2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

LA-LIO: Robust Localizability-Aware LiDAR-Inertial Odometry for Challenging Scenes

Junjie Huang[1], Yunzhou Zhang[1*], Qingdong Xu[1], Song Wu[1], Jun Liu[1], Guiyuan Wang[2], Wei Liu[2]

***Abstract*— Modern robotic systems are increasingly deployed in complex and diverse environments, and reliable localization under challenging conditions becomes crucial for the safe and efficient operation of these systems. The odometry based on LiDAR is prone to system collapse caused by computational divergence under conditions of aggressive motion and information deficiency in spatial geometry. To enhance the robustness of systems in challenging scenes, this work proposes LA-LIO, robust localizability-aware LiDAR inertial odometry. It mainly consists of three parts. Firstly, this paper presents a LiDAR degeneration detection method that enables stable degeneration assessment. Secondly, a method for segmenting LiDAR point clouds is proposed to alleviate the issue of excessive distortion in point clouds under aggressive motion scenes. The last is an Errors State Kalman Filter (ESKF) method with adaptive weights to utilize the existing spatial information as much as possible to improve the stability of the system in degenerated scenarios. The proposed method is evaluated and compared in multiple experiments, demonstrating the performance and reliability improvements of this approach in challenging environments.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) is a crucial technology for autonomous mobile robots to navigate in unknown environments [1]. Three-dimensional Light Detection and Ranging (LiDAR) sensors can directly acquire precise and dense scene structure information over a large range, making them one of the mainstream sensors in outdoor robotics and autonomous driving. Many LiDAR odometry systems [2]–[6] are built upon assumptions of slow motion and well-constrained spatial conditions. However, in real-life scenarios, there are often extreme conditions such as aggressive motion and insufficient constraints.

Many existing LiDAR inertial odometry systems [2]–[6] are based on a frame-by-frame approach similar to visual odometry [7], waiting for the LiDAR to complete a full rotation and then processing the resulting scan to update [8], [9] the self-motion. This is actually an artificial assumption [10], which would lead to a slow frequency of the LiDAR odometry system, which in turn would cause many problems. Since the LiDAR points are sampled sequentially at different times, this will introduce distortion in the point cloud, which are generally corrected using an Inertial Measurement Unit (IMU). And IMU is also generally used to predict the initial position during point cloud registration. Thus the accuracy of the motion recovery of the IMU is important, especially when the system is under violent motion, if the IMU cannot recover the motion between two frames of the point cloud well, the system is prone to crash. Therefore, we take advantage of the streaming sensor property of LiDAR to segment a frame of LiDAR points so that the output of the odometry can reach the frequency of the IMU, which improves the robustness of the odometry in aggressive motion scenarios.

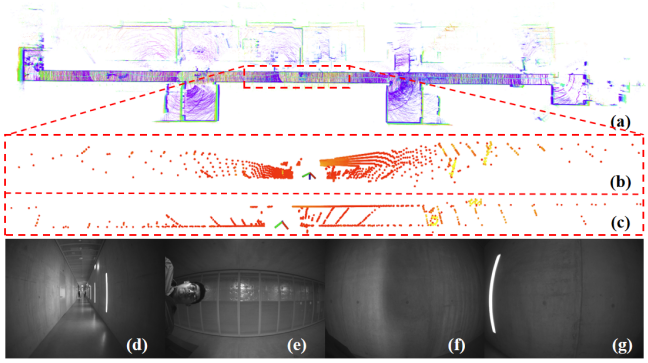

Fig. 1. To evaluate the performance of the LA-LIO system on the HILTI Exp07 Long Corridor sequence, where some frames are degenerate, but still able to construct high-precision point cloud maps. (a) presents the overall effect, (b) shows the oblique view of the degenerate single-frame point cloud, (c) displays the top-down view of the degenerate single-frame point cloud, and (d), (e), (f), and (g) are images from the front, top, left, and right perspectives, respectively, under the pose of the degenerate frame.

To achieve robustness, odometry require high quality state estimation in challenging scenarios. A common solution to this problem is to increase the redundancy of the sensing system [11]–[14], but it creates new problems such as synchronization issues and payload constraints [15]. Another approach is to analyze the effective constraints of the degenerate scenario [16] and make full use of this information to improve the robustness. In order to identify geometric degeneration in the environment, we need a reliable mathematical model to determine whether the point cloud at the current pose can consistently produce reliable localization information. For cases where information about the spatial geometry is lacking, effective constraints need to be fully utilized. Fig. I. shows the performance of our degeneration processing subsystem in the dataset. We first use the degeneration detector to determine the state of the system, and for the degenerate case we propose an adaptive weighting strategy to keep the system as stable as possible.

*The corresponding author of this paper
[1]Junjie Huang, Yunzhou Zhang, Qingdong Xu, Song Wu and Jun Liu are with College of Information Science and Engineering, Northeastern University, Shenyang 110819, China. zhangyunzhou@mail.neu.edu.cn
[2]Guiyuan Wang and Wei Liu are with Jiangsu Shuguang Optoelectronics Co., Ltd., Yangzhou, China. 1241139676@qq.com, xlkinhat@sina.com
This work was supported by National Natural Science Foundation of China (No. 61973066) and Major Science and Technology Projects of Liaoning Province(No. 2021JH1/10400049).

Our main contributions are as follows:

- We propose a LIO system that integrates two subsystems, a high-frequency odometry and a degradation-processing odometry, utilizing the distinct advantages of each to enhance the robustness.
- We propose a LiDAR point cloud frame degeneration detection method and adaptively cut between two subsystems to ensure the accuracy and robustness of the LIO system.
- We propose a point cloud segmentation method that increases the output frequency of the odometry and improves the robustness of the LIO system in strenuous motion scenarios.
- We propose an adaptive weight assignment method that emphasizes the importance of a small number of constraints and thus improves the robustness of the system to degeneration.

II. Related Works

A. LiDAR inertial Odometry

Many excellent LIO systems only discuss scenarios where their own motion is relatively slow, like FAST-LIO2 [13] downsamples raw LiDAR points, performs state prediction by IMU and compensates for point cloud distortion, then constructs a point-to-plane as an observation using the iKD-tree [17] and performs the state update [18] directly in the framework of an iterative error state Kalman filter. LIO-SAM [4] proposes a tightly coupled LiDAR inertial odometry method based on factor graph optimization [19] to obtain a globally consistent position of the robot by optimizing the inclusion of LiDAR odometry factors, IMU pre-integration factors, GPS factors and loop closure factors. All of the above methods require the use of an IMU to recover the motion during the point cloud accumulation period. There is a large amount of noise in the measurement of the IMU, the error brought by the longer integration will inevitably affect the accuracy of the odometry, and also limit the output frequency of the LiDAR odometry. This results in their inability to reliably localize in scenarios of overly aggressive movement.Continuous time methods [20] such as CLINS [21] are able to ignore motion compensation errors caused by aggressive motion through a non-rigid alignment [22] strategy. However, this requires more control points to fit drastically changing trajectories, which greatly affects the efficiency [23] of the system.

In order to reduce the cumulative error from long time integration of IMU, there are also a number of excellent works that use segmentation of LiDAR point cloud data to improve the robustness. LLOL [10] treats the rotating LiDAR as a stream sensor by dividing the LiDAR points. SR-LIO [24] reduces the IMU pre-integration error by dividing the complete LiDAR frame into three segments and then combining them to get a new frame. Both of these systems perform segmentation but still complete point cloud frames for position updating, and are unable to handle a greater degree of aggressive motion. The latest Point-LIO [25] proposes a point-by-point updating of the position, which fundamentally solves the point cloud distortion problem, but due to too few constraints for each update this method is sensitive to degenerate scenes.

B. Degeneration analysis and treatment

Regarding the localizability of degenerate scenes, coastal navigation [26] is one of the earlier attempts to model the information content in the environment. Censi [27] propose an information matrix approach to assess localizability by performing an eigenvalue decomposition of the information matrix to analyze the degree of constraints in each direction. Liu et al. [28] apply this approach to a 2D scene, aiming to maximize the determinant of the information matrix to estimate the localizability.

In terms of LiDAR degeneration detection, Zhang [16] propose a new degeneration factor to characterize the geometric degeneration by analyzing the optimization problem of state estimation, aiming at determining the direction of LiDAR degeneration and not updating the state in the direction of degeneration, but this method can only be applied to optimization-based state estimation problems. Weikun Zhen [15] propose a new method to determine whether a robot at its current position can consistently produce accurate localization results by performing a singular value decomposition of normal vectors. Subsequently, Zhen [29] introduce the concept of perturbation constraints for assessing LiDAR degeneration. However, the degeneration assessment thresholds computed by these methods lack a clear physical interpretation, making them less applicable across different datasets.

After identifying the degree of degeneration, there are various processing methods available. For instance, LVI-SAM [11] propose a loosely coupled system integrating visual and LiDAR systems. The system conducts degeneration detection on both visual and LiDAR systems, when one subsystem encounters issues, the subsystem automatically switches to the other to ensure stability. Weikun [29] suggests utilizing UWB sensors to provide additional effective localization information during LiDAR degeneration. Nowadays, many sensor fusion SLAM solutions [11]–[14] include degeneration assessment for individual sensors. However, these methods often only assess sensor degeneration and then discard the information from the degenerate sensor, leading to the loss of potentially useful information.

III. Method

The system pipeline is illustrated in Fig. 2, where LiDAR scans and IMU measurements are input together into the system. The system primarily consists of three main parts: The first part is the LiDAR degeneration Detector (see III-A), which is used to determine whether a single-frame LiDAR point cloud contains sufficient constraint information. If sufficient constraints are present, the system proceeds to the second part, the high-frequency odometry subsystem (see III-B). This section enhances robustness to aggressive motions

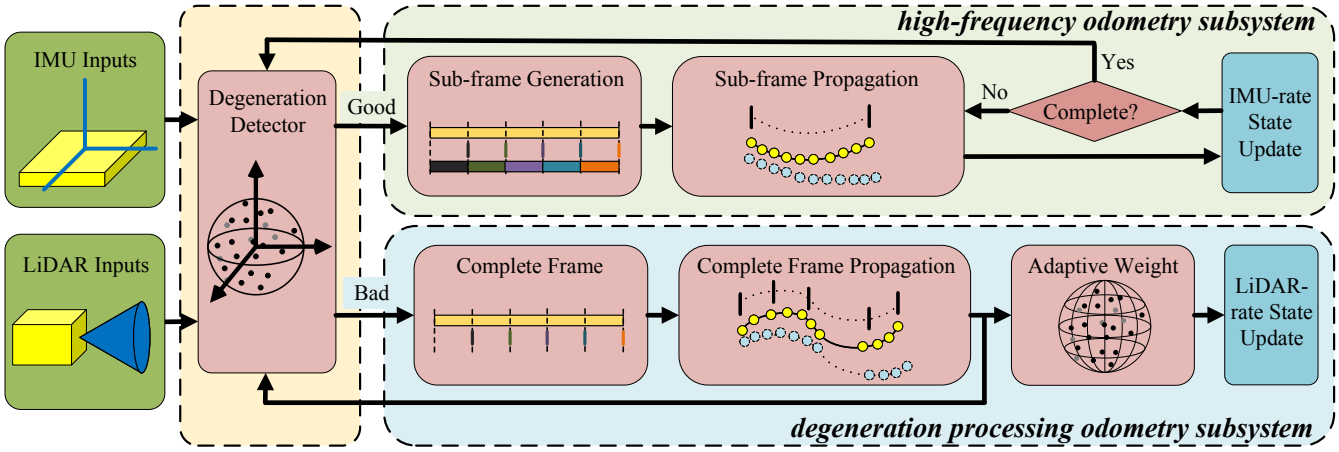

Fig. 2. System overview. It is divided into three parts, namely, the degeneration detector, the high-frequency odometry subsystem and the degeneration processing odometry subsystem.

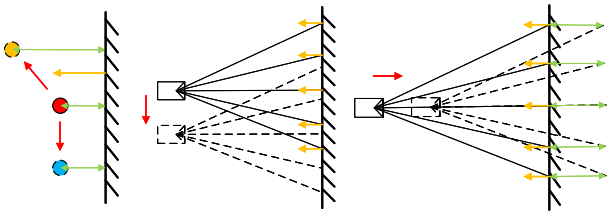

Fig. 3. The left side shows the change in the residuals resulting from a change in the position of a point; the center shows no change in the residuals for motions perpendicular to the normal vector, indicating degeneration in that direction; and the right side shows the largest change in the residuals for motions along the normal vector, indicating no degeneration in that direction. Where red arrows indicate the direction of motion, yellow arrows indicate the normal vector, and green bi-directional arrows indicate the residuals.

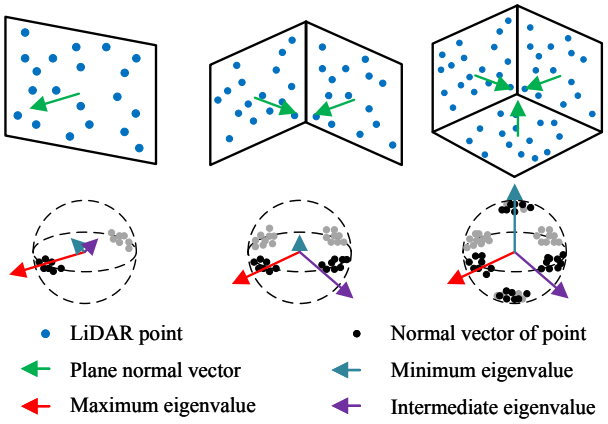

Fig. 4. Schematic diagram of degeneration analysis.

by segmenting the LiDAR point cloud frames. If constraint information is insufficient, the system enters the third part, the degeneration processing odometry subsystem (see III-C), which enhances robustness to states with poor constraint conditions through adaptive weighting. Finally, we introduce the same forward propagation (see III-D) and status update module (see III-E) of the two subsystems.

A. Degeneration Detector

We present a degeneration detection method designed to determine whether a complete frame of point cloud data contains enough information to produce accurate and reliable localization results.

First of all, whether we use the framing subsystem or the degeneration processing subsystem, we use the distance from the current frame point to the plane fitted to the local map as the residual for state updating. As shown in Fig. 3, if the robot is slightly perturbed in a certain direction and the residuals do not change much, the constraint is weak in that direction; otherwise, the constraint is strong. As can be seen from Equation (13), the residual change is largest for motion along the normal vector, while the residual change is smallest for motion perpendicular to the normal vector (the change is zero). This shows that the overall normal vector of a complete frame can reflect the constraints of the current frame.These constraints effectively limit the current position, and when there are fewer constraints, or even no constraints, in certain directions, the information in the point cloud for that frame is insufficient to estimate an accurate robot state.

In order to analyze whether the constraints of the current frame are sufficient, we normalize each normal vector of this frame, and then move the starting point of all the normal vectors under the origin of the world coordinate system to construct a constraint sphere matrix as in the following equation:

$$\boldsymbol{N}_{pre} = [\boldsymbol{n}_1, \boldsymbol{n}_2, \boldsymbol{n}_3, \cdots, \boldsymbol{n}_m] \tag{1}$$

where $\boldsymbol{N}_{pre}$ represents the matrix composed of normal vectors of the current frame, and $\boldsymbol{n}_i$ denotes the i-th normal vector, m denotes the number of normal vectors. We need to take the reciprocal of each normal vector to obtain a new normal vector. This operation not only facilitates subsequent analysis of degeneration directions but also aligns with actual constraint situations. For a plane, the residual is the distance from a point to the plane, which does not have a positive or negative sign. Therefore, a plane should have two normal vectors. Consequently, the resulting final constraint matrix $\boldsymbol{N}$ is as follows:

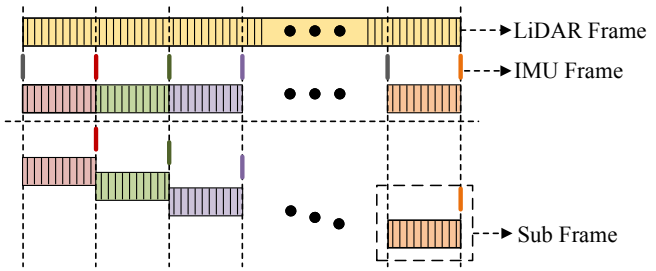

Fig. 5. The complete data frame is segmented into sub-frames, each of which consists of one IMU data and several LiDAR points.

$$\boldsymbol{N} = \left[\boldsymbol{N}_{pre}, -\boldsymbol{N}_{pre}\right] \tag{2}$$

This constraint sphere reflects all constraints of the current frame. The more uniform this sphere is, the better the constraints of this frame. We treat each normalized normal vector as a point, and we analyze these points using Principal Component Analysis (PCA), as shown in Fig. 4. Firstly, we calculate the covariance matrix of these points:

$$\boldsymbol{C} = \frac{1}{2m}\boldsymbol{N}\boldsymbol{N}^T \tag{3}$$

then the covariance matrix is decomposed into eigenvalues:

$$\boldsymbol{C} = \boldsymbol{Q}\begin{bmatrix}\lambda_1 & 0 & 0\\ 0 & \lambda_2 & 0\\ 0 & 0 & \lambda_3\end{bmatrix}\boldsymbol{Q}^T \tag{4}$$

where $\boldsymbol{Q}$ is a matrix consisting of eigenvectors. We obtain three eigenvalues $\{\lambda_1, \lambda_2, \lambda_3\}$, which are sorted. We define the degeneracy condition number as the following equation, with a smaller condition number indicates a worse state constraint:

$$k = \frac{\lambda_{min}}{\lambda_{max}} \tag{5}$$

We define two thresholds, threshold t_1 for entering the degeneration processing subsystem from the high-frequency odometry subsystem, and threshold t_2 for entering the high-frequency odometry subsystem from the degeneration processing subsystem. We consider that the degeneration occurs continuously, so we generally set t_1 slightly smaller than t_2, thus ensuring that the system does not repeatedly switch between the two subsystems frequently.

B. Sub-frame Generation

In this section, we introduce our sub-frame generation module. A complete data frame should include IMU frames for position prediction and require LiDAR points for pose update. Based on the continuous acquisition of LiDAR, we consider the original scan as a continuous stream of point cloud data, and segment a frame of scanning based on IMU data to obtain sub-frames for higher-frequency sub-scans. Fig. 5 illustrates the core idea of our segmented scan.

Due to the small number of segmented sub-frame point clouds, there may be insufficient sub-frame point cloud constraints resulting in degeneration. But this is irrelevant because after the degeneration detection of the complete frame, we can ensure that the point cloud constraints of this complete frame are sufficient. Although a few sub-frames are subject to degeneration, other sub-frames are immediately available to provide constraint information for the state quantities.

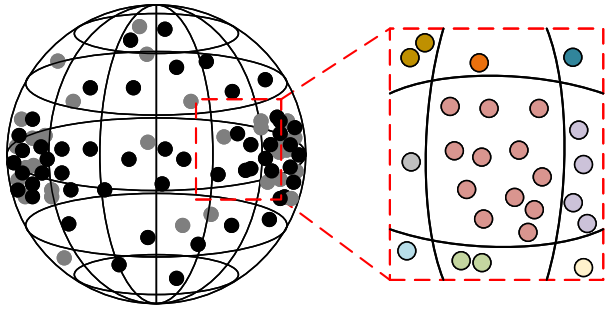

Fig. 6. On the left is the division of all normal vectors according to latitude and longitude curves, and on the right is an enlarged detail. The same color represents classification into the same surface patch.

To reduce redundant computations in the degeneration detector, we accumulate the corresponding normal vectors of the sub-frames after each state update, and when the accumulation reaches a complete frame we feed the normal vectors of that complete frame into the degeneration detector.

C. Adaptive Weight

We introduce our proposed adaptive weight assignment method that emphasizes the importance of a small number of dispersed constraints.

First of all, in the degenerate case, many normal vector constraints are oriented in one direction. For instance, if the current frame mostly illuminates a flat surface, having a small portion of features perpendicular to this plane would be highly effective constraints. Thus, we need to amplify the constraints of these features. To achieve this, we divide the constraint sphere into n surface patches based on latitude and longitude, as shown in Fig. 6, with the area of each patch serving as its weight value. The area s_i of each patch and the number $\boldsymbol{p}_i$ of normal vectors in each patch are as follows:

$$\begin{aligned}\boldsymbol{S} &= [s_1, s_2, s_3, \cdots, s_n]\\ \boldsymbol{P} &= [\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, \cdots, \boldsymbol{p}_n]\end{aligned} \tag{6}$$

The same weight w_i for each point within each surface patch is determined by the area of the surface patch and the number of points within the surface patch. As a result, some anomalously dispersed points will receive higher weights, while points that are clustered together will have lower weights. The weights w_i are computed as follows:

$$w_i = \frac{s_i}{size(\boldsymbol{p}_i)} \quad i \in (1 \sim n) \tag{7}$$

We use the ESKF method for state estimation, the observations are residuals of each point to the plane, and the covariance of the observations indicates the uncertainty of the residuals. The common approach nowadays is to fix the covariance of all points [6], [25], [30]. We adaptively allocate the covariance of each point based on the weight obtained

for each point, thus obtaining the final covariance matrix. Since the calculated weights are dimensionless, to apply the weights to the covariance, we use Equation (8) to scale our calculated weights and actual covariance to the same scale. We use the following formula to obtain the covariance of each point:

$$R_{j,j} = \left(\frac{a * 2m}{\sum_1^{2m} w_j} w_j \right)^{-1} \qquad j \in (1 \sim 2m) \tag{8}$$

where a denotes the inverse of the commonly used covariance per point, $2m$ denotes the number of constraints, and $R_{j,j}$ denotes the covariance per point. Since each point is independent of the others, the covariance between any two points is zero. As a result, the final covariance matrix $\boldsymbol{R}$ is a diagonal matrix with diagonal elements $R_{j,j}$. We use the ESKF method for state estimation, and it only requires replacing the original covariance matrix with the new covariance matrix when computing the kalman gain in Equation (14).

D. Propagation

Both subsystems employ the same state variables for forward propagation, and these include the nominal state $\boldsymbol{x}$, error state $\boldsymbol{\delta x}$, IMU input $\boldsymbol{u}$, and noise $\boldsymbol{w}$:

$$\begin{aligned}
&\mathcal{M} \triangleq \mathbb{R}^6 \times SO(3) \times \mathbb{R}^9 \\
&\boldsymbol{x} \triangleq \begin{bmatrix} {}^G\boldsymbol{p}_b^T & {}^G\boldsymbol{v}_b^T & {}^G\boldsymbol{q}_b^T & \boldsymbol{b}_a^T & \boldsymbol{b}_g^T & {}^G\boldsymbol{g}^T \end{bmatrix}^T \in \mathcal{M} \\
&\delta\boldsymbol{x} \triangleq \begin{bmatrix} {}^G\delta\boldsymbol{p}_b^T & {}^G\delta\boldsymbol{v}_b^T & {}^G\delta\boldsymbol{\theta}_b^T & \delta\boldsymbol{b}_a^T & \delta\boldsymbol{b}_g^T & {}^G\delta\boldsymbol{g}^T \end{bmatrix}^T \in \mathbb{R}^{18} \\
&\boldsymbol{u} \triangleq \begin{bmatrix} \boldsymbol{a}_m^T & \boldsymbol{\omega}_m^T \end{bmatrix}^T \\
&\boldsymbol{w} \triangleq \begin{bmatrix} \boldsymbol{n}_a^T & \boldsymbol{n}_\omega^T & \boldsymbol{n}_{ba}^T & \boldsymbol{n}_{b\omega}^T \end{bmatrix}^T
\end{aligned} \tag{9}$$

where the IMU measurements $\boldsymbol{a}_m$ and $\boldsymbol{\omega}_m$ serve as inputs to predict the nominal state values, the IMU position ${}^G\boldsymbol{p}_b$, velocity ${}^G\boldsymbol{v}_b$, and attitude ${}^G\boldsymbol{q}_b$ in the global coordinate system. Additionally, $\boldsymbol{b}_a$ and $\boldsymbol{b}_g$ represent the IMU biases, while ${}^G\boldsymbol{g}$ denotes gravity in the global coordinate system. Measurement noise for $\boldsymbol{a}_m$ and $\boldsymbol{\omega}_m$ is represented by $\boldsymbol{n}_a$ and $\boldsymbol{n}_\omega$, respectively, while the random walk noise of IMU biases is denoted by $\boldsymbol{n}_{ba}$ and $\boldsymbol{n}_{b\omega}$. These elements collectively contribute to the discrete-time propagation model used for predicting nominal state values:

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) \tag{10}$$

Utilizing a linearized discrete-time error state transfer Equation (11), we forward propagate the covariance matrix of the error state according to Equation (12).

$$\delta\boldsymbol{x}_{k+1} = (I + \boldsymbol{F}_k\Delta t)\delta\boldsymbol{x}_k + (\boldsymbol{C}_k\Delta t)\boldsymbol{w} \tag{11}$$

$$\boldsymbol{P}_{k+1} = (I+\boldsymbol{F}_k\Delta t)\boldsymbol{P}_k(I+\boldsymbol{F}_k\Delta t)^T + (\boldsymbol{C}_k\Delta t)\boldsymbol{Q}_k(\boldsymbol{C}_k\Delta t)^T \tag{12}$$

The detailed expressions of $f(\boldsymbol{x}_k, \boldsymbol{u}_k)$, $\boldsymbol{F}_k$, $\boldsymbol{C}_k$, and $\boldsymbol{Q}_k$ can be find in [31]. Furthermore, the outcomes of the forward propagation are utilized for motion compensation and registration.

E. Update

Since the normal vector is crucial for both degeneration detection and adaptive weighting, the direct method is adopted. This involves downsampling the sub-frame LiDAR points and searching for the 5 nearest neighbors for each point. Subsequently, point-to-plane residuals are computed after fitting the local plane patches.

$$h_j\left(\boldsymbol{x}_k^\kappa\right) = {}^G\boldsymbol{n}_j^T\left({}^G\boldsymbol{q}_{b_k}^\kappa\left({}^b\boldsymbol{q}_l\,{}^l\boldsymbol{p}_j + {}^b\boldsymbol{p}_l\right) + {}^G\boldsymbol{p}_{b_k}^\kappa - {}^G\boldsymbol{c}_j\right) \tag{13}$$

Where $h_j\left(\boldsymbol{x}_k^\kappa\right)$ represents the j-th row of the function $h\left(\boldsymbol{x}_k^\kappa\right)$. ${}^b\boldsymbol{q}_l$ and ${}^b\boldsymbol{p}_l$ denote the pre-calibrated extrinsics between the LiDAR and IMU. ${}^G\boldsymbol{n}_j$ and ${}^G\boldsymbol{c}_j$ stand for the normal vector and centroid of the j-th plane patch, respectively. ${}^l\boldsymbol{p}_j$ represents the corresponding j-th LiDAR point. The symbol κ indicates the κ-th iteration. These normal vectors are stored for degeneration detection purposes and, if needed, utilized for calculating adaptive weights.

If it is the high-frequency odometry subsystem, we follow the approach of FAST-LIO2 [6] for calculating the kalman gain. However, if it is the degeneration processing subsystem, we first utilize the adaptive weighting method to compute a new observation covariance matrix $\boldsymbol{R}$. Subsequently, we employ the $\boldsymbol{R}$ matrix for calculating the kalman gain. Then, we iteratively execute the error state update operation:

$$\boldsymbol{K} = (\boldsymbol{H}^T\boldsymbol{R}^{-1}\boldsymbol{H} + \boldsymbol{P}_k^{-1})^{-1}\boldsymbol{H}^T\boldsymbol{R}^{-1} \tag{14}$$

$$\boldsymbol{x}_k^{\kappa+1} = \boldsymbol{x}_k^\kappa + \boldsymbol{K}\left(\boldsymbol{0} - h(\boldsymbol{x}_k^\kappa)\right) - (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})\left(\boldsymbol{x}_k^\kappa - \boldsymbol{x}_k^0\right) \tag{15}$$

Throughout the aforementioned iterations, we have concurrently conduct corrections to the nominal state and resets of the error state. Upon convergence of the iterations, the state covariance is updated using the following equation:

$$\boldsymbol{P}_k = (\boldsymbol{I} - \boldsymbol{K}\boldsymbol{H})\,\boldsymbol{P}_k \tag{16}$$

IV. Experience

The main system performance evaluations here are performed using the M2DGR [32], the Newer College dataset [33], the HILTI Challenge Dataset [34], the Laser Degradation Challenge dataset contributed by the FAST-LIVO algorithm [13], and our own recorded datasets, of which the first three are rotating LiDAR, and the last two are Livox LiDAR. We compare our algorithm with state-of-the-art LIO systems: FAST-LIO2, Faster-LIO [30], LIO-SAM, and Point-LIO.

In all experiments, FAST-LIO2 disables online external parameter estimation, and LIO-SAM disables loop closure optimization. For the other systems, we use default parameters to provide the best results for each algorithm. All experiments are conducted on an Intel i7-11700H CPU, using the Robot Operating System (ROS) in Ubuntu 20.04, with 32GB of RAM.

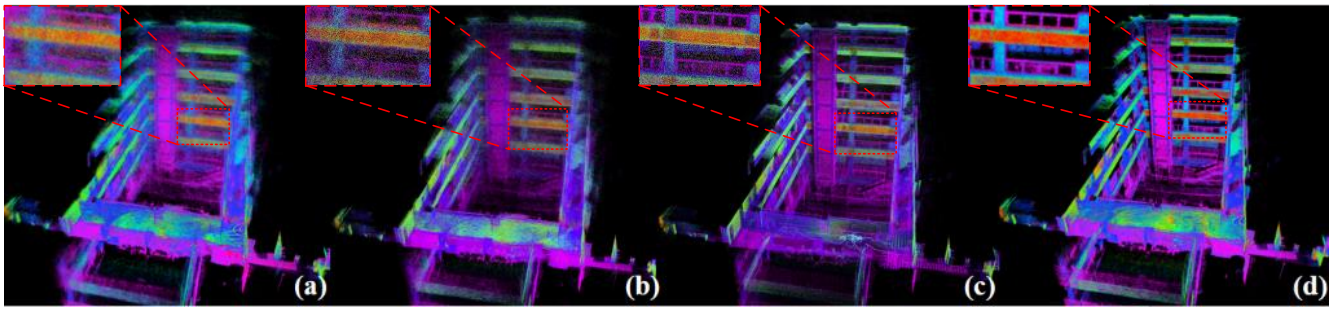

Fig. 7. Point cloud maps constructed by the different systems in our recorded strenuous motion dataset

TABLE I
ABSOLUTE POSE ERROR (RMSE, METERS)

Seq ID	LIO-SAM [4]	Point-LIO [25]	Faster-LIO [30]	FAST-LIO2 [6]	Ours	Length (m)
street_01	0.604	0.491	0.341	0.373	0.342	752.04
street_02	3.097	2.986	2.915	2.826	2.806	1484.62
street_03	0.143	0.130	0.136	0.138	0.134	423.91
street_04	1.242	0.561	0.652	0.512	0.517	840.43
ncd_1	0.127	0.131	0.112	0.141	0.116	479.04
ncd_2	0.121	0.098	0.089	0.093	0.082	97.20
ncd_3	0.173	0.147	0.160	0.158	0.136	695.66

A. System Accuracy

To validate the proposed LIO's accuracy on conventional datasets, Table I shows the Absolute Pose Error (APE), where red font indicates the highest precision and blue font indicates the second. We evaluate the localization accuracy using a total of 4 street sequences from the M2DGR dataset and 3 sequences from the Newer College dataset. In the experiments below, we apply 4-point skipping and 0.5m voxel filtering to filter the point clouds for FAST-LIO2, Faster-LIO, Point-LIO, and our system. In this general scenarios due to good constraints, our high-frequency odometry subsystem plays a major role and the cumulative error of the multi-frame integration of the IMU is relatively small, which results in not being able to show our advantages. But as can be seen from Table I, in this general scenario our LA-LIO system is still comparable to the existing state-of-the-art LIO systems in terms of accuracy. Where ncd_1 denotes the quad_with_dynamics sequence in the Newer College dataset, ncd_2 denotes the dynamic_spinning sequence in the Newer College dataset, and ncd_3 denotes the parkland_mound sequence in the Newer College dataset.

B. Aggressive Motion Test

We utilize the dynamic-spinning sequence from the Newer College dataset to evaluate the robustness of our system under aggressive motion. In this sequence, the sensor system is manually flipped by the operator, reaching maximum angular velocities of up to 183 degrees per second. In this kind of strenuous movement scenario, the multi-frame IMU integration will bring a large cumulative error, our LA-LIO system through the sub-frame generation module

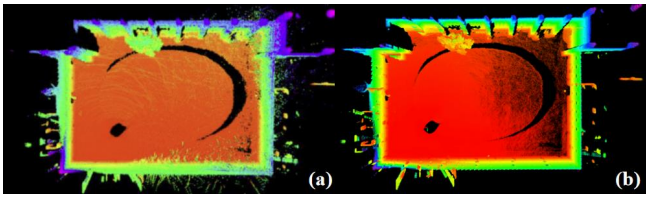

Fig. 8. Visualization of point cloud maps constructed from the dynamic spinning sequence of the Newer College dataset, (a) point cloud map constructed for the FAST-LIO2 system, (b) point cloud map constructed for our LA-LIO system.

allows the system to carry out an observation correction immediately after each frame of IMU integration, which greatly reduces the cumulative error of multi-frame IMU integration. Despite the ground truth remaining within a small range in the dynamic-spinning sequence, our LA-LIO system still demonstrates slight improvements in accuracy. Visualizations from Fig. 8 depict that our LA-LIO method exhibits higher accuracy in such intense motion scenarios, resulting in clearer and more complete point cloud reconstructions.

We also validate the robustness of different systems under intense motion scenarios using our own recorded dataset. We use the Livox-AVIA LiDAR while facing another wall in the corridor to ensure the point cloud does not degrade, we quickly wiggle the LiDAR to record data. In Fig. 7, it can be observed that the point cloud maps constructed by FAST-LIO2 and Faster-LIO in our recorded data appear blurry. Both Point-LIO and our LA-LIO system can construct highly accurate point cloud maps since they are both frame-splitting systems, which can significantly reduce the error of IMU pre-integration.

C. Degenerate Scene Test

We use the LiDAR challenging dataset from the FAST-LIVO algorithm and our own recorded dataset with degeneration scenes to validate the robustness of our LA-LIO system in scenarios with both degeneration and intense motion. Fig. 9 shows the LiDAR degeneration challenge dataset contributed by the FAST-LIVO algorithm. This data shows the Livox LiDAR illuminating one wall, the floor as well as the ceiling of the hall, and the system is consistently under-constrained in the left and right directions. FAST-LIO2, Faster-LIO, and Point-LIO algorithms show significant

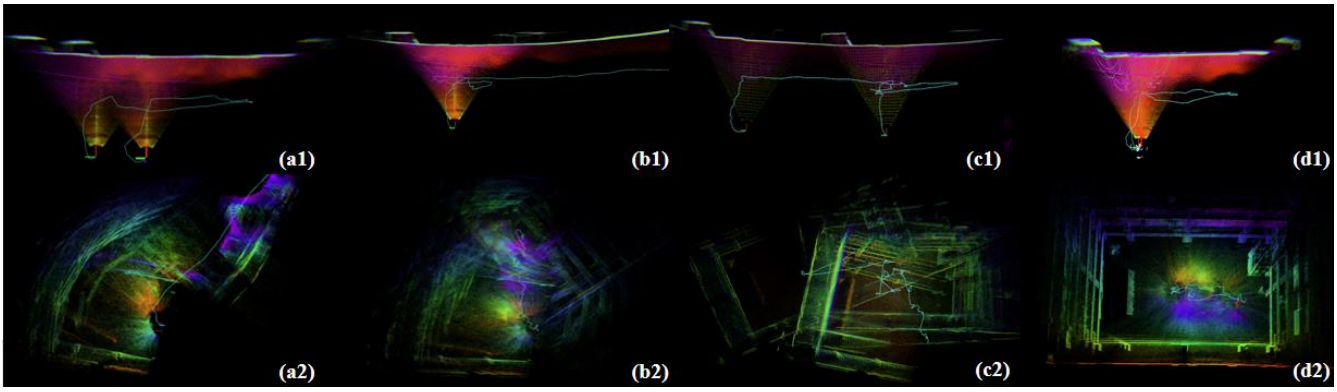

Fig. 9. The top section displays the results of the LiDAR degeneration challenge dataset contributed by the FAST-LIVO algorithm, while the bottom section shows our self-recorded dataset. Here, (a), (b), (c), and (d) respectively depict the point cloud maps constructed by FAST-LIO2, Faster-LIO, Point-LIO, and LA-LIO.

drifts in the degraded state of the LiDAR, which results in a final end-to-end position deviation is large. Our adaptive weighting module, however, by emphasizing the importance of a small number of constraints in the degenerate direction, for this degenerate case, can alleviate the problem of the accumulation and propagation of the positioning system's error in that degenerate direction. In the end, our LA-LIO system can still maintain the stability of the system, construct a point cloud map that conforms to the real situation, and return to the initial position.

We also conduct tests on the HILTI Challenge Dataset 2022 sequence Exp07 Long Corridor, as shown in Fig. I. This dataset is recorded in a long corridor, with multiple segments of LiDAR data in degenerate states. However, our LA-LIO system can detect degeneration and construct clear, high-precision maps.

To validate the robustness of our system in scenarios with both intense motion and degenerate conditions, we used our own dataset. We rapidly move the Livox-AVIA LiDAR in a spacious hall, and there are several segments where all LiDAR points are projected onto the ground. In such extreme scenarios, the FAST-LIO2 system fail to maintain stable localization, resulting in a complete loss of position estimation. While Faster-LIO do not completely lose localization, the appearance of degenerate scenes cause drifting in the localization, ultimately leading to distorted point cloud maps. Point-LIO is sensitive to degeneration, resulting in the construction of multiple point cloud maps. However, our LA-LIO system, by switching adaptively between the high-frequency odometry subsystem and the degeneration processing subsystem, can perform robust localization in such extreme scenarios where both violent motion and degradation occur, and ultimately reconstruct a high-precision point cloud map, as shown in Fig. 9.

D. Efficiency Test

In this section, we analyze the time consumption of the LA-LIO system using the M2DGR dataset. The LiDAR model in this dataset is the LiDAR Velodyne VLP-32C with a frequency of 10Hz, and the IMU model is the Handsfree A9, 9-axis with a frequency of 150Hz. Each frame contains approximately 10302 valid points. The additional degeneration detection module and adaptive weight computation impose minimal computational burden, allowing the overall system to achieve real-time performance.

TABLE II
TIME CONSUMPTION FOR EACH MODULE OF OUR SYSTEM (MS)

Steps	LA-LIO$_1$	LA-LIO$_2$	FAST-LIO2 [6]
Degeneration detection	0.11	0.11	-
Forward propagation + backward propagation	3.90	4.85	4.73
Adaptive weight	-	1.02	-
State update	29.38	27.97	23.41
Total runtime	34.16	34.33	28.46

The system's time analysis is presented in Table II. For the degeneration detection module, since we use the previously recorded normal vectors for degeneration assessment calculations without the need to recompute the point-to-plane matching, the average time consumption for this module is 0.11ms, which can be practically neglected. LA-LIO$_1$ represents the running time when it is always in the high-frequency odometry subsystem, where the LiDAR data is divided into 15 frames according to the IMU frequency, the forward and backward propagation time and state update time accumulate over the 15 frames, and the extra consumptions are mainly in the state updating for many times compared to the FAST-LIO2. LA-LIO$_2$ represents the running time when it is always in the degradation processing subsystem. The adaptive weight computation is performed only when matching points to planes, and on average, two computations are required for each state update. Therefore, the computation time per update is 0.512ms. Compared with the FAST-LIO2 system, the main time consumption is that it is the computation of the weights and not able to simplify the computation of the state update because the covariance of each point is not the same. All in all, the overall system takes very little time and is very fast.

V. Conclusion

In this paper, a robust LIO system in challenging environment is proposed, which can stably locate in aggressive moving scenes and degenerate scenes, and at the same time ensure the positioning accuracy in general scenes. For the localizability analysis of a frame of point cloud, we can detect the localizability of a frame of point cloud stably by PCA of the normal vector of point surface residual. For the aggressive motion scene, we effectively reduce the IMU integration error by dividing the LiDAR point cloud frame. For degenerate scenes, we adaptively assign weights to each point through the analysis of constraints. This method can emphasize the importance of effective constraints, so as to achieve stable positioning by grasping a small number of effective constraints in scenes with lack of spatial structure. Finally, it is tested in various extreme scenarios, which proves that the robustness of our method is better than the existing advanced odometry. In the future, we will explore how to make full use of effective constraints to achieve more robust positioning in degenerate scenes.

References

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[3] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4758–4765.

[4] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2020, pp. 5135–5142.

[5] W. Xu and F. Zhang, "Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.

[6] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[7] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[8] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. Spie, 1992, pp. 586–606.

[9] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." in *Robotics: science and systems*, vol. 2, no. 4. Seattle, WA, 2009, p. 435.

[10] C. Qu, S. S. Shivakumar, W. Liu, and C. J. Taylor, "Llol: Low-latency odometry for spinning lidars," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 4149–4155.

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

[12] J. Lin and F. Zhang, "R 3 live: A robust, real-time, rgb-colored, lidar-inertial-visual tightly-coupled state estimation and mapping package," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10 672–10 678.

[13] C. Zheng, Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang, "Fast-livo: Fast and tightly-coupled sparse-direct lidar-inertial-visual odometry," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4003–4009.

[14] J. Lin, C. Zheng, W. Xu, and F. Zhang, "R ̂2 live: A robust, real-time, lidar-inertial-visual tightly-coupled state estimator and mapping," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7469–7476, 2021.

[15] W. Zhen, S. Zeng, and S. Soberer, "Robust localization and localizability estimation with a rotating laser scanner," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 6240–6245.

[16] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 809–816.

[17] Y. Cai, W. Xu, and F. Zhang, "ikd-tree: An incremental kd tree for robotic applications," *arXiv preprint arXiv:2102.10808*, 2021.

[18] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[19] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," *Georgia Institute of Technology, Tech. Rep*, vol. 2, p. 4, 2012.

[20] C. Park, P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan, "Elastic lidar fusion: Dense map-centric continuous-time slam," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1206–1213.

[21] J. Lv, K. Hu, J. Xu, Y. Liu, X. Ma, and X. Zuo, "Clins: Continuous-time trajectory estimation for lidar-inertial system," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 6657–6663.

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

[23] K. Chen, R. Nemiroff, and B. T. Lopez, "Direct lidar-inertial odometry: Lightweight lio with continuous-time motion correction," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3983–3989.

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

[25] D. He, W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang, "Point-lio: Robust high-bandwidth light detection and ranging inertial odometry," *Advanced Intelligent Systems*, p. 2200459, 2023.

[26] N. Roy, W. Burgard, D. Fox, and S. Thrun, "Coastal navigation-mobile robot navigation with uncertainty in dynamic environments," in *Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C)*, vol. 1. IEEE, 1999, pp. 35–40.

[27] A. Censi, "On achievable accuracy for range-finder localization," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 4170–4175.

[28] Z. Liu, W. Chen, Y. Wang, and J. Wang, "Localizability estimation for mobile robots based on probabilistic grid map and its applications to localization," in *2012 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2012, pp. 46–51.

[29] W. Zhen and S. Scherer, "Estimating the localizability in tunnel-like environments using lidar and uwb," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4903–4908.

[30] C. Bai, T. Xiao, Y. Chen, H. Wang, F. Zhang, and X. Gao, "Faster-lio: Lightweight tightly coupled lidar-inertial odometry using parallel sparse incremental voxels," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4861–4868, 2022.

[31] J. Sola, "Quaternion kinematics for the error-state kalman filter," *arXiv preprint arXiv:1711.02508*, 2017.

[32] J. Yin, A. Li, T. Li, W. Yu, and D. Zou, "M2dgr: A multi-sensor and multi-scenario slam dataset for ground robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2266–2273, 2021.

[33] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The newer college dataset: Handheld lidar, inertial and vision with ground truth," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4353–4360.

[34] L. Zhang, M. Helmberger, L. F. T. Fu, D. Wisth, M. Camurri, D. Scaramuzza, and M. Fallon, "Hilti-oxford dataset: A millimeter-accurate benchmark for simultaneous localization and mapping," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 408–415, 2022.