^j is in the local LiDAR frame while local LiDAR map points of $\mathbb{E}_m$ are registered in the global coordinates, to find the nearest points of E_k^j in $\mathbb{E}_m$, E_k^j is projected into global coordinates using $p_w = \mathbf{T}_W^k \times E_k^j$. Then, the residual of LiDAR edge constraint can be defined as follows:

$$d_{\mathcal{E}}\left(E_k^j\right) = \frac{|(p_w - p5) \times (p_w - p_1)|}{|p_5 - p_1|}. \tag{14}$$

3) Lidar Planar Constraints: For each planar LiDAR point $P_k^m \in \mathcal{P}_k$ of current keyFrame, we also find five nearest points in the local planar LiDAR map $\mathbb{P}_m$ as illustrated in Fig. 7(c). The LiDAR planar constraint is defined as the following equation:

$$d_{\mathcal{P}}\left(P_k^m\right) = \frac{(p_w - p_1)^T((p_3 - p_5) \times (p_3 - p_1))}{|(p_3 - p_5) \times (p_3 - p_1)|} \tag{15}$$

where $p_w = \mathbf{T}_W^k \times P_k^m$

TABLE I

RELATIVE TRANSLATION (%)/ROTATION (DEG/100M) ERROR COMPARISON OF THE PROPOSED HVL-SLAM WITH EXISTING METHODS ON SEQUENCES 00–10 OF THE KITTI DATASET

Sequence (Total length [m])	00 (3724)	01 (2453)	02 (5067)	03 (560)	04 (393)	05 (2205)	06 (1232)	07 (694)	08 (3222)	09 (1705)	10 (919)	avg
Ours(*)	0.64/0.22	1.74/0.39	0.83/0.21	0.81/0.39	0.34/0.32	0.36/0.16	0.34/0.16	0.45/0.24	1.07/0.34	0.72/0.24	0.70/0.28	0.72/0.27
GAC-Mapping(*)	1.68/0.78	15.22/0.87	1.98/0.77	1.94/1.10	2.74/1.10	1.13/0.52	1.19/0.58	0.94/0.67	1.69/0.87	1.56/0.80	3.14/1.37	3.02/0.85
DV-LOAM(*)	0.66/0.30	1.74/0.43	0.84/0.36	0.85/0.48	0.39/0.62	0.54/0.30	0.70/0.33	0.42/0.22	0.93/0.27	0.75/0.27	0.87/0.47	0.78/0.36
CamVox(*)	0.77/0.35	×	0.89/0.34	0.84/0.31	0.70/0.71	0.59/0.34	0.89/0.48	0.44/0.31	2.24/0.83	0.79/0.37	0.76/0.37	×
RGB-L(*)	0.66/0.28	3.81/1.17	0.84/0.31	8.15/0.33	1.72/0.15	0.44/0.21	0.79/0.44	0.63/0.29	1.20/0.37	0.79/0.27	0.71/0.34	1.79/0.38
Ours(-)	0.65/0.27	1.74/0.39	0.79/0.23	0.81/0.39	0.34/0.32	0.46/0.20	0.54/0.22	0.53/0.29	1.02/0.29	0.64/0.27	0.70/0.28	0.75/ 0.29
GAC-Mapping(-)	2.05/0.88	15.22/0.87	2.08/0.79	1.94/1.10	2.74/1.04	1.47/0.73	1.06/0.54	0.88/0.78	1.84/0.94	1.56/0.80	3.14/1.37	3.09/0.89
PLVL-SLAM[20](-)	0.99/#	1.87/#	1.38/#	0.65/#	0.42/#	0.72/#	0.61/#	0.56/#	1.27/#	1.06/#	0.83/#	0.94/0.36
CamVox(-)	1.46/0.61	-	1.62/0.53	0.84/0.31	0.70/0.71	0.62/0.37	1.04/0.60	0.61/0.38	2.24/0.83	0.70/0.33	0.76/0.37	1.06/0.50
RGB-L(-)	0.89/0.38	3.83/1.24	0.89/0.32	1.00/0.33	1.72/0.15	0.55/0.25	0.71/0.29	0.85/0.40	1.19/0.38	0.69/0.29	0.76/0.33	1.19/0.40
LIMO[2](-)	1.12/#	0.91/#	#	#	0.53/#	#	#	#	#	#	#	0.93/0.26
DEMO[1](-)	1.05/#	1.87/#	0.93/#	0.99/#	1.23/#	1.04/#	0.96/#	1.16/#	1.24/#	1.17/#	1.14/#	1.14/0.49
DVL-SLAM[5](-)	0.93/#	1.47/#	1.11/#	0.92/#	0.67/#	0.82/#	0.92/#	1.26/#	1.32/#	0.66/#	0.70/#	0.98/0.40
DV-LOAM(-)	0.65/0.30	1.74/0.43	0.97/0.33	0.85/0.48	0.39/0.42	0.54/0.30	0.65/0.33	0.51/0.33	0.89/0.32	0.73/0.32	0.87/0.47	0.80/0.38
SDV-LOAM(-)	0.66/0.30	11.4/0.27	0.71/0.23	0.87/0.20	0.60/0.52	0.66/0.30	0.50/0.27	0.84/0.53	1.09/0.37	0.63/0.34	0.68/0.41	1.69/0.34
A-LOAM(-)	0.87/0.36	2.64/0.55	4.88/1.55	1.20/0.63	1.23/0.40	0.66/0.30	0.62/0.28	0.58/0.43	1.18/0.43	1.10/0.45	1.46/0.53	1.49/0.53
F-LOAM(-)	0.78/0.37	2.64/0.56	1.43/0.45	1.21/0.62	1.20/0.39	0.65/0.31	0.65/0.37	0.54/0.41	1.11/0.43	1.05/0.44	1.46/0.49	1.16/0.44
T-LOAM(-)	2.55/1.49	3.43/0.74	1.99/1.47	1.80/1.47	1.67/1.43	1.24/0.98	0.89/0.49	1.69/1.71	2.41/1.31	1.37/1.16	2.26/1.05	1.94/1.21

Best translation/rotation error are highlighted in red, and the second best are highlighted with blue, respectively. (*) means with loop closure while (-) represents no loop closure. # represents this result is not provided in the original paper. × represents running failed.

4) Lidar Ground Constraints: Similar to planar constraints, the LiDAR ground constraint is illustrated as Fig. 7(d), and is defined as follows:

$$d_{\mathscr{P}}\left(G_k^n\right) = \frac{(p_w - p_1)^T((p_3 - p_5) \times (p_3 - p_1))}{|(p_3 - p_5) \times (p_3 - p_1)|} \tag{16}$$

where $p_w = \mathbf{T}_W^k \times G_k^n$

Moreover, unlike LOAM [33] which directly uses the distance function as residuals, the line/plane fitting quality w is also taken into account to weight the residual block, which can be evaluated as the following equation:

$$w = \sqrt{\left(\lambda_3^2 - \lambda_f^2\right)/\lambda_3^2} \tag{17}$$

where λ are eigenvalues in ascending order ($\lambda_3 \geq \lambda_2 \geq \lambda_1 \geq 0$), and can be obtained based on the local geometry distribution of LiDAR features. λ_f is λ_2 in the line fitting case, and becomes λ_1 in the plane fitting case.

In addition, to let the visual point residual have the same metric with the LiDAR edge, planar and ground constraints, we use the inverse covariance $\{(1/\delta_v^2), (1/\delta_e^2), (1/\delta_p^2), (1/\delta_g^2)\}$ to adjust their weights. To keep efficiency while obtaining accurate ego-motion estimation, only those map points observed by more than n keyframes are involved to (Section III-D). In this letter, n is set to 3. The max number of edge, planar, and ground feature factors are set to 1000, 1000, and 200, respectively. The nonlinear problem (Section III-D) is solved by Ceres Solver [35], and the covariance is updated only in the first iteration.

E. Loop Closure Detection and Pose-Graph Optimization

Our system is developed based on ORB-SLAM2, with the keyframe generation strategy remaining consistent with that of ORB-SLAM2. The only modification is the introduction of a condition that permits the addition of a keyframe at most every five frames to prevent the local point cloud map from becoming overly sparse. Loop closure detection is performed whenever a new keyframe is generated. In our framework, the loop closure module depends on the DBoW proposed by [17] and pretrained visual vocabulary from ORB-SLAM2. Whenever a new keyframe is generated, the frame is represented as BoWs vector, which will be used to detect loop candidates as done in ORB-SLAM2. After detecting the loop between two keyframes, the relative pose between the two base frames is calculated by calculating the relative transformation between two submap point clouds using the voxelized generalized-iterative-closest point (V-GICP) [36] algorithm. The relative pose is then added as a constraint edge to the pose graph.

IV. EXPERIMENTS

A. Validation on the KITTI Odometry Dataset

The precision of HVL-SLAM is primarily evaluated on the KITTI training odometry benchmark [37]. The comparison includes the following state-of-the-art LiDAR-based odometry frameworks, such as A-LOAM [30], F-LOAM [31], T-LOAM [38], and visual LiDAR fusion methods, such as DEMO [1], LIMO [2], CamVox [3], RGB-L [4], GAC-Mapping [10], DVL-SLAM [5], DV-LOAM [21], SDV-LOAM [22], and PLVL-SLAM [12]. We directly take the results provided in the papers of PLVL-SLAM, LIMO, DEMO, and DVL-SLAM for comparison. The results of other methods are obtained through running their open source code. The evaluation results of all sequences are presented in Table I, and Fig. 8 shows sample trajectories of the proposed method.

In the whole KITTI training dataset, we achieved an average translation error of 0.75% and a rotation error of 0.0029 deg/m, which outperforms those state-of-the-art visual LiDAR fusion odometry approaches. While LIMO produces better average rotation estimates than ours, it frequently requires additional semantic label information to identify moving objects and adjust the weights of various types of landmarks. Obtaining such semantic information typically requires the use of GPU devices and takes time, making it unsuitable for deployment on mobile robots with limited computational resources and power. Besides, compared with those LiDAR-based methods, our approach also shows better performance than F-LOAM

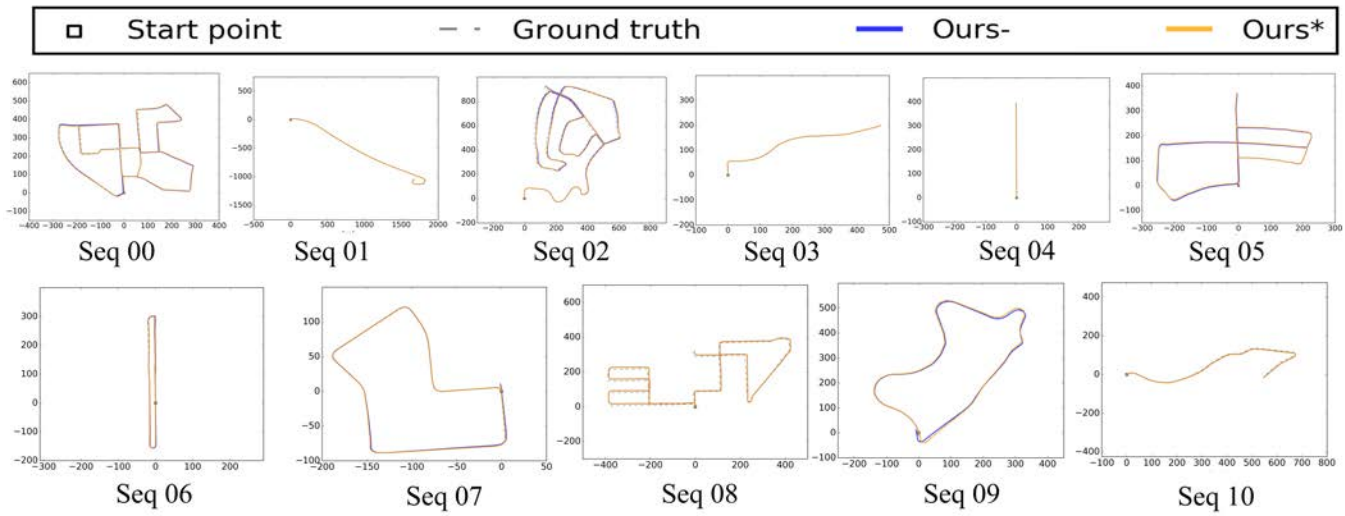

Fig. 8. Trajectory diagram concerning the training set.

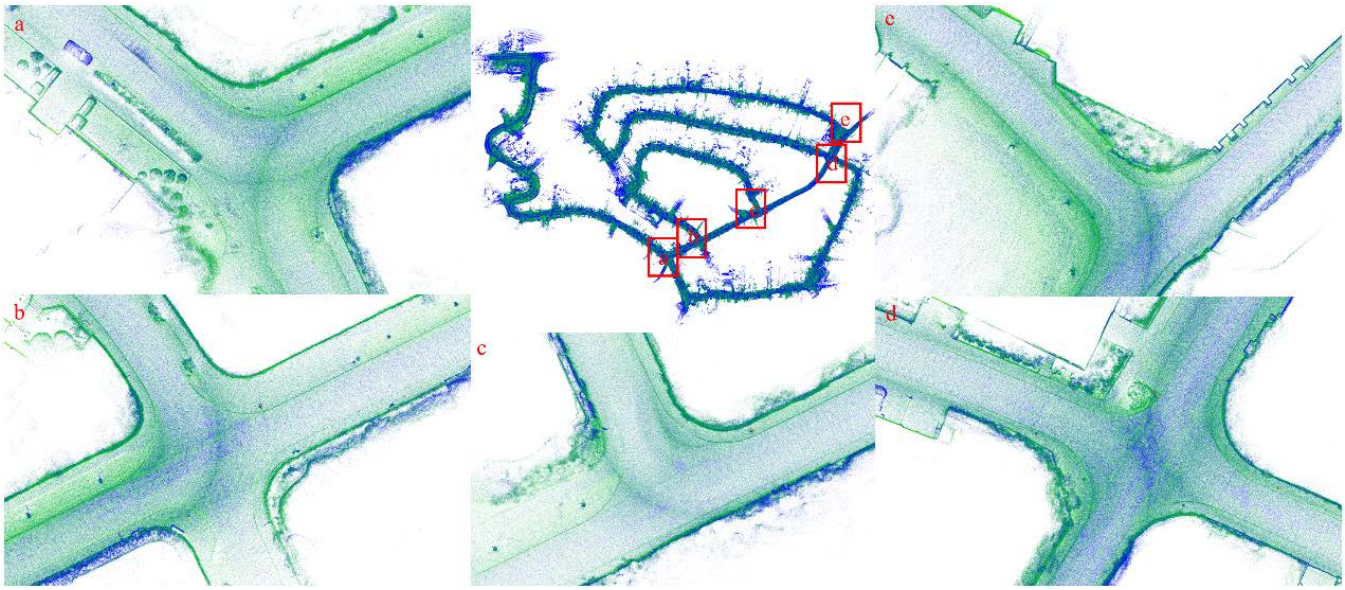

Fig. 9. LiDAR map generated by HVL-SLAM from the KITTI 02 sequence.

and T-LOAM. Figs. 9 and 10 show the mapping results of the LiDAR point clouds for the KITTI 02 and 08 sequences using the method proposed in this paper. Fig. 11 illustrates the visual-LiDAR fusion map.

Furthermore, we also perform the error analysis according to the path length and vehicle speed, and the comparison curves are shown in Fig. 12. When the car moves faster, our system leads to a decrease in accuracy partially because of the poor feature association. This explains why our method performs relatively poorly on sequence 01, a very challenging case with very fast moving speed. The position and orientation changes of the proposed method on the KITTI odometry sequence 00 are shown in Fig. 13. Our approach has significantly reduced the localization error of translation and orientation.

B. Validation on Our Dataset

To verify our algorithm's accuracy in the real world, the proposed HVL-SLAM system is further tested in the Wuhan University parking-lot, library, and Jiangxia scenario with numerous trees, vehicles, roads, buildings, and sidewalks, which can be seen in Fig. 14. The parking-lot test scene has two loop closures, while the library dataset does not include any loop closures. In contrast, the Jiangxia dataset, depicted in Fig. 1, contains three loop closures.

1) Experiment I: The primary experiment is designed to demonstrate that HVL-SLAM can accomplish low-drift ego-motion and the reestablished map with global consistency. The comparisons of the trajectory from HVL-SLAM, with two mainstream SLAM frameworks and the ground truth are evaluated on our Whu-parking-lot dataset shown in Fig. 15(a), and the result is evaluated using evo tools [39], shown in Table II. To verify the accuracy of poses estimated by our system, we compared it with DV-LOAM [21], which is a direct visual-LiDAR SLAM approach, and LeGO-LOAM [34], which improves feature extraction with segmentation and clustering for efficiency improvement while adding loop detection

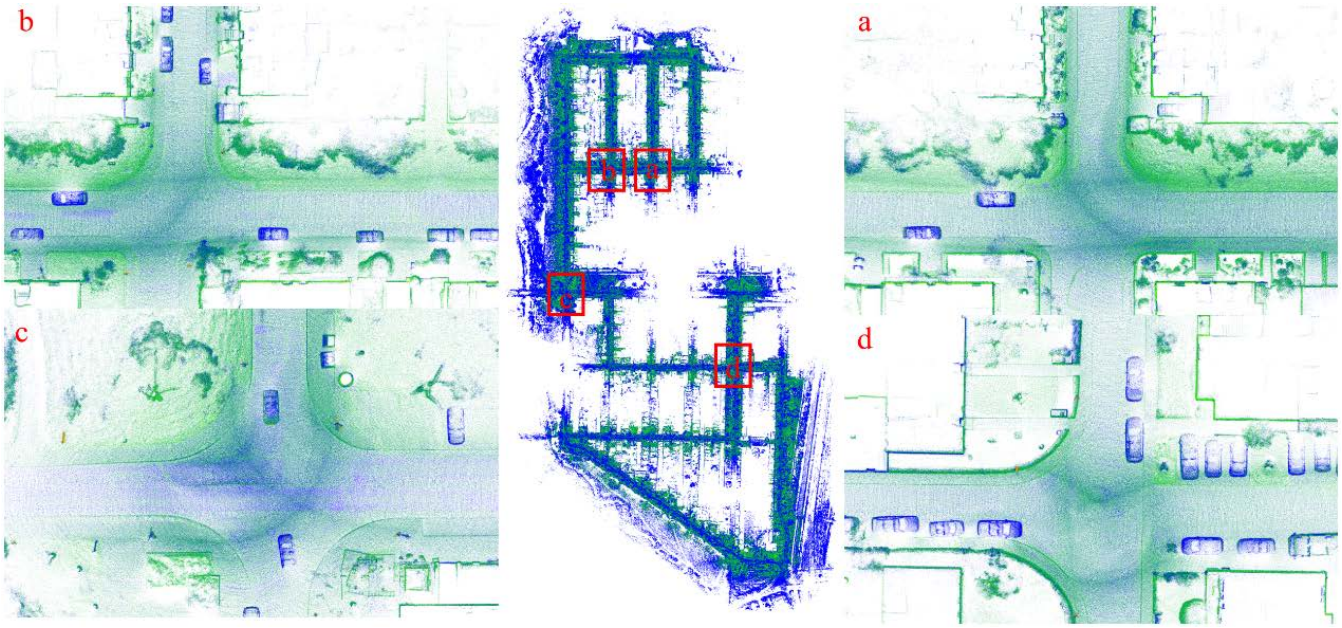

Fig. 10. LiDAR map generated by HVL-SLAM from the KITTI 08 sequence.

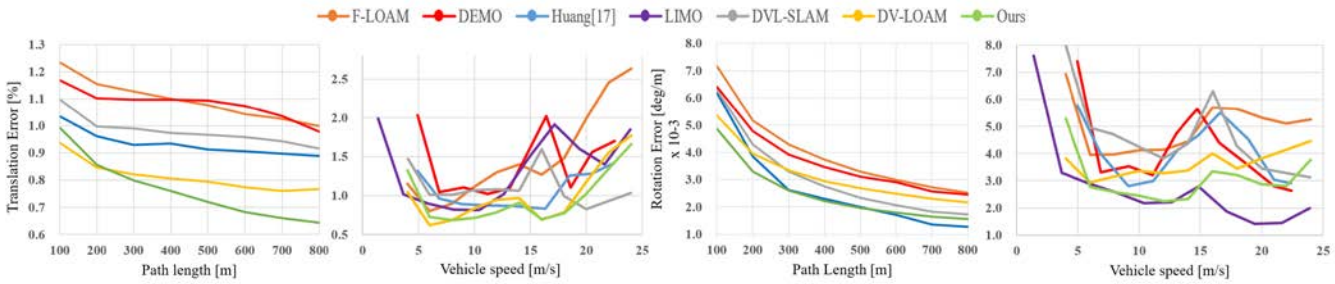

Fig. 11. Rotation and translation error analysis. Since LIMO only provided error curves according to vehicle speed, so we only draw DEMO, DVL-SLAM, DV-LOAM, PLVL-SLAM [12] and ours error curves in the path length analysis.

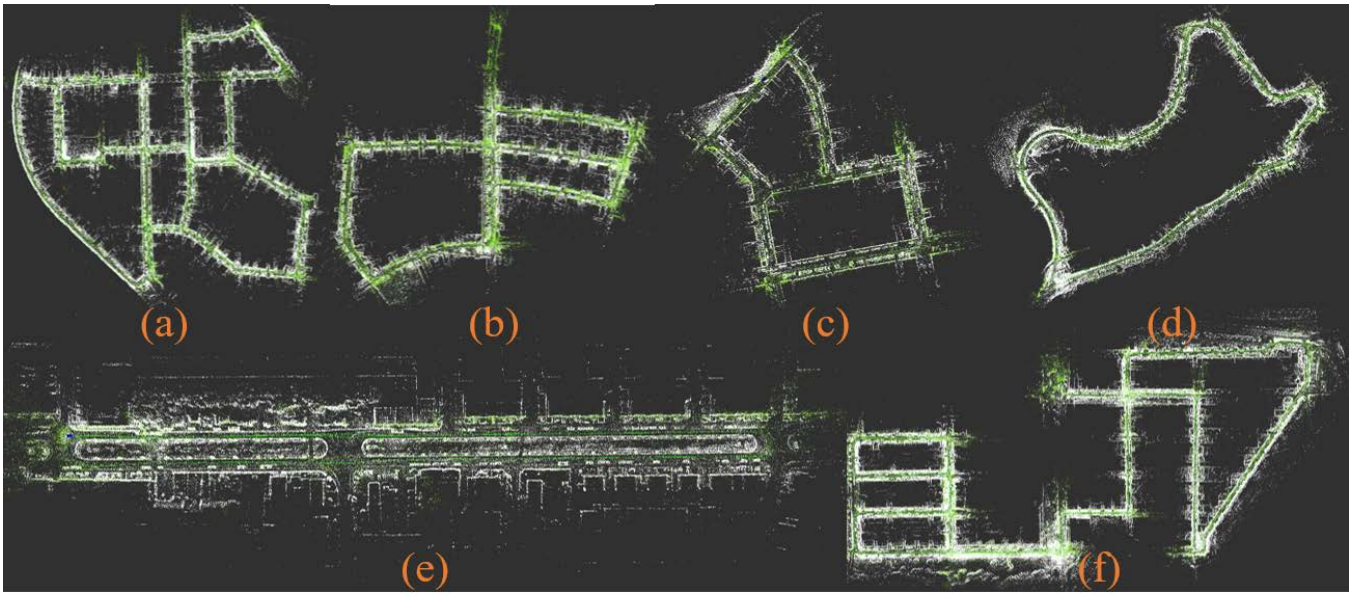

Fig. 12. Visual-LiDAR fusion map generated by HVL-SLAM. (a)–(f) Correspond to KITTI sequence 00, 05, 07, 09, 06, 08, respectively.

for long-run drift reduction. As we can see from Table II, ours method (1.73 m) shows better root mean-square error (RMSE) than LeGO-LOAM (2.48 m) and DV-LOAM (2.24 m).

2) Experiment II: The following experiment is designed to validate the proposed scheme's stability in more complex, deteriorating scenarios. The library scene is a highly

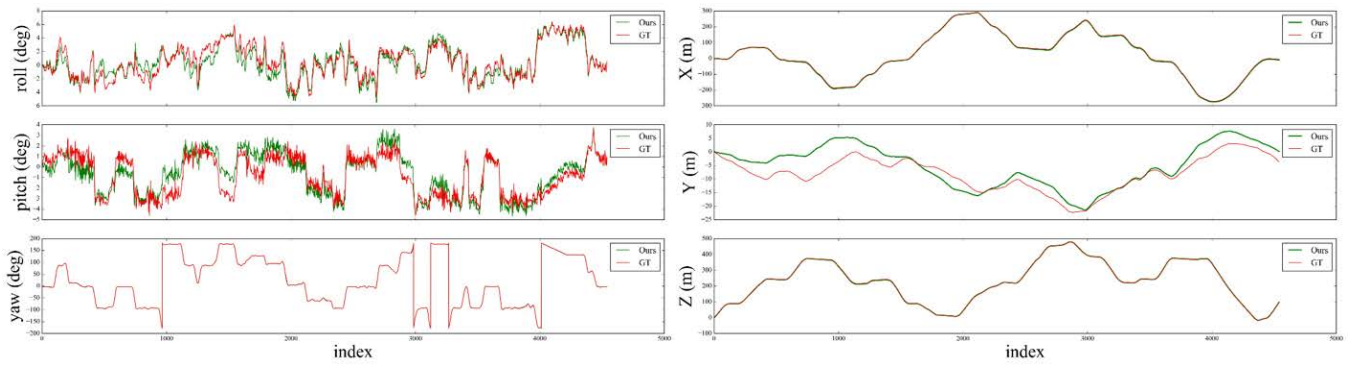

Fig. 13. Change of position and orientation contrasted with ground truth on the KITTI sequence 00 odometry benchmark. Note that the coordinate of the result has been converted from $xyz \rightarrow zxy$ through calibration.

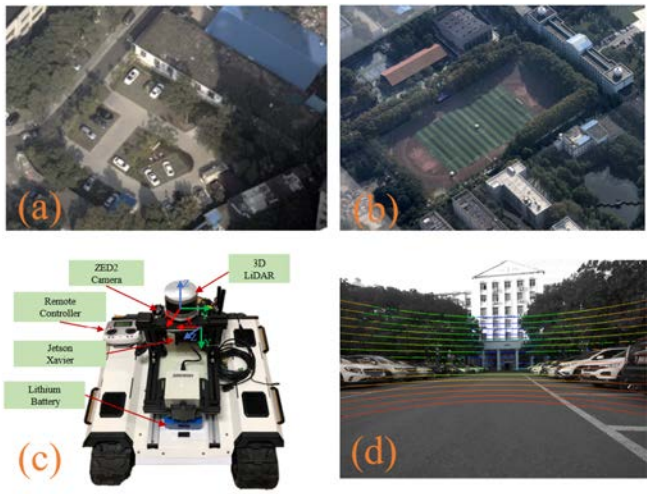

Fig. 14. Experimental scenes and robot platform. (a) and (b) Are parking lot and library experiment scenes, respectively. (c) Robot experimental platform mainly equipped with VLP-16 and ZED2. (d) Sample data, the LiDAR points are projected to the image plane, and color-coded by their depth.

TABLE II
ABSOLUTE POSE ERROR (APE) (UNIT:M)

dataset (Length [m])	parking-lot (200)				library (700)			
Method	Ours	DV-LOAM	LeGO-LOAM	MULLS	Ours	F-LOAM	LeGO-LOAM	MULLS
Max	2.53	3.38	3.77	3.67	5.78	7.91	6.72	38.85
Mean	1.67	2.10	2.32	2.28	2.37	4.31	3.49	13.70
Min	0.16	0.25	0.19	0.55	0.48	1.23	0.74	5.70
RMSE	1.73	2.24	2.48	2.41	2.61	4.49	3.69	15.56
Std	0.60	0.76	0.87	0.78	1.08	1.25	1.21	7.37

Best results are highlighted in red with second best in blue.

dynamic dataset with many moving vehicles and pedestrians, thereby intensifying the challenges of visual odometry. DV-LOAM [21] and SDV-LOAM [22] fails in this scene due to the influence of dynamic obstacles and the lack of relocalization capabilities provided by visual feature maps. In addition, since a large number of laser points are located on the ground, and the surrounding trees are sparsely distributed, F-LOAM that directly uses edge features and planar features for scan-to-map pose estimation has a large error, especially in elevation, as shown in Fig. 15(a). By introducing a two-step optimization for pose estimation, LeGO-LOAM shows better performance in ego-motion estimation.

TABLE III
RELATIVE TRANSLATION (%)/ROTATION (DEG/100 M) ERROR ON THE WHOLE KITTI TRAINING DATASET WITH DIFFERENT FEATURE DEPTH EXTRACTION METHODS

seq.	Env.	RGB-L	CamVox	GAC-Mapping	LIMO	Ours
00	Urban	0.89/0.38	1.46/0.61	2.38/0.33	0.88/0.39	0.75/0.32
01	Highway	3.83/1.24	5.21/1.61	×	×	1.86/0.70
02	Suburb	0.89/0.32	2.75.0.34	0.89/0.36	0.89/0.36	0.81/0.32
03	Country	1.00/0.33	6.82/0.36	2.59/1.40	2.45/1.51	0.87/0.28
04	Country	1.72/0.15	1.14/0.44	7.50/0.46	0.78/0.58	1.09/0.13
05	Urban	0.55/0.25	1.10/0.27	2.39/0.29	0.75/0.37	0.57/0.26
06	Urban	0.71/0.29	1.04/0.60	2.95/0.30	0.78/0.36	0.70/0.23
07	Urban	0.85/0.40	1.45/0.44	2.59/0.48	0.84/0.45	0.80/0.38
08	Suburb	1.19/0.38	2.24/0.83	2.76/0.43	1.25/0.44	1.08/0.37
09	Suburb	0.69/0.29	1.28/0.39	3.05/0.31	×	0.71/0.29
10	Suburb	0.76/0.33	1.54/0.30	3.05/0.34	0.92/0.41	0.81/0.37
avg		1.19/0.40	2.37/0.56	×	×	0.91/0.33

Best results are highlighted in red with second best in blue.

Our method combines the advantages of F-LOAM and LeGO-LOAM and adds visual features for joint optimization, which greatly improves the accuracy of pose estimation and effectively suppresses the accumulation of errors in elevation. Although the scene contains many dynamic vehicles and pedestrians, our method can rely on the relocalization capabilities based on the visual-LiDAR fusion feature map to maintain a good localization effect, even if it fails temporarily due to the influence of dynamic objects. Due to the efficient feature depth implementation and visual-LiDAR joint optimization, our system achieves the smallest RMSE compared to state-of-the-art laser odometry methods, and the mapping results are shown in Fig. 16.

3) Experiment III: To further prove the performance of the proposed system in a larger environment. We conducted a real-world experiment using a test vehicle. The mapping result is shown in Fig. 1, and the details of LiDAR map is shown as Fig. 18. Fig. 19 displays the mapping results of other methods on this dataset. Compared to these methods, our proposed HVL-SLAM achieves superior global consistency results with the assistance of the loop closure detection module. Fig. 20 illustrates the first two loops observed in the experiment, and more detailed results can be seen in the video attachment.

C. Ablation Study

We also perform an evaluation of each module in our system.

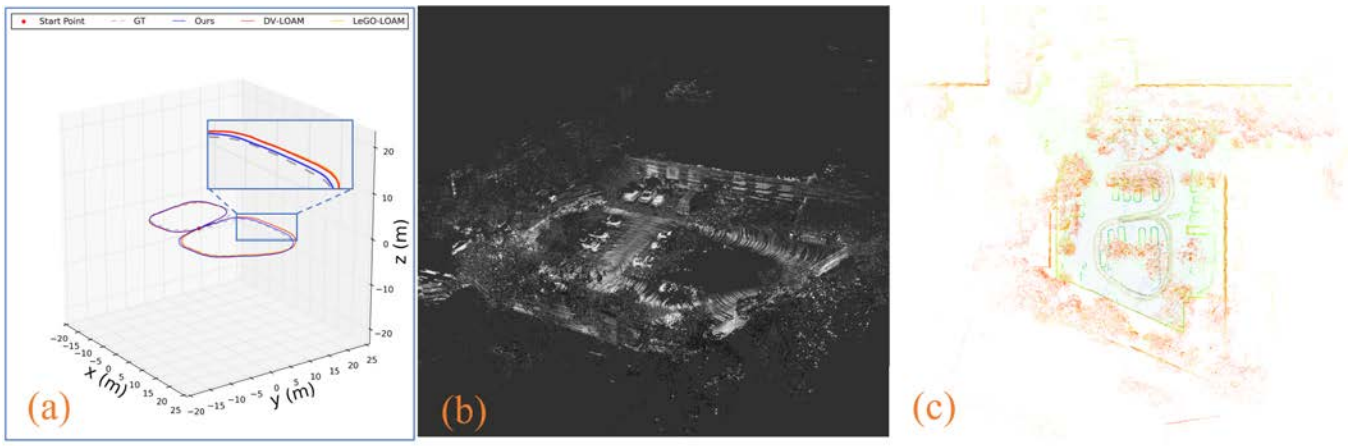

Fig. 15. Results of experiment I, where (a) 3-D trajectories and (b) point cloud map color-encoded with images. (c) 3-D LiDAR map.

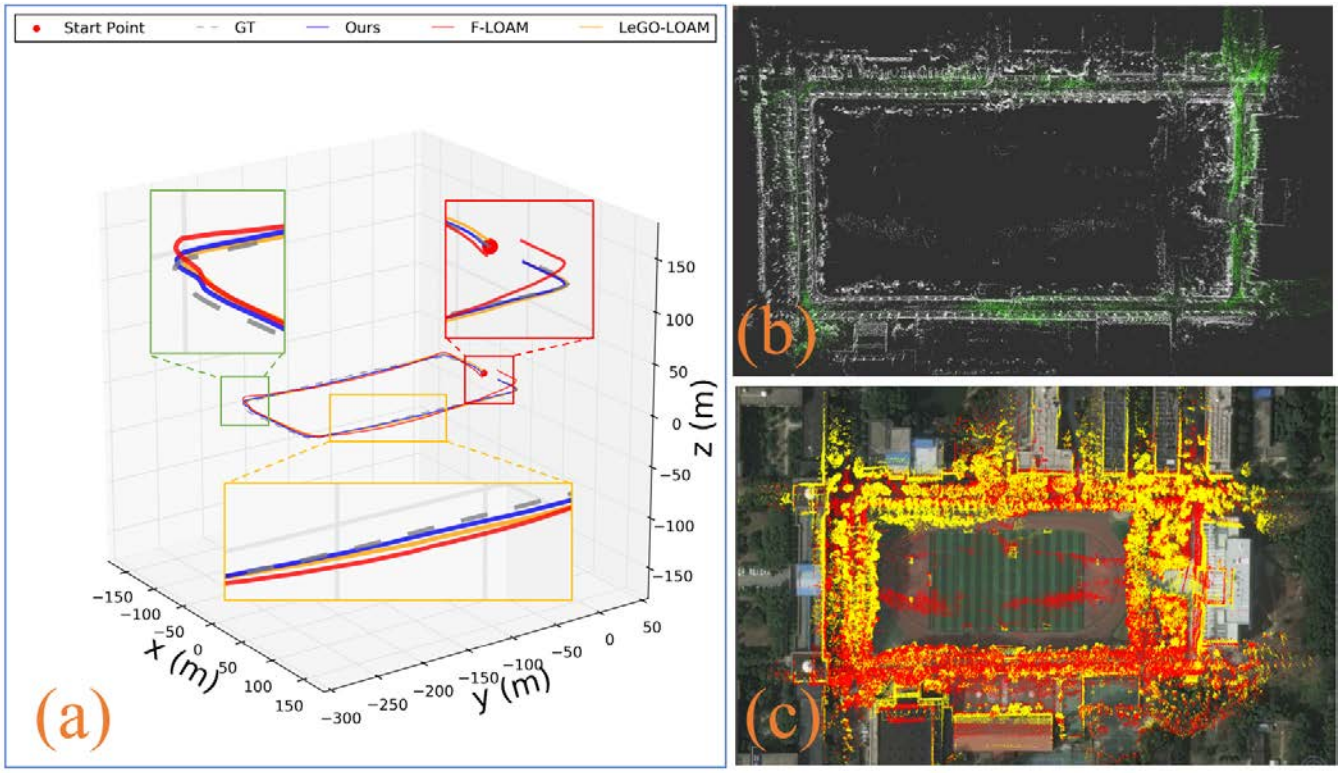

Fig. 16. Ours results on experiment II. (a) 3-D trajectories of different methods. (b) Visual LiDAR fusion map, where white represents corner laser points and green represents visual feature points. (c) LiDAR mapping result, which shows a good alignment with Google Map.

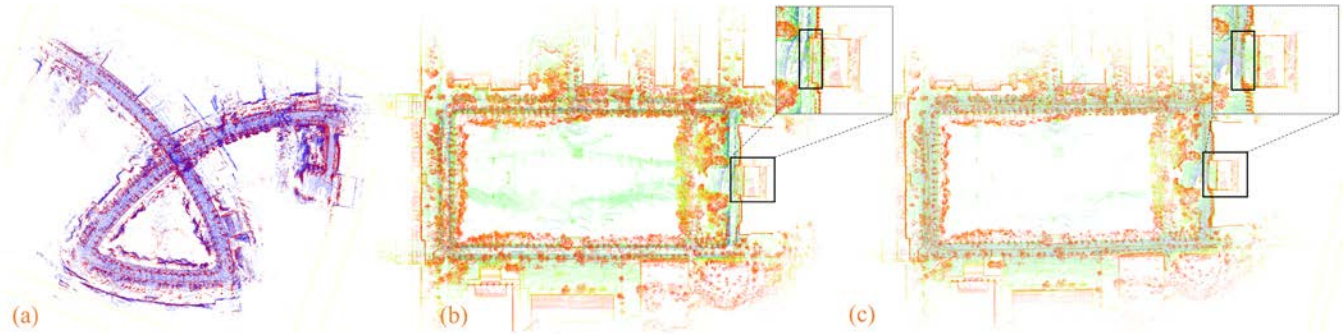

Fig. 17. Mapping results on experiment II. (a) GAC-Mapping results. (b) LeGO-LOAM mapping results. (c) Ours mapping result.

TABLE IV
FEATURE DEPTH EXTRACTION EFFICIENCY

	times (ms)					features num.					average time of each feature (10^{-3} ms)				
	RGB-L	CamVox	GAC-Mapping	Limo	Ours	RGB-L	CamVox	GAC-Mapping	Limo	Ours	RGB-L	CamVox	GAC-Mapping	Limo	Ours
00	2.07	7.27	42.85	17.98	12.61	811.57	653.18	1425.64	331.55	1205.63	0.26	1.13	3.12	5.42	1.05
01	1.97	6.48	39.55	×	11.76	777.52	627.87	1462.60	×	1185.58	0.26	1.04	2.74	×	1.00
02	2.10	7.26	43.43	18.09	13.07	788.77	635.96	1344.01	277.57	1143.50	0.27	1.15	3.25	6.52	1.15
03	2.16	7.36	43.47	18.27	13.14	830.97	668.77	1302.56	243.02	1125.79	0.26	1.11	3.35	7.52	1.17
04	2.12	7.27	43.78	18.17	13.44	734.38	593.63	1238.96	188.69	1022.90	0.26	1.24	3.55	9.63	1.32
05	2.10	7.18	42.80	18.22	13.03	837.40	674.21	1454.69	332.99	1229.31	0.25	1.08	2.96	5.47	1.07
06	2.04	7.15	42.96	18.10	12.78	778.95	624.25	1455.98	246.89	1201.92	0.27	1.16	2.96	7.33	1.07
07	2.05	7.07	42.57	18.05	12.89	820.36	659.79	1419.52	372.16	1191.03	0.25	1.08	3.01	4.85	1.09
08	2.03	7.76	42.44	18.13	12.81	794.98	667.22	1394.43	280.81	1161.35	0.26	1.16	3.06	6.46	1.11
09	2.06	7.19	42.97	×	13.07	831.69	672.50	1397.02	×	1199.17	0.25	1.09	3.10	×	1.10
10	2.12	7.14	43.04	18.60	13.06	847.30	683.92	1505.28	340.53	1233.04	0.26	1.07	2.87	5.46	1.07
avg	2.07	7.19	42.71	18.18	12.88	804.90	651.03	1400.06	290.47	1172.66	0.26	1.12	3.09	6.52	1.11

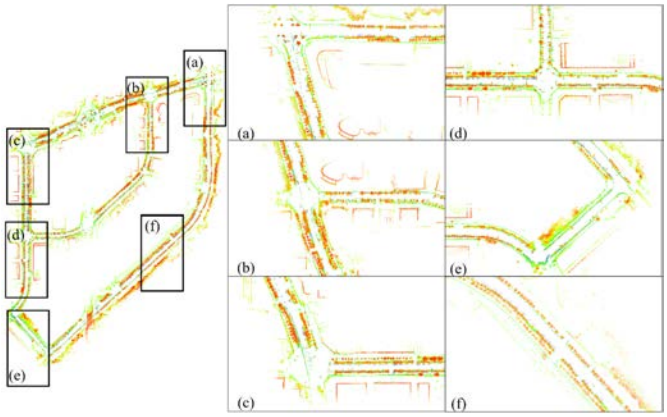

Fig. 18. LiDAR map generated by HVL-SLAM from Urban sequence with zoomed-in views. (a)–(f) Highlighting detailed sections of the left area.

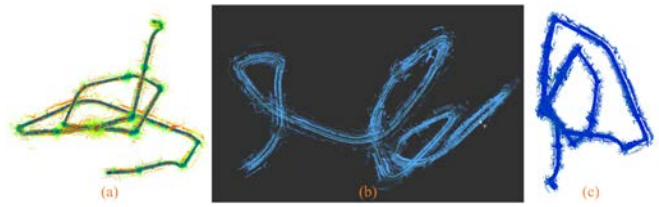

Fig. 19. Mapping results. (a)–(c) Correspond to LeGO-LOAM, GAC-Mapping, and MULLS, respectively.

1) Experiment of Depth Fitting: To validate the effectiveness of the feature depth fitting algorithm proposed in this article, we replaced the feature depth extraction modules of LIMO [2], CamVox [3], Gac-Mapping [10], and our proposed LSDT method in the RGB-L [4] system. Experiments were conducted on the KITTI odometry dataset, and the results are shown in Table III. Meanwhile, to demonstrate efficiency advantages over other methods, we present the time efficiency comparison with other methods in Table VI.

2) Experiment of Hybrid Tracking and Joint Local Mapping: As shown in Table V, we build three kinds of systems, they are feature depth extraction module with hybrid tracking module (Ours(1)), feature depth extraction module with hybrid tracking module and LiDAR scan-to-map optimization module (Ours(2)), and feature depth extraction module with hybrid tracking module and visual-LiDAR jointly local optimization module(Ours(3)), respectively.

TABLE V
RELATIVE TRANSLATION (%)/ROTATION (DEG/100 M) ERROR ON THE WHOLE KITTI TRAINING DATASET

seq.	ORB-SLAM (mono)	LeGO-LOAM	CamVox	Ours(1)	Ours(2)	Ours(3)
00	2.63/0.29	1.51/0.7	1.46/0.61	0.72/0.30	0.69/0.31	0.65/0.27
01	-	-	-	6.87/2.21	2.66/0.74	1.74/0.39
02	5.99/0.27	1.96/0.78	1.62/0.53	0.93/0.25	1.03/0.49	0.79/0.23
03	1.86/0.36	1.41/1.00	0.84/0.31	1.23/0.24	1.12/0.21	0.81/0.39
04	1.78/0.20	1.69/0.83	0.70/0.71	0.77/0.45	0.37/0.41	0.34/0.32
05	2.71/0.27	1.01/0.54	0.62/0.37	0.71/0.28	0.62/0.30	0.46/0.20
06	6.06/0.22	0.90/0.47	1.04/0.60	0.81/0.25	0.57/0.22	0.54/0.22
07	4.92/1.89	0.81/0.56	0.61/0.38	0.83/0.45	0.60/0.34	0.53/0.29
08	13.73/0.32	1.48/0.68	2.24/0.83	1.15/0.37	1.06/0.32	1.02/0.29
09	3.56/0.51	1.57/0.80	0.70/0.33	0.86/0.25	0.77/0.31	0.64/0.27
10	6.34/1.20	1.81/0.85	0.76/0.37	0.73/0.27	0.70/0.31	0.70/0.28
avg	4.96/0.55	1.42/0.72	1.06/0.50	1.41/0.48	0.93/0.37	0.75/0.29

The loop detection modules in these methods are disabled for odometry-only comparison. Since the scale is unknown in the monocular version of ORB-SLAM2, the trajectory coordinate, as well as the scale are aligned before trajectory evaluation. The comparison results show that the feature depth extraction module reduces the translation error to a great extent compared with monocular ORB-SLAM2 (from 4.96% down to 1.41%). Furthermore, compared to CamVox which requires a dense point cloud to provide depth, our proposed depth fitting method also shows better ego-motion estimation performance. In addition, the hybrid tracking module also plays an important role in improving the robustness of pose estimation, especially for degraded scenes, such as the highway. The scan-to-map matching method is used widely in LiDAR-based SLAM systems to maintain the accuracy of ego-motion estimation. In Ours(2), one can see from Table V, the scan-to-map module dramatically reduced the pose error, and the average relative translation error reduced from 1.41% to 0.93% while the average relative rotation error reduced from 0.48° /100 m to 0.37°/100 m. The translation accuracy and orientation accuracy are significantly improved compared to the LeGO-LOAM that also uses scan-to-map optimization. Furthermore, when visual measurements are added to the objective function with dynamic weights, the translation and rotation error drop by about 20%, thus we can conclude that the joint optimization can largely improve the pose estimation accuracy in our system.

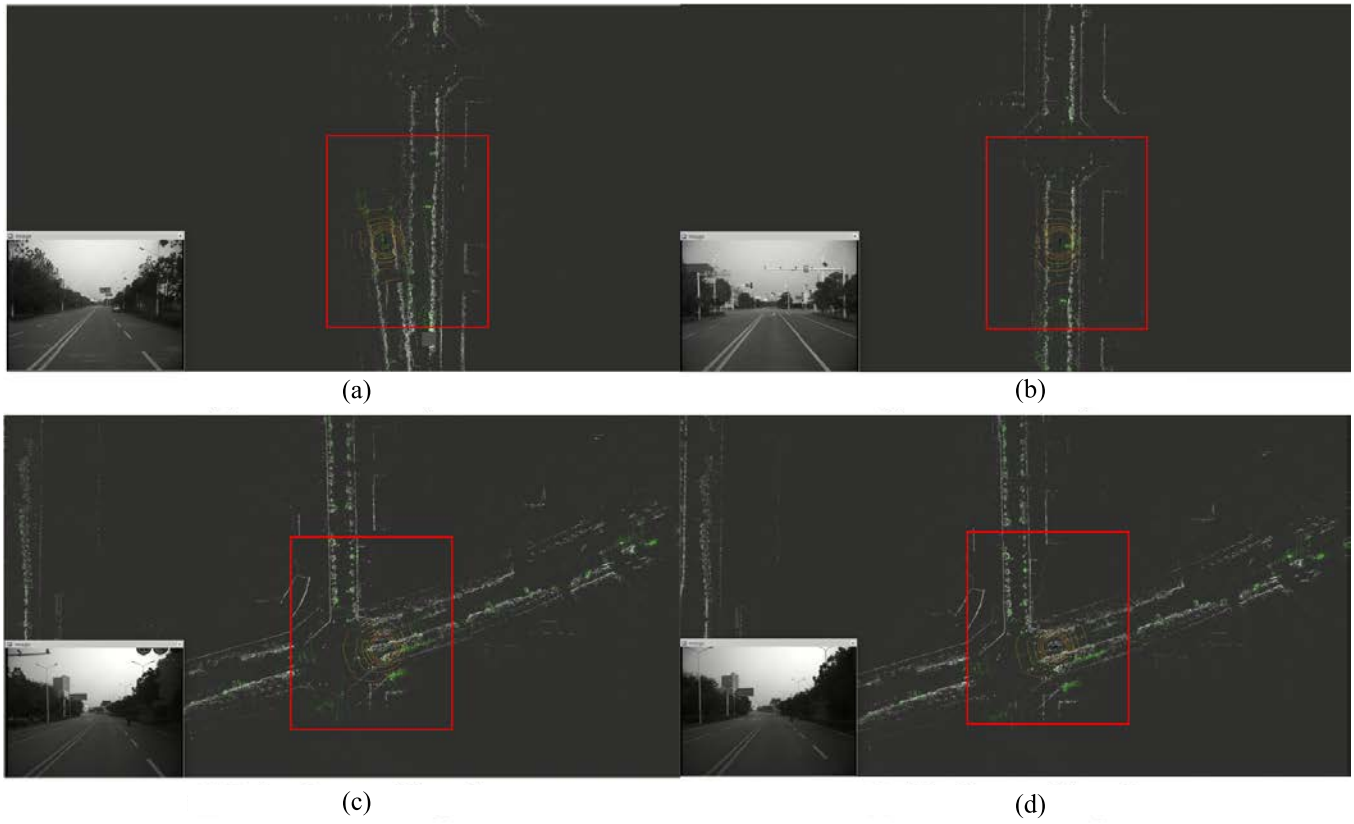

Fig. 20. Loop-closure detection and pose graph optimization results of HVL-SLAM in experiment III. Ground LiDAR points omitted for better visualization. (a) Before the first loop closure. (b) After the first loop closure. (c) Before the second loop closure. (b) After the second loop closure.

TABLE VI
TIME COST(MS)

modules	KITTI-00 datdaset		Jiangxia dataset	
	GAC-Mapping	Ours	GAC-Mapping	Ours
depth extraction	42.85	12.61	28.01	2.18
frame-to-frame tracking	113.01	24.00	87.39	9.52
local mapping	62.32	98.49	22.38	106.13

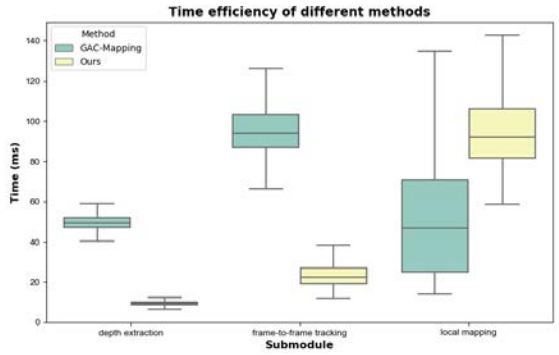

Fig. 21. Time efficiency of GAC-mapping and ours system in KITTI-00 dataset.

D. Timing Analysis

All the computations are conducted by a computer (Intel Core i7-7700 CPU, 32-GB RAM) in a Linux operation system. The software is implemented in C++ with robot operating system (ROS) communication interface. The time cost of the key operations is presented by Table VI. Although the visual LiDAR joint optimization (local mapping) is time-consuming, it only performs when a keyframe is generated, and with the help of OpenMP, our algorithm operates at an average frequency of 8Hz in KITTI dataset. Fig. 21 visualizes the comparison of time efficiency between the proposed method and the GAC-Mapping [10] method across various modules.

V. CONCLUSION

This article presents HVL-SLAM, an accurate and efficient LiDAR monocular SLAM approach. It comprises of three main parts: a simple but effective feature depth extraction module that based on LiDAR segmentation and Delaunay triangles is used to furnish depth information even with sparse LiDAR sensor; a hybrid visual-LiDAR tracking module that uses both photometric error and reprojection error to provide robust and accurate ego-motion estimation; a visual and LiDAR joint optimization module is applied to further improve the pose estimation accuracy each time a new keyframe is generated while keeping efficiency. Experiments on the KITTI odometry benchmark and our collected data in various environments have shown that our proposed HVL-SLAM is effective and robust, and can provide high-quality fusion mapping results over the state-of-the-art visual LiDAR odometry frameworks, especially in the feature degraded scenarios.

In future work, the system will be further optimized by incorporating a dynamic objects culling module to improve the accuracy and robustness of ego-motion estimation, and mapping tasks in dynamic environments.

REFERENCES

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

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

[3] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "CamVox: A low-cost and accurate LiDAR-assisted visual SLAM system," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, May 2021, pp. 5049–5055.

[4] F. Sauerbeck, B. Obermeier, M. Rudolph, and J. Betz, "RGB-L: Enhancing indirect visual SLAM using LiDAR-based dense depth maps," in *Proc. 3rd Int. Conf. Comput., Control Robot. (ICCCR)*, Mar. 2023, pp. 95–100.

[5] Y.-S. Shin, Y. S. Park, and A. Kim, "Direct visual SLAM using sparse depth for camera-LiDAR system," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Brisbane, QC, Australia, May 2018, pp. 5144–5151.

[6] K. Huang, J. Xiao, and C. Stachniss, "Accurate direct visual-laser odometry with explicit occlusion handling and plane detection," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, Montreal, QC, Canada, May 2019, pp. 1295–1301.

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

[8] A. Reinke, X. Chen, and C. Stachniss, "Simple but effective redundant odometry for autonomous vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, May 2021, pp. 9631–9637.

[9] Y. Seo and C.-C. Chou, "A tight coupling of vision-LiDAR measurements for an effective odometry," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Paris, France, Jun. 2019, pp. 1118–1123.

[10] J. He, Y. Zhou, L. Huang, Y. Kong, and H. Cheng, "Ground and aerial collaborative mapping in urban environments," *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 95–102, Jan. 2021, doi: 10.1109/LRA.2020.3032054.

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

[12] S.-S. Huang, Z.-Y. Ma, T.-J. Mu, H. Fu, and S.-M. Hu, "LiDAR-monocular visual odometry using point and line features," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Paris, France, May 2020, pp. 1091–1097.

[13] X. Chen, I. Vizzo, T. Läbe, J. Behley, and C. Stachniss, "Range image-based LiDAR localization for autonomous vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 5802–5808.

[14] A. Y. Hata and D. F. Wolf, "Feature detection for vehicle localization in urban environments using a multilayer LiDAR," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 2, pp. 420–429, Feb. 2016.

[15] Z. Wang, J. Fang, X. Dai, H. Zhang, and L. Vlacic, "Intelligent vehicle self-localization based on double-layer features and multilayer LiDAR," *IEEE Trans. Intell. Vehicles*, vol. 5, no. 4, pp. 616–625, Dec. 2020.

[16] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1437–1451, Feb. 2018.

[17] D. Galvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012.

[18] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12708–12717.

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

[20] C. Park, P. Moghadam, 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.

[21] W. Wang, J. Liu, C. Wang, B. Luo, and C. Zhang, "DV-LOAM: Direct visual LiDAR odometry and mapping," *Remote Sens.*, vol. 13, no. 16, p. 3340, 2021.

[22] Z. Yuan, Q. Wang, K. Cheng, T. Hao, and X. Yang, "SDV-LOAM: Semi-direct visual-LiDAR odometry and mapping," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 9, pp. 11203–11220, Sep. 2023, doi: 10.1109/TPAMI.2023.3262817.

[23] K. Cao et al., "Tightly-coupled LiDAR-visual SLAM based on geometric features for mobile agents," in *Proc. IEEE Int. Conf. Robot. Biomimetics (ROBIO)*, Koh Samui, Thailand, Dec. 2023, pp. 1–8, doi: 10.1109/robio58561.2023.10354794.

[24] C. Shu and Y. Luo, "Multi-modal feature constraint based tightly coupled monocular visual-LiDAR odometry and mapping," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 5, pp. 3384–3393, May 2023, doi: 10.1109/TIV.2022.3215141.

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

[26] S. Lee, H. Lim, and H. Myung, "Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3D point cloud," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Kyoto, Japan, Oct. 2022, pp. 13276–13283, doi: 10.1109/IROS47612.2022.9981561.

[27] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3D laser scans for online operation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Daejeon, South Korea, Oct. 2016, pp. 163–169.

[28] H. Lim, M. Oh, and H. Myung, "Patchwork: Concentric zone-based region-wise ground segmentation with ground likelihood estimation using a 3D LiDAR sensor," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6458–6465, Oct. 2021.

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

[30] T. Qin and S. Cao. (2018). *A-loam*. [Online]. Available: https://github.com/HKUST-Aerial-Robotics/A-LOAM

[31] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM: Fast LiDAR odometry and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 4390–4396.

[32] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "MULLS: Versatile LiDAR SLAM via multi-metric linear least square," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, May 2021, pp. 11633–11640.

[33] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot., Sci. Syst.*, D. Fox, L. E. Kavraki, and H. Kurniawati, Eds., Berkeley, CA, USA: Univ. California, 2014, pp. 1–9.

[34] 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. (IROS)*, Madrid, Spain, Oct. 2018, pp. 4758–4765.

[35] S. Agarwal, K. Mierle, and T. C. S. Team. (Mar. 2022). *Ceres Solver*. [Online]. Available: https://github.com/ceres-solver/ceres-solver

[36] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Voxelized GICP for fast and accurate 3D point cloud registration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Xi'an, China, May 2021, pp. 11054–11059.

[37] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[38] P. Zhou, X. Guo, X. Pei, and C. Chen, "T-LOAM: Truncated least squares LiDAR-only odometry and mapping in real time," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–13, 2022, Art. no. 5701013, doi: 10.1109/TGRS.2021.3083606.

[39] M. Grupp. (2017). *Evo: Python Package for the Evaluation of Odometry and Slam*. [Online]. Available: https://github.com/MichaelGrupp/evo