Towards High-Performance Solid-State-LiDAR-Inertial Odometry and Mapping

Kailai Li, Meng Li, and Uwe D. Hanebeck

***Abstract*—We present a novel tightly-coupled LiDAR-inertial odometry and mapping scheme for both solid-state and mechanical LiDARs. As frontend, a feature-based lightweight LiDAR odometry provides fast motion estimates for adaptive keyframe selection. As backend, a hierarchical keyframe-based sliding window optimization is performed through marginalization for directly fusing IMU and LiDAR measurements. For the Livox Horizon, a newly released solid-state LiDAR, a novel feature extraction method is proposed to handle its irregular scan pattern during preprocessing. LiLi-OM (Livox LiDAR-inertial odometry and mapping) is real-time capable and achieves superior accuracy over state-of-the-art systems for both LiDAR types on public data sets of mechanical LiDARs and in experiments using the Livox Horizon. Source code and recorded experimental data sets are available at https://github.com/KIT-ISAS/lili-om.**

***Index Terms*—Sensor fusion, localization, mapping**

I. Introduction

ESTIMATING six-DoF egomotion plays a fundamental role in a wealth of applications ranging from robot navigation and inspection to virtual/augmented reality, see [1]–[4]. With the booming of autonomous driving, light detection and ranging (LiDAR) sensors have gained tremendous popularity [5], [6]. Compared to visual sensors, 3D LiDARs provide lighting-invariant and accurate perception of the surroundings with long detection range and high robustness. Thus, they are broadly deployed to mobile agents for odometry and mapping.

Essentially, LiDAR-based odometry requires computing six-DoF egomotion given consecutive frames of point clouds. This can be performed via scan-matching algorithms, e.g., the iterative closest point (ICP) [7] method. In typical mobile perception scenarios, 3D LiDARs output a large streaming volume of raw scans in the form of unorganized point clouds. Meanwhile, real-time processing (e.g., at 10 Hz) is required given limited computing resources. To improve the performance of LiDAR odometry, much attention has been dedicated to scan-matching with points representing local geometric features [8], [9]. In [10], feature points are extracted from object edges and planes for scan-matching using point-to-edge and point-to-plane metrics, which enable accurate LiDAR odometry in real time. Based thereon, in [11], an image-based approach [12] was further applied to preprocessing. A two-stage optimization scheme was tailored for ground vehicles to enable light-weight, robust LiDAR odometry and mapping.

This work was supported by German Research Foundation (DFG) under grant HA 3789/16-1.

The authors are with the Intelligent Sensor-Actuator-Systems Laboratory (ISAS), Institute for Anthropomatics and Robotics, Karlsruhe Institute of Technology (KIT), Germany. E-mails: `kailai.li@kit.edu`, `uyjjl@student.kit.edu`, `uwe.hanebeck@kit.edu`

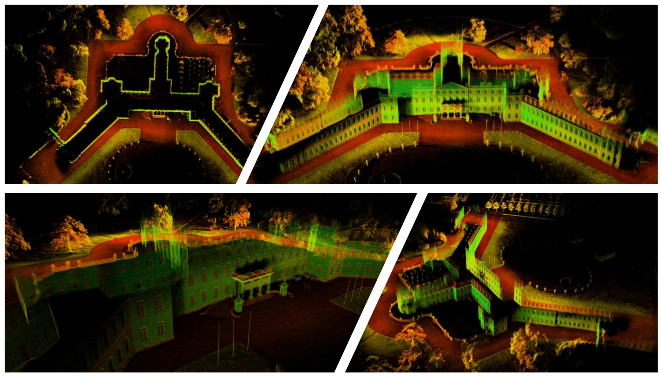


Fig. 1: 3D map of Schloss Karlsruhe from LiLi-OM.

Typical LiDAR scan rates are relatively low and the perceived point clouds are in principle distorted due to sensor egomotion. Thus, performing LiDAR-only odometry is prone to deterioration under fast motion or in complex scenes. Inertial sensors, however, measure instant motion at a much higher frequency and can bridge the gap between consecutive LiDAR frames, improving the robustness and accuracy of LiDAR-based egomotion estimation. In [10], [11], an inertial measurement unit (IMU) was used to de-skew point clouds and initialize motion estimates for scan-matching-based LiDAR odometry. Though decoupled or loosely-coupled LiDAR-inertial fusion schemes are appealing for runtime and deploying classic recursive filters (e.g., the EKF), they may cause information loss and inaccurate estimates [13].

Thus, there has been a growing focus on tightly-coupled LiDAR-inertial odometry, where point cloud and IMU measurements are fused in a joint optimization or filtering framework. Pre-integrated IMU readings are often employed for de-skewing the LiDAR scan per frame [14]. In [15], an optimization-based approach was proposed for LiDAR-inertial odometry using a maximum a posteriori (MAP) formulation incorporating both LiDAR and IMU residuals in a sliding window fashion. An additional method using rotational constraints was proposed to refine the final pose and map, which delivers similar or better tracking accuracy as [10]. However, real-time processing is hard to achieve in practice as sensor readings of every frame are exploited.

To improve the runtime efficiency of LiDAR-inertial odometry, an iterated error-state Kalman filter was introduced in [16] based on a robocentric formulation. It runs in real time and shows superior tracking accuracy over existing LiDAR-only odometry systems. In [17], a tightly-coupled LiDAR-inertial odometry system was proposed based on the incremental smoothing and mapping framework iSAM2 [18]. However, the system relies heavily on [10] to produce LiDAR odometry factors for further constraining the pre-integrated IMU states in the factor graph formulation. Unlike direct fusion of IMU and LiDAR measurements within a unified scheme, this can result in loss of constraint information posed by landmarks. Also, correlations between LiDAR and IMU measurements might be largely discarded. To guarantee high odometry accuracy, the system requires nine-axis IMU readings of high frequency (500 Hz as used in [17]) to de-skew the point cloud and initialize LiDAR odometry. Fusion with additional sensor modalities (e.g., GPS signals) is often needed at certain spots to achieve precise localization.

More importantly, conventional LiDARs rely on mechanical spinning mechanisms to enable a 360-degree FoV, while the vertical resolution is fairly limited. Though some products with high vertical resolution emerged recently (e.g., from Hesai[1] and Velodyne[2]), their very high prices prohibit mass market supply for robotics industry and research.

Very recently, *solid-state LiDAR*s have hit the consumer market with much better affordability based on various working principles. Existing types often have non-repetitive and irregular scan patterns with small FoVs to reach more uniform and higher resolution. So far, solid-state-LiDAR-based odometry has not been well investigated. In [19], the LiDAR odometry system in [10] was adapted to Livox Mid-40[3], a solid-state LiDAR with a circular FoV of $38.4°$. Compared with its baseline [10], it employs similar feature-based scan-matching and delivers comparable tracking accuracy with improved runtime via parallelization. To the best of the authors' knowledge, no published research on tightly-coupled solid-state-LiDAR-inertial fusion exists to date.

In this paper, we provide a specific study on solid-state-LiDAR-inertial odometry and mapping. Instead of Livox Mid-40, we choose Livox Horizon[4] (released in Q1, 2020), which is designed for vehicular perception with an FoV of $81.7° \times 25.1°$. Scanning at 10 Hz, it reaches a similar but more uniform FoV coverage compared with typical 64-line mechanical LiDARs. Moreover, it is substantially cheaper than most existing 3D LiDARs of comparable performance.

Common feature extraction methods for spinning LiDARs are not applicable for solid-state ones due to their irregular and non-repetitive scan patterns. For Livox Mid-40, scanning of a single laser head is specially regulated to form a circular coverage. Hence, the approach in [19] traverses along the incident and deflection angle to choose point candidates, at which the local smoothness of the scan line is computed for feature extraction as in [10]. This, however, cannot be applied to Livox Horizon as it sweeps in a rather unregulated manner. Such a pattern is more generic for reaching broader FoVs of uniform coverage, which can potentially become common for future types of solid-state LiDARs. But the limited FoVs can still thwart odometry performance in some circumstances, especially under fast motion or with insufficient features.

[1]`https://www.hesaitech.com/en/Pandar128`
[2]`https://velodynelidar.com/products/alpha-prime`
[3]`https://www.livoxtech.com/mid-40-and-mid-100`
[4]`https://www.livoxtech.com/de/horizon`

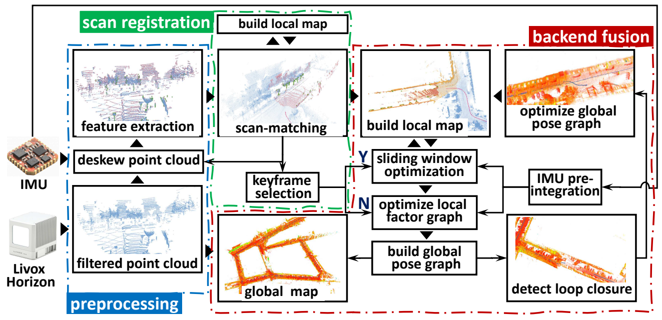


Fig. 2: System pipeline

Contributions

Considering the state of the art above, we propose a novel tightly-coupled LiDAR-inertial odometry and mapping scheme with a specific variant for solid-state LiDARs (pipeline given in Sec. II). A novel feature extraction approach is tailored to the irregular scan pattern of Livox Horizon (Sec. III). To directly fuse LiDAR and IMU measurements in a unified manner, a hierarchical keyframe-based scheme is proposed using sliding window optimization (Sec. IV). The proposed system is generically applicable for both conventional and the deployed solid-state LiDAR. It runs in real time and delivers superior odometry accuracy over existing systems (Sec. V). We release the proposed system with open-source code and new solid-state-LiDAR-inertial data sets recorded by Livox Horizon and Xsens MTi-670. Thanks to the low hardware costs and real-time performance on portable platforms, our system provides a cost-effective solution for mobile perception in various scenarios.

II. System Pipeline

The proposed LiDAR-inertial odometry and mapping scheme is shown in Fig. 2. The 3D LiDAR (e.g., Livox Horizon) streams out point clouds at a typical frequency of 10 Hz and is synchronized with a six-axis IMU providing gyroscope and accelerometer readings at higher frequency (e.g., 200 Hz for Xsens MTi-670[5]). We want to estimate the six-DoF egomotion of the LiDAR frame and obtain a globally consistent map simultaneously. The raw point clouds from LiDAR scan are first downsampled and rotationally de-skewed using gyroscope data. Then, feature points representing planes and edges are extracted (Sec. III-A). Given the preprocessed scans, a lightweight scan-matching-based registration module runs in a frame-to-model manner for fast motion estimation with point-to-edge and point-to-plane metrics being exploited (Sec. III-B). The obtained egomotion estimates are further used to de-skew the translational distortion of the current sweep as well as select keyframes adaptively to scene transitions. Parallel to preprocessing and LiDAR odometry, LiDAR and IMU measurements are fused at the backend in a unified manner via the proposed keyframe-based sliding window optimization (Sec. IV).

[5]`https://www.xsens.com/products/mti-600-series`

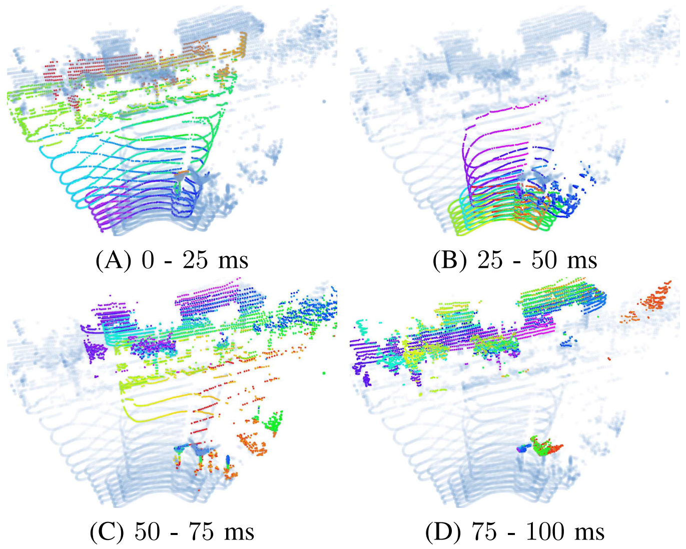


Fig. 3: Scan pattern of Livox Horizon with real-world scene. Points are rendered according to time stamps with rainbow color scale. Blue blobs depict the full sweep.

The fusion window usually covers several (e.g., three) keyframes. As the window slides, keyframe states (denoted with ˘ on top as follows) are optimized in the current window

$$\breve{\mathbf{x}} = \left[\breve{\mathbf{t}}^\top, \breve{\mathbf{v}}^\top, \breve{\mathbf{q}}^\top, \breve{\mathbf{b}}^\top\right]^\top \in \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{S}^3 \times \mathbb{R}^6 \subset \mathbb{R}^{16}. \quad (1)$$

Here, $\breve{\mathbf{t}} \in \mathbb{R}^3$ and $\breve{\mathbf{q}} \in \mathbb{S}^3$ denote the keyframe position and orientation (represented by unit quaternions), respectively. $\breve{\mathbf{v}} \in \mathbb{R}^3$ is the velocity and $\breve{\mathbf{b}} = [\breve{\mathbf{b}}_a^\top, \breve{\mathbf{b}}_g^\top]^\top \in \mathbb{R}^6$ denotes the term incorporating the IMU bias of the accelerometer (subscript 'a') and the gyroscope (subscript 'g'). After being slid over, the two oldest optimized keyframes are used as constraints to optimize the in-between regular-frame poses via factor graph optimization. Preintegrated inertial measurements are hereby exploited for pose initialization. A global pose graph is maintained to incorporate all poses of LiDAR frames. Loop closure is checked in a keyframe basis using ICP, and when necessary, a global graph optimization is invoked to guarantee the reconstructed map to be globally consistent.

III. Feature-Based Solid-State LiDAR Scan-Matching

A. Feature extraction for irregular scan pattern

Existing feature extraction methods for 3D LiDARs are not well applicable for Livox Horizon [11], [19]. Fig. 3 illustrates its scan pattern (at 10 Hz) by dividing one LiDAR sweep into four stages. The Livox Horizon is instrumented with a multi-laser sensing module with an array of six vertically-aligned laser diodes sweeping back and forth through the prisms non-repetitively. Consequently, the six point readings obtained from the multi-laser transceiver are vertically aligned and perceived simultaneously (rendered as the same color). The scan shows an irregular "brushing" pattern covering the 81.7° × 25.1° FoV uniformly. Over the integration time of 100 ms per frame, the angular resolution reaches 0.2° to 0.4° horizontally and vertically with a similar FoV coverage to 64-line spinning

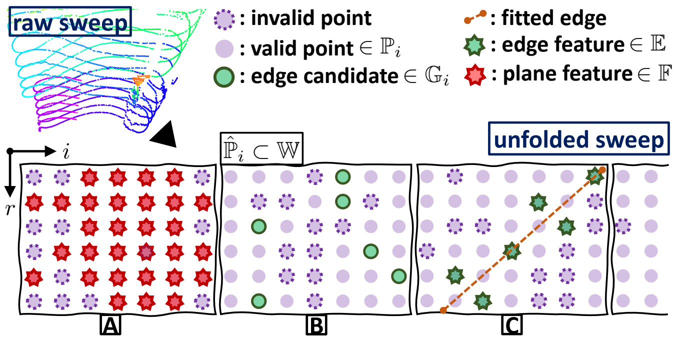

raw sweep
: invalid point
: valid point $\in \mathbb{P}_i$
: edge candidate $\in \mathbb{G}_i$
: fitted edge
: edge feature $\in \mathbb{E}$
: plane feature $\in \mathbb{F}$
$\hat{\mathbb{P}}_i \subset \mathbb{W}$
unfolded sweep
i
r
A
B
C

Fig. 4: Illustration of proposed feature extraction method.

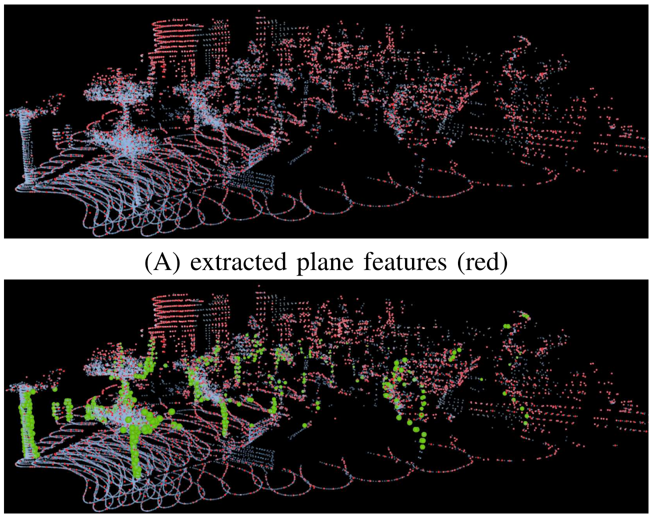
Algorithm 1: Feature Extraction for Livox Horizon

```
    Input: single sweep $\mathbb{W}$
    Output: edge feature set $\mathbb{E}$, plane feature set $\mathbb{F}$
1   $\mathbb{E} \leftarrow \emptyset, \mathbb{F} \leftarrow \emptyset$;
2   $\{\hat{\mathbb{P}}_i = \{\mathbf{X}_r\}_{r=1}^6 \,|\, \mathbf{X}_r \in \mathbb{R}^{3\times 7}\}_{i=1}^{\tau} \leftarrow$ split($\mathbb{W}$);
3   for $i \leftarrow 1$ to $\tau$ do
4       $\mathbb{P}_i \leftarrow$ getValidPoints($\hat{\mathbb{P}}_i$);
5       $\Sigma_i \leftarrow$ computeCovariance($\mathbb{P}_i$);
6       $\{(\lambda_1, \lambda_2, \lambda_3)\} \leftarrow$ eig($\Sigma_i$);      // $\lambda_1 \le \lambda_2 \le \lambda_3$
        /* check plane feature */
7       if $\lambda_1/\lambda_2 < 0.3$ then
8           $\mathbb{F} \leftarrow \mathbb{F} \cup \mathbb{P}_i$;
9       else
            /* check edge feature */
10          $\mathbb{G}_i \leftarrow \emptyset$;
11          for $r \leftarrow 1$ to 6 do
12              $\mathbf{x}_r \leftarrow$ getCurv($\mathbf{X}_r$);
13              $\mathbb{G}_i \leftarrow \mathbb{G}_i \cup \mathbf{X}_r$;
14          $\Gamma_i \leftarrow$ computeCovariance($\mathbb{G}_i$);
15          $\{(\lambda_1, \lambda_2, \lambda_3)\} \leftarrow$ eig($\Gamma_i$);
16          if $\lambda_2/\lambda_3 < 0.25$ then
17              $\mathbb{E} \leftarrow \mathbb{E} \cup \mathbb{G}_i$;
```

LiDARs. Compared with Mid-40 that regulates its single laser head for circular FoV coverage, Horizon has a more irregular scan pattern due to its rather unregulated sweeping motion.

To extract plane and edge feature points for Livox Horizon, we propose a new two-stage approach in Alg. 1 (illustrated in Fig. 4). Given a frame of raw scan, we unfold and split the sweep $\mathbb{W}$ in its time domain, where 6 × 7-point patches are assigned one after another without overlapping (Alg. 1, line 2). In each raw patch $\hat{\mathbb{P}}_i$, inconcrete point readings are first removed (Alg. 1, line 4). For the valid points $\mathbb{P}_i$, we perform eigendecomposition of the covariance of their 3D coordinates (Alg. 1, line 5-6). If the second largest eigenvalue is substantially larger than the smallest one ($\lambda_1/\lambda_2 < 0.3$), then all points in the patch are extracted as plane features (Fig. 4-A and Alg. 1, line 7-8). For a non-plane patch, we search the point with largest curvature on each scan line and perform eigendecomposition for the six points (Alg. 1, line 9-15). If the largest eigenvalue is substantially larger than the second largest one ($\lambda_2/\lambda_3 < 0.25$), then the points form a line and they are extracted as edge features (Fig. 4-C and Alg. 1, line 16-17). Otherwise, no feature is extracted from the current patch (Fig. 4-B).

We show an example of extracted features in one frame of Livox Horizon scan in Fig. 5. Note that the proposed feature extraction algorithm is purely performed in the time domain of every sweep given time stamps of the perceived point array. This is radically different from the approach in [19] that computes local smoothness of point candidates selected through spatial retrieval. Also for traditional spinning LiDARs, common point cloud segmentation or feature extraction methods [10], [11], [20] are performed in (transformed) spatial domains, e.g., extracting features w.r.t. the horizontal scan angles. For Livox Horizon, the number of extracted plane features are usually much more than edge features due to its uniform scan coverage (statistics are given in Sec. V-D). We further associate each feature point with its corresponding edge's direction vector or plane's normal vector to represent local geometries. It will be further exploited for weighting the LiDAR residual term in the backend fusion module.


(A) extracted plane features (red)

(B) extracted edge (green) and plane (red) features

Fig. 5: Feature extraction for point cloud (blue) from Horizon.

B. Point-to-edge and point-to-plane metric

Given the extracted edge and plane features, both the frontend registration module and the sliding window optimization for backend fusion in Fig. 2 exploit the scan-matching formulation to incorporate LiDAR measurements for egomotion estimation. Following metrics are applied.

Point-to-edge metric: Suppose an edge feature $\mathbf{p}^{\ell} \in \mathbb{E}$ is extracted from sweep $\mathbb{W}$ w.r.t. the LiDAR frame ℓ and its associated unit vector $\boldsymbol{\nu}_{\mathrm{e}}$ indicating the edge direction is available. We first search its nearest five edge points in the corresponding local feature map $\mathbb{M}^{\omega}$ and compute their coordinates' mean value $\bar{\mathbf{e}}^{\omega}$ and covariance matrix (w.r.t. the global frame ω). Based thereon, an eigendecomposition is performed. If the largest eigenvalue is significantly larger than the rest, the five points in $\mathbb{M}^{\omega}$ form a line with its direction vector $\mathbf{n}_{\mathrm{e}}$ being the eigenvector corresponding to the largest eigenvalue. We then take two points $\acute{\mathbf{e}}^{\omega} = \bar{\mathbf{e}}^{\omega} + \delta \mathbf{n}_{\mathrm{e}}$ and $\grave{\mathbf{e}}^{\omega} = \bar{\mathbf{e}}^{\omega} - \delta \mathbf{n}_{\mathrm{e}}$ on the fitted line and exploit a point-to-edge metric of the following form

$$\mathscr{D}_{\mathrm{e}}(\mathbf{x}^{\omega}, \mathbf{p}^{\ell}, \mathbb{M}^{\omega}) = \frac{\|(\mathbf{p}^{\omega} - \acute{\mathbf{e}}^{\omega}) \times (\mathbf{p}^{\omega} - \grave{\mathbf{e}}^{\omega})\|}{\|\acute{\mathbf{e}}^{\omega} - \grave{\mathbf{e}}^{\omega}\|} .$$

Here, $\mathbf{p}^{\omega} = \mathbf{R}(\mathbf{q})\,\mathbf{p}^{\ell} + \mathbf{t}$ denotes the scan point w.r.t. the global frame given current LiDAR pose $\mathbf{x}^{\top} = [\,\mathbf{q}^{\top}, \mathbf{t}^{\top}\,]^{\top}$. $\mathbf{R}(\mathbf{q})$ is the rotation matrix given by $\mathbf{q}$. Typically, $\delta = 0.1$.

Point-to-plane metric: If the feature point $\mathbf{p}^{\ell}$ indicates a plane (with a normal $\boldsymbol{\nu}_{\mathrm{s}}$), we also transform it into its world coordinates $\mathbf{p}^{\omega}$ and find its nearest five plane feature points in the corresponding local feature map w.r.t. the global frame, namely $\mathbb{S}^{\omega} = \{\mathbf{s}_j^{\omega}\} \subset \mathbb{M}^{\omega}$, with $j = 1, ..., 5$. Similar to the implementation for [10], we solve an overdetermined linear equation $\mathbf{A}_{\mathrm{s}}\mathbf{u}_{\mathrm{s}} = \mathbf{c}$ via QR decomposition for plane fitting, with $\mathbf{A}_{\mathrm{s}} = [\,\mathbf{s}_1^{\omega}, ..., \mathbf{s}_5^{\omega}\,] \in \mathbb{R}^{5\times 3}$ and $\mathbf{c} = [\,-1, ..., -1\,] \in \mathbb{R}^5$. We normalize the fitted normal vector as $\mathbf{n}_{\mathrm{s}} = \mathbf{u}_{\mathrm{s}} / \, ||\mathbf{u}_{\mathrm{s}}||$ and the point-to-plane metric can be established as

$$\mathscr{D}_{\mathrm{s}}(\mathbf{x}^{\omega}, \mathbf{p}^{\ell}, \mathbb{M}^{\omega}) = |\mathbf{u}_{\mathrm{s}}^{\top}\mathbf{p}^{\omega} + 1| / \, ||\mathbf{u}_{\mathrm{s}}|| = \left|\mathbf{n}_{\mathrm{s}}^{\top}\mathbf{p}^{\omega} + 1/ \, ||\mathbf{u}_{\mathrm{s}}||\,\right| .$$

Here, we also have $\mathbf{p}^{\omega} = \mathbf{R}(\mathbf{q})\,\mathbf{p}^{\ell} + \mathbf{t}$.

C. Metric weighting function

In order to quantify the contribution of each LiDAR residual during sensor fusion, we propose a metric weighting function according to the association quality as follows

$$\upsilon_{\circ}(\mathbf{p}^{\ell}) = \lambda \cdot (\boldsymbol{\nu}_{\circ})^{\top}\mathbf{n}_{\circ} \cdot \exp\big(-\textstyle\sum_{j=1}^{5} |\gamma(\mathbf{p}^{\ell}) - \gamma_j|\big) .$$

Here, $\circ$ indicates the type of feature correspondences, namely edges (subscript 'e') or planes (subscript 's'). For an edge feature correspondence, $\boldsymbol{\nu}_{\mathrm{e}}$ and $\mathbf{n}_{\mathrm{e}}$ denote the direction vector of the edge line at $\mathbf{p}^{\ell}$ and the line approximated by its nearest five edge features, respectively. Similarly, $\boldsymbol{\nu}_{\mathrm{s}}$ and $\mathbf{n}_{\mathrm{s}}$ denote the plane normal at point $\mathbf{p}^{\ell}$ and the one formed by its nearest five plane features, respectively. Moreover, $\gamma(\mathbf{p}^{\ell})$ and γ_j are the reflectance values of the feature point $\mathbf{p}^{\ell}$ and its associated nearest five features, respectively. Therefore, the proposed metric weighting function considers both geometric and appearance consistencies of feature associations. We set the constant $\lambda = 15$ from experience.

D. Feature-based scan-matching

At the frontend, we perform a light-weight, feature-based scan-matching in a frame-to-model manner. A trust region method is used to estimate the current pose $\mathbf{x}^{\omega}$ by minimizing

$$\min_{\mathbf{x}^{\omega}} \left\{ \sum_{i=1}^{m} \big(\mathscr{D}_{\mathrm{e}}(\mathbf{x}^{\omega}, \mathbf{p}_i^{\ell}, \mathbb{M}^{\omega})\big)^2 + \sum_{j=1}^{n} \big(\mathscr{D}_{\mathrm{s}}(\mathbf{x}^{\omega}, \mathbf{p}_j^{\ell}, \mathbb{M}^{\omega})\big)^2 \right\},$$

which incorporates all the edge ($\mathbf{p}_i^{\ell} \in \mathbb{E}$) and plane ($\mathbf{p}_j^{\ell} \in \mathbb{F}$) correspondences between the current scan and the local map $\mathbb{M}^{\omega}$. Here, $\mathscr{D}_{\mathrm{e}}$ and $\mathscr{D}_{\mathrm{s}}$ denote the point-to-edge and point-to-plane metrics, respectively. The frontend local feature map $\mathbb{M}^{\omega}$ is updated with a width of typically 20 recent frames given the optimized poses. The cost function above is similar to other popular LiDAR odometry systems [10], [15], [17], [19]. However, we restrict the optimization time for fast motion estimation, based on which the translational scan distortion

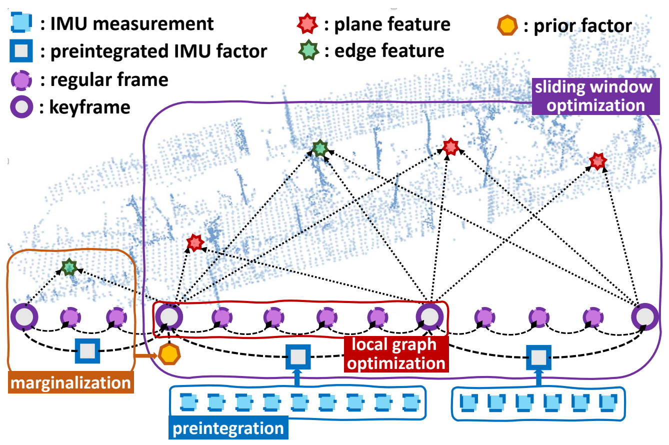


Fig. 6: Proposed tightly-coupled LiDAR-inertial fusion scheme using keyframe-based sliding window optimization.

is corrected and keyframes are selected. The accuracy of state estimation is pursued using the proposed backend fusion scheme shown below.

IV. Tightly-Coupled LiDAR-Inertial Fusion via Keyframe-Based Sliding Window Optimization

A. *Keyframe-based fusion hierarchy*

Keyframe-based schemes have originally been proposed and widely applied to visual odometry to achieve accurate tracking in real time [21]. In [17], keyframes from LiDAR odometry frontend [11] are fused to constrain IMU factors via iSAM2 (thus indirect fusion). The tightly-coupled LiDAR-inertial odometry in [15] realized direct fusion of LiDAR and pre-integrated IMU measurements via sliding window optimization. However, real-time performance is not generally achievable as the scheme fuses LiDAR sweeps of every frame.

Therefore, it is of importance to maintain the sparsity of the optimization scheme for direct LiDAR-inertial fusion at the backend. Shown in Fig. 6, the proposed fusion scheme exploits keyframes to establish sliding windows, where LiDAR and pre-integrated IMU measurements at keyframes are fused in a unified manner via nonlinear optimization. As the window slides forward after optimization, we construct a local factor graph incorporating the two oldest keyframe poses as constraints and the regular frames poses initialized by IMU measurements. A small-scale factor graph optimization is invoked to obtain regular-frame poses at LiDAR frequency.

Setting up keyframes can critically affect odometry accuracy due to the IMU drift during the time interval of two consecutive keyframes. We introduce two criteria for keyframe selection: (1) If the overlapping ratio between features of the current frame and the local feature map is smaller than 60% or (2) if the time difference to the last keyframe is more than a certain number of (e.g., two) regular frames, the current frame is then selected as a new keyframe. Here, restricting the frame interval between keyframes helps mitigate the drift issue for IMU pre-integration.

B. *Sliding window optimization for keyframes*

We compute keyframe states of form (1) by directly fusing LiDAR and pre-integrated IMU measurements, all observed at keyframes. For a sliding window of τ_{w}-keyframe width, the optimal keyframe states $\breve{\mathbf{X}} = [\breve{\mathbf{x}}_1^\top, ..., \breve{\mathbf{x}}_{\tau_{\mathrm{w}}}^\top]^\top \in \mathbb{R}^{15\times\tau_{\mathrm{w}}}$ are obtained by minimizing

$$\min_{\breve{\mathbf{X}}^w} \left\{ \|\mathscr{R}_{\mathrm{P}}(\breve{\mathbf{X}}^w)\|^2 + \sum_{k=1}^{\tau_{\mathrm{w}}} \mathscr{J}_{\mathrm{L}}(\breve{\mathbf{x}}_k^w) + \sum_{k=1}^{\tau_{\mathrm{w}}} \mathscr{J}_{\mathrm{I}}(\breve{\mathbf{x}}_k^w) \right\} \quad (2)$$

in the form of maximum a posterior (MAP). $\mathscr{R}_{\mathrm{P}}(\breve{\mathbf{X}}^w)$ denotes the prior residual term representing the measurements that are marginalized out due to window-sliding. $\mathscr{J}_{\mathrm{L}}(\breve{\mathbf{x}}_k^w)$ and $\mathscr{J}_{\mathrm{I}}(\breve{\mathbf{x}}_k^w)$ denote the keyframe-wise LiDAR and IMU error terms. Details about the three components follow.

Prior factor: In order to bound the computational burden without substantial information loss, we exploit marginalization in the sliding window optimization. Here, the oldest keyframe and its measurements are marginalized out via Schur-complement [22]. A new prior is computed accordingly and added on top of the existing prior factor to carry the estimate from the removed keyframe to the next window.

LiDAR term: The LiDAR term incorporates geometric constraints from LiDAR measurements into the fusion scheme. When aligning the observed edge ($\breve{\mathbf{p}}_{k,i}^{\ell} \in \breve{\mathbb{E}}_k$) and plane ($\breve{\mathbf{p}}_{k,j}^{\ell} \in \breve{\mathbb{F}}_k$) features to the local feature map $\breve{\mathbb{M}}_k^w$ observed by recent 30 keyframes, the term is defined as $\mathscr{J}_{\mathrm{L}}(\breve{\mathbf{x}}_k^w) =$

$$\sum_{i=1}^{m} \tilde{\upsilon}_{\mathrm{e},i} \big(\mathscr{D}_{\mathrm{e}}(\breve{\mathbf{x}}_k^w, \breve{\mathbf{p}}_{k,i}^{\ell}, \breve{\mathbb{M}}_k^w)\big)^2 + \sum_{j=1}^{n} \tilde{\upsilon}_{\mathrm{s},j} \big(\mathscr{D}_{\mathrm{s}}(\breve{\mathbf{x}}_k^w, \breve{\mathbf{p}}_{k,j}^{\ell}, \breve{\mathbb{M}}_k^w)\big)^2.$$

Here, $\mathscr{D}_{\mathrm{e}}$ and $\mathscr{D}_{\mathrm{s}}$ are the point-to-edge and point-to-plane metrics in Sec. III-B with fixed feature correspondences, respectively. $\tilde{\upsilon}_{\mathrm{e}}$ and $\tilde{\upsilon}_{\mathrm{s}}$ denote normalized weights among feature correspondences of each type. The local map is updated given the optimized poses as the window slides.

IMU term: The error term for IMU incorporates the relative motion constraints between keyframes into the fusion scheme. To avoid repropagating IMU states each time the optimization window slides, raw inertial readings are pre-integrated between two consecutive keyframes k and $k+1$ as in [15], [22]. The term is defined as

$$\mathscr{J}_{\mathrm{I}}(\breve{\mathbf{x}}_k^w) = \|\mathscr{R}_{\mathrm{I}}(\breve{\mathbf{x}}_k^w, \hat{\mathbf{z}}_{k+1}^k)\|^2_{\mathbf{C}_{k+1}^k}, \text{with}\, \mathscr{R}_{\mathrm{I}}(\breve{\mathbf{x}}_k^w, \hat{\mathbf{z}}_{k+1}^k) =$$

$$\begin{bmatrix} \breve{\mathbf{R}}_k^{w\top}\big(\breve{\mathbf{t}}_{k+1}^w - \breve{\mathbf{t}}_k^w + \frac{1}{2}\mathbf{g}^w\Delta\tau_k^2 - \mathbf{v}_k^w\Delta\tau_k\big) - \hat{\boldsymbol{\alpha}}_{k+1}^k \\ \breve{\mathbf{R}}_k^{w\top}\big(\breve{\mathbf{v}}_{k+1}^w + \mathbf{g}^w\Delta\tau_k - \breve{\mathbf{v}}_k^w\big) - \hat{\boldsymbol{\beta}}_{k+1}^k \\ 2\big[(\mathbf{q}_k^w)^{-1} \otimes \mathbf{q}_{k+1}^w \otimes (\hat{\boldsymbol{\gamma}}_{k+1}^k)^{-1}\big]_{\mathrm{vec}} \\ \breve{\mathbf{b}}_{\mathrm{a},k+1} - \breve{\mathbf{b}}_{\mathrm{a},k} \\ \breve{\mathbf{b}}_{\mathrm{g},k+1} - \breve{\mathbf{b}}_{\mathrm{g},k} \end{bmatrix}$$

being the preintegrated measurement residual at keyframe k. $\hat{\mathbf{z}}_{k+1}^k = [\,\hat{\boldsymbol{\alpha}}_{k+1}^{k\top}, \hat{\boldsymbol{\beta}}_{k+1}^{k\top}, \hat{\boldsymbol{\gamma}}_{k+1}^{k\top}\,]^\top$ is the pre-integrated IMU measurements incorporating gyroscope and accelerometer readings from keyframe k to $k+1$. $\Delta\tau_k$ denotes the time interval between consecutive keyframes k and $k+1$. We use the operator $[\,\cdot\,]_{\mathrm{vec}}$ to take out the vector part of a quaternion. Due to space constraints, we do not provide the derivation of IMU pre-integration and the corresponding noise covariance $\mathbf{C}_{k+1}^k$ for the Mahalanobis norm above. A dedicated introduction can be found in [22].

The nonlinear least square problem in (2) can be solved using typical solvers, e.g., the trust region method. Constrained by the optimized keyframes, regular LiDAR frames in between are obtained via local graph optimization. After the fusion window slides over, the newly obtained LiDAR poses are inserted into a global pose graph with only keyframe feature maps maintained for mapping purpose. To detect potential loop closures, we search in the global graph, e.g., in a radius of 10 m, to find keyframe nodes that are spatially close but with enough temporal distance (e.g., 20 keyframes). An ICP is performed between the current feature scan and the candidate feature map, from which a fitting score is computed for loop closure detection. Once confirmed, a global pose graph optimization is invoked by imposing the constraint from the ICP. As depicted in Fig. 2, the keyframe local map $\breve{\mathbb{M}}^{w}$ at backend is then updated by corrected poses to further incorporate the LiDAR constraint.

V. Evaluation

A. Implementation and evaluation setup

We implement the proposed LiDAR-inertial odometry and mapping system in C++ using ROS [23]. The three modules shown in Fig. 2 are structured as three individual nodes. The nonlinear optimization problem in (2) is solved using the Ceres Solver [24]. We use GTSAM [25] to perform factor graph optimization for rectifying the global pose graph at loop closures. Our system is developed for Livox Horizon with the name LiLi-OM. It is, however, also applicable for conventional spinning LiDARs thanks to its generic backend fusion. Thus, two versions of the system are evaluated: (1) the original LiLi-OM for Livox Horizon with the proposed feature extraction approach and (2) its variant LiLi-OM* using the preprocessing module of [10] for spinning LiDARs. Evaluations are conducted based on public data sets (recorded using conventional LiDARs) and experiments (including data sets from Livox Horizon). All LiDAR frame rates are 10 Hz.

B. Public data set

We deploy LiLi-OM* to compare with competing state-of-the-art systems. These include works on (1) LiDAR odometry: A-LOAM[6] (open-source version of LOAM [10]), LeGO-LOAM [11] (shortened as LeGO) and (2) LiDAR-inertial odometry: LIO-mapping (shortened as LIOM) [15], LINS [16], LIO-SAM [17]. For evaluation, we use the EU long-term data set (*UTBM*) that provides two long urban navigation sequences recorded by a Velodyne HDL-32E and a six-axis IMU (100 Hz) [26]. LIO-SAM requires nine-axis IMU measurements. Thus, we include the *UrbanLoco* and *UrbanNav* data sets [5] recorded using a HDL-32E and Xsens MTi-10 IMU (nine-axis, 100 Hz). The RMSE of the absolute position error (APE) is computed for the final estimated trajectory based on the ground truth using the script in [27].

Shown in Tab. I[7], the proposed LiLi-OM* achieves the best tracking accuracy (bold) for all sequences in real time. LIO-SAM [17] requires nine-axis IMU readings for de-skewing and frontend odometry, thereby not applicable for *UTBM*.

[6]`https://github.com/HKUST-Aerial-Robotics/A-LOAM`

[7]Data sets abbr.: *UTBM-1*: UTBM-20180719, *UTBM-2*: UTBM-20180418-RA, *UL-1*: UrbanLoco-HK-20190426-1, *UL-2*: UrbanLoco-HK-20190426-2, *UN-1*: UrbanNav-HK-20190314, *UN-2*: UrbanNav-HK-20190428.

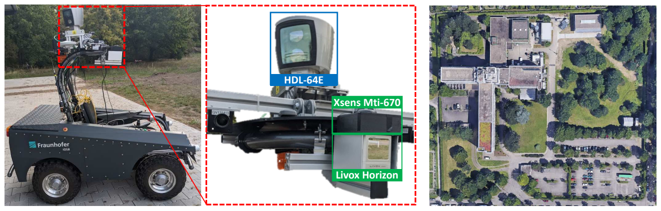


Fig. 7: Experimental setup for *FR-IOSB* data set.

For the remaining sequences, LIO-SAM still shows worse tracking accuracy than LiLi-OM though it additionally exploits orientation measurements from a magnetometer. This mainly results from the unified fusion scheme of LiLi-OM where LiDAR and inertial measurements are directly fused. LIOM fails on *UrbanNav* data sets (denoted as ✗) and shows large drift on *UTBM-1*. It also cannot run in real time with recommended configurations. LOAM delivers large tracking errors on *UTBM* as the implementation limits the iteration number in scan-matching for real-time performance.

TABLE I: APE (RMSE) in meters on public data sets

dataset	LOAM	LeGO	LIOM	LINS	LIO-SAM	LiLi-OM*
UTBM-1	479.51	17.12	468.75	16.90	–	**8.61**
UTBM-2	819.95	6.46	12.95	9.31	–	**6.45**
UL-1	2.39	2.22	2.53	2.27	2.54	**1.59**
UL-2	2.58	2.30	2.00	2.99	2.50	**1.20**
UN-1	11.20	2.70	✗	2.19	2.28	**1.08**
UN-2	12.70	4.15	✗	4.80	5.31	**3.24**

C. Experiment

To further test LiLi-OM in real-world scenarios, we set up a sensor suite composed of a Livox Horizon and an Xsens MTi-670 IMU. The total cost is about 1700 Euros (Q1, 2020), which is much less than conventional LiDAR-inertial setups.

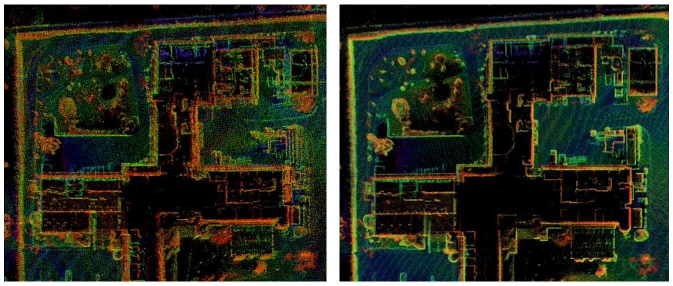
1) FR-IOSB data set: Shown in Fig. 7-(A), a mobile platform is instrumented with the proposed Livox-Xsens suite. For comparison with high-end mechanical spinning LiDARs, we set up a Velodyne HDL-64E onboard and synchronize it with an Xsens MTi-G-700 IMU (six-axis, 150 Hz). Three sequences were recorded at the Fraunhofer IOSB campus of Fig. 7-(B): (1) *Short* for a short path in structured scenes, (2) *Tree* recorded in bushes, and (3) *Long* for a long trajectory.

Both LiLi-OM and LiLi-OM* are tested. For comparison, we run LOAM, LeGO and Livox-Horizon-LOAM (shortened as LiHo)[8], a LOAM variant adapted to Livox Horizon with point clouds deskewed by IMU. Tab. II shows superior tracking accuracy (bold) of the proposed systems with our low-cost hardware setup performing equally well as the high-end one in the same scenario. We show reconstructed maps (partial) on sequence *Long* in Fig. 8, where LiLi-OM* delivers superior mapping quality using the proposed sensor fusion scheme.

TABLE II: End-to-end position error in meters on *FR-IOSB*

			Velodyne HDL-64E			Livox Horizon	
dataset	length	speed	LOAM	LeGO	LiLi-OM*	LiHo	LiLi-OM
Short	0.49 km	2.15 m/s	0.78	**0.25**	**0.34**	5.04	**0.25**
Tree	0.36 km	1.12 m/s	0.21	78.22	$< \mathbf{0.1}$	0.13	$< \mathbf{0.1}$
Long	1.10 km	1.71 m/s	0.43	0.82	$< \mathbf{0.1}$	3.91	**0.34**

[8]`https://github.com/Livox-SDK/livox_horizon_loam`



(A) LOAM (B) LiLi-OM*

Fig. 8: Mapping result comparison on sequence *Long*.

2) KA-Urban data set: Shown in Fig. 9-(A), the proposed Livox-Xsens sensor suite was further deployed onboard a backpack platform for large-scale test in urban scenarios. Five sequences were recorded in Karlsruhe, Germany with end-to-end locations registered from satellite images. Beside standard configuration of LiLi-OM, we deactivate its loop closure module to evaluate the odometry accuracy without global correction. LiHo is run for comparison.

Shown in Tab. III, LiLi-OM without loop closure (denoted as LiLi-OM-O) delivers much less drift than LiHo. When exploiting loop closure constraints, LiLi-OM shows very small end-to-end errors. In order to justify the benefits of multi-sensor fusion, we provide another special configuration of LiLi-OM with IMU constraints totally removed (including de-skewing). Evaluations are done on long-distance sequences (the last three ones) with loop closure both on and off. Denoted by the second entry after "/", this configuration is superior to LiHo (also pure LiDAR odometry), but inferior to the standard one using LiDAR-inertial fusion.

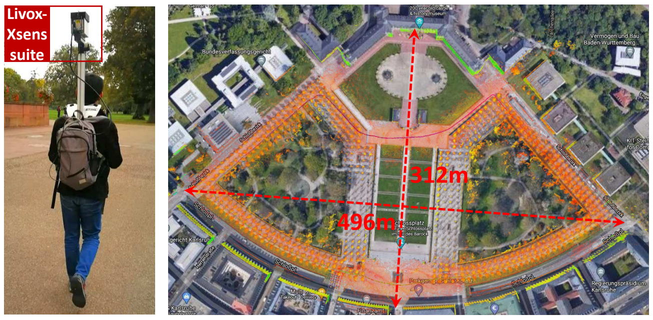
The mapping result of LiLi-OM on *Schloss-1* is given in Fig. 1, where the Schloss Karlsruhe is digitalized in high-precision point cloud using the proposed sensor suite. Result of running LiLi-OM on *Schloss-2* is visualized in Fig. 9-(B). An area of $496\,\mathrm{m} \times 312\,\mathrm{m}$ is mapped accurately with global consistency (compared to satellite image). Sequence *East* was recorded while cycling through eastern Karlsruhe for a long distance under fast and dynamic motion. Shown in Fig. 10, LiLi-OM delivers accurate odometry and mapping results using the proposed low-cost sensor suite.

TABLE III: End-to-end position error in meters on *KA-Urban*

dataset	length	speed	LiHo	LiLi-OM-O	LiLi-OM
Campus-1	0.50 km	1.43 m/s	1.47	1.11 / –	**0.13** / –
Campus-2	0.20 km	1.57 m/s	0.40	0.21 / –	**0.19** / –
Schloss-1	0.65 km	1.03 m/s	1.55	0.95 / 1.21	**0.15** / 0.24
Schloss-2	1.10 km	1.49 m/s	8.34	4.41 / 5.58	**0.08** / 0.12
East	3.70 km	3.11 m/s	109.62	15.66 / 19.28	**1.28** / 3.43

D. Runtime

All evaluations are done on a laptop (Intel Core i5-7300HQ CPU, 8GB RAM) with all four CPU cores involved. For all the data sets recorded with different devices in the evaluation, LiLi-OM delivers real-time performance (at LiDAR frame rate). The three nodes in Fig. 2 run in parallel and their average runtime per frame on representative sequences is collected in Tab. IV.



(A) (B)

Fig. 9: (A) Recording *KA-Urban* with proposed sensor suite. (B) Map from LiLi-OM on *Schloss-2* (path ends at origin).

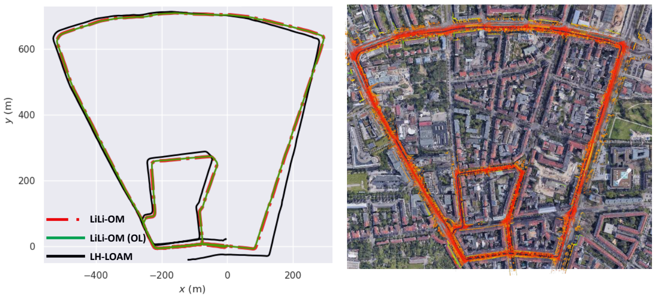


Fig. 10: Test results on sequence *East*.

Preprocessing for feature extraction and scan registration at frontend are light-weight. Runtime is dominated by backend fusion.

TABLE IV: Runtime of LiLi-OM per frame in ms

	Velodyne HDL			Livox Horizon		
node	*UTBM-2*	*UL-1*	*Long*	*Long*	*Schloss-2*	*East*
preprocessing	12.72	13.31	30.14	9.99	10.21	11.76
scan registration	22.29	23.71	16.71	22.69	27.15	25.30
backend fusion	50.27	57.62	60.92	58.86	54.56	41.81

Constructing the LiDAR constraints at backend fusion often takes substantial time. In Tab. V, we show statistics on feature extraction for LiLi-OM with typical configurations. Extracted features are counted per frame at preprocessing, while only associated features are counted at backend fusion per keyframe. Due to the relatively uniform FoV coverage of Livox Horizon, more plane features are extracted than edges on average. To guarantee runtime efficiency, features are always downsampled using a voxel grid filter before providing the LiDAR constraints.

TABLE V: Average feature numbers

	preprocessing / frame			backend fusion / keyframe	
dataset	raw points	edges	planes	edges	planes
Schloss-1	22579	207	12791	162	3688
Schloss-2	22230	633	13128	425	2649
East	21544	337	13109	247	2363

VI. Conclusion

In this work, we propose a novel sensor fusion method for real-time LiDAR-inertial odometry and mapping. A keyframe-based hierarchical scheme is established for directly fusing LiDAR and (pre-integrated) IMU measurements via sliding window optimization. Given the optimized keyframe states, regular-frame poses are obtained via factor graph optimization. The proposed LiDAR-inertial odometry and mapping system is universally applicable for both conventional LiDARs and solid-state LiDARs of small FoVs. For the latter use case, a novel feature extraction method is designed for the irregular and unique scan pattern of Livox Horizon, a newly released spinning free, solid-state LiDAR with much lower price than conventional 3D LiDARs. We conduct evaluations on both public data sets of conventional LiDARs and experiments using the Livox Horizon. Results show that the proposed system is real-time capable and delivers superior tracking and mapping accuracy over state-of-the-art LiDAR/LiDAR-inertial odometry systems. The proposed system, LiLi-OM, is featured as a cost-effective solution of high-performance LiDAR-inertial odometry and mapping using solid-state LiDAR.

There is still much potential to exploit for the proposed system. The deployed Livox-Xsens suite is lightweight and LiLi-OM is developed for universal egomotion estimation (not only for planar motion as [11]). Thus, it should be, for instance, tested onboard unmanned aerial vehicles in applications such as autonomous earth observation or environmental modeling coping with aggressive six-DoF egomotion. For large-scale odometry and mapping with limited computational resources, advanced map representations can be employed to improve memory as well as runtime efficiency. Potential options include volumetric mapping using TSDF (Truncated Signed Distance Fields) [3] or mapping with geometric primitives (especially in man-made environment [28]).

Acknowledgment

We would like to thank Thomas Emter from Fraunhofer IOSB for providing the robot platform and assistance in recording the *FR-IOSB* data set.

References

[1] J. Engel, J. Sturm, and D. Cremers, “Camera-Based Navigation of a Low-Cost Quadrocopter,” in *Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012)*, Vilamoura, Portugal, Oct. 2012, pp. 2815–2821.

[2] S. Bultmann, K. Li, and U. D. Hanebeck, “Stereo Visual SLAM Based on Unscented Dual Quaternion Filtering,” in *Proceedings of the 22nd International Conference on Information Fusion (Fusion 2019)*, Ottawa, Canada, July 2019, pp. 1–8.

[3] V. Reijgwart, A. Millane, H. Oleynikova, R. Siegwart, C. Cadena, and J. Nieto, “Voxgraph: Globally Consistent, Volumetric Mapping Using Signed Distance Function Submaps,” *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 227–234, 2020.

[4] O. Kähler, V. A. Prisacariu, J. P. C. Valentin, and D. W. Murray, “Hierarchical Voxel Block Hashing for Efficient Integration of Depth Images,” *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 192–197, 2016.

[5] W. Wen, Y. Zhou, G. Zhang, S. Fahandezh-Saadi, X. Bai, W. Zhan, M. Tomizuka, and L.-T. Hsu, “Urbanloco: A Full Sensor Suite Dataset for Mapping and Localization in Urban Scenes,” in *Proceedings of the 2020 International Conference on Robotics and Automation (ICRA 2020)*, Paris, France, May 2020, pp. 2310–2316.

[6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision Meets Robotics: The KITTI Dataset,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[7] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, “Comparing ICP Variants on Real-World Data Sets,” *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[8] A. Segal, D. Haehnel, and S. Thrun, “Generalized-ICP,” in *Proceedings of the 2009 Robotics: Science and Systems (RSS 2009)*, vol. 2, no. 4, Edinburgh, UK, June 2009.

[9] Y. Chen and G. Medioni, “Object Modelling by Registration of Multiple Range Images,” *Image and Vision Computing*, vol. 10, no. 3, pp. 145–155, 1992.

[10] J. Zhang and S. Singh, “LOAM: Lidar Odometry and Mapping in Real-Time,” in *Proceedings of the 2014 Robotics: Science and Systems (RSS 2014)*, vol. 2, no. 9, Berkeley, California, USA, July 2014.

[11] T. Shan and B. Englot, “LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain,” in *Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018)*, Madrid, Spain, Oct. 2018, pp. 4758–4765.

[12] I. Bogoslavskyi and C. Stachniss, “Fast Range Image-Based Segmentation of Sparse 3D Laser Scans for Online Operation,” in *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2016)*, Daejeon, Korea, Oct. 2016, pp. 163–169.

[13] J. Tang, Y. Chen, X. Niu, L. Wang, L. Chen, J. Liu, C. Shi, and J. Hyyppä, “LiDAR Scan Matching Aided Inertial Navigation System in GNSS-Denied Environments,” *Sensors*, vol. 15, no. 7, pp. 16 710–16 728, 2015.

[14] C. Le Gentil, T. Vidal-Calleja, and S. Huang, “IN2LAAMA: Inertial Lidar Localization Autocalibration and Mapping,” *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 275–290, 2021.

[15] H. Ye, Y. Chen, and M. Liu, “Tightly Coupled 3D Lidar Inertial Odometry and Mapping,” in *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA 2019)*, Montreal, Canada, May 2019, pp. 3144–3150.

[16] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, “LINS: A Lidar-Inertial State Estimator for Robust and Efficient Navigation,” in *Proceedings of the 2020 International Conference on Robotics and Automation (ICRA 2020)*, Paris, France, May 2020, pp. 8899–8906.

[17] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and R. Daniela, “LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping,” in *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2020)*, Las Vegas, Nevada, USA, Oct. 2020, pp. 5135–5142.

[18] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, “iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree,” *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[19] J. Lin and F. Zhang, “Loam Livox: A Fast, Robust, High-Precision LiDAR Odometry and Mapping Package for LiDARs of Small FoV,” in *Proceedings of the 2020 International Conference on Robotics and Automation (ICRA 2020)*, Paris, France, May 2020, pp. 3126–3131.

[20] B. Wu, A. Wan, X. Yue, and K. Keutzer, “SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud,” in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA 2018)*, Brisbane, Australia, May 2018, pp. 1887–1893.

[21] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, “Keyframe-Based Visual–Inertial Odometry Using Nonlinear Optimization,” *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[22] T. Qin, P. Li, and S. Shen, “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[23] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, “ROS: An Open-source Robot Operating System,” *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, pp. 5–10, 2009.

[24] S. Agarwal, K. Mierle, and Others, “Ceres Solver,” http://ceres-solver.org, 2016.

[25] F. Dellaert, “Factor Graphs and GTSAM: A Hands-on Introduction,” Georgia Institute of Technology, Atlanta, GA, USA, Tech. Rep. GT-RIM-CP&R-2012-002, Sep. 2012.

[26] Z. Yan, L. Sun, T. Krajnik, and Y. Ruichek, “EU Long-term Dataset with Multiple Sensors for Autonomous Driving,” in *Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2020)*, Las Vegas, Nevada, USA, Oct. 2020, pp. 10 697–10 704.

[27] M. Grupp, “evo: Python Package for the Evaluation of Odometry and SLAM,” https://github.com/MichaelGrupp/evo, 2017.

[28] H. Möls, K. Li, and U. D. Hanebeck, “Highly Parallelizable Plane Extraction for Organized Point Clouds Using Spherical Convex Hulls,” in *Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA 2020)*, Paris, France, May 2020, pp. 7920–7926.