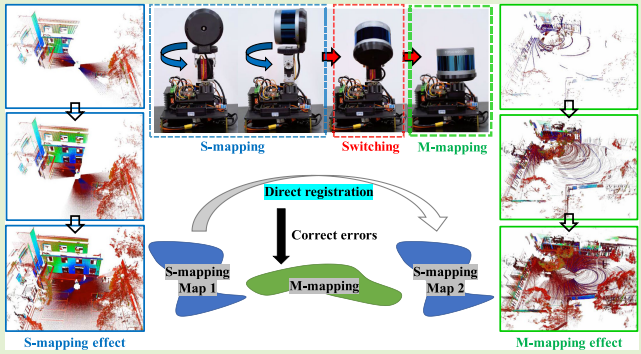
MS-Mapper: A LiDAR Mapping System by Switching Between Mobile and Stationary Configurations

Zeyu Wan, Changjian Jiang, Xiaolong Wang, Ruilan Gao, and Yu Zhang

***Abstract*—This article introduces the design and implementation of MS-Mapper, a LiDAR mapping system incorporating a mechanical structure based on a step motor and steering gear. The hardware configuration of MS-Mapper allows for adjustable vertical and horizontal orientations of the LiDAR sensor. The MS-Mapper has two configurations during the mapping process: stationary mapping (S-mapping), which is dedicated to generating detailed point cloud maps and providing pose drift correction, mobile mapping (M-mapping), which is utilized for completing map cavities and ensuring comprehensive coverage mapping. This dual-configuration approach employed by MS-Mapper significantly enhances pose estimation accuracy, particularly in challenging environments such as lengthy corridors. To further improve pose estimation accuracy, we propose a residual selection-based bundle adjustment (BA) algorithm. Additionally, we propose a dynamic Gaussian mixture model (GMM) tree map representation, which is designed to accommodate dynamic growth. This representation effectively reduces the number of redundant Gaussian models and minimizes data association costs compared to utilizing a dense point cloud. Furthermore, a configuration selection algorithm is proposed to detect pose estimation degeneration and determine optimal opportunities for switching configurations. In summary, the MS-Mapper system amalgamates the strengths of the traditional terrestrial laser scanner (TLS) and LiDAR simultaneous localization and mapping (SLAM) approaches. MS-Mapper provides a robust solution for achieving accurate mapping results.**



***Index Terms*— LiDAR mapping, pose estimation, robotics.**

I. Introduction

LiDAR mapping is commonly approached through two methods: the traditional terrestrial laser scanner (TLS) and LiDAR simultaneous localization and mapping (SLAM). The TLS yields accurate results. However, it relies on manual labor for transportation, which makes TLS inefficient for capturing extensive areas rapidly. Conversely, LiDAR SLAM is an automated process; however, its LiDAR odometry (LO) experiences gradual drift over time, leading to map distortions and estimation failures [1].

In response to industry demands for accurate and convenient mapping, especially in challenging environments [2], such as lengthy corridors, we designed the MS-Mapper, we have amalgamated the strengths of TLS and LiDAR SLAM to devise a novel LiDAR sensor. It is an adaptable LiDAR mapping system depicted in Fig. 1. Its hardware is a step motor and steering gear-based mechanical structure.

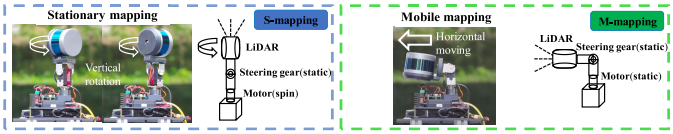


Fig. 1. Hardware design of MS-Mapper has two configurations: S-mapping and M-mapping. S-mapping requires the LiDAR to be positioned vertically and rotate slowly around its center using a motor. The base remains static. M-mapping requires the LiDAR to be positioned horizontally through a steering gear.

Manuscript received 9 December 2023; revised 3 January 2024; accepted 6 January 2024. Date of publication 17 January 2024; date of current version 29 February 2024. This work was supported in part by STI 2030-Major Projects under Grant 2021ZD0201403; in part by NSFC through Autonomous Intelligent Unmanned Systems under Grant 62088101; and in part by the Open Research Project of the State Key Laboratory of Industrial Control Technology, Zhejiang University, China, under Grant ICT2022B04. The associate editor coordinating the review of this article and approving it for publication was Prof. Chang-Hee Won. *(Zeyu Wan and Yu Zhang contributed equally to this work.) (Corresponding author: Yu Zhang.)*

The authors are with the State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Zhejiang University, Hangzhou 310007, China (e-mail: zeyuwan@zju.edu.cn; changjianjiang@zju.edu.cn; xlking@zju.edu.cn; gaoruilan@zju.edu.cn; zhangyu80@zju.edu.cn).

Digital Object Identifier 10.1109/JSEN.2024.3352044

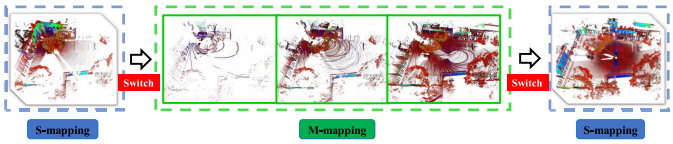


Fig. 2. S-mapping primarily focuses on pose drift correction. M-mapping allows for a wide field of view similar to LiDAR SLAM.

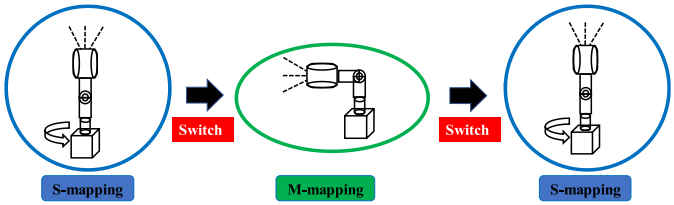


Fig. 3. MS-Mapper is first put at a fixed location and starts a round S-mapping. Then switches to M-mapping for motion. When the configuration selection module suggests a signal, MS-Mapper stops moving, and switches back to S-mapping.

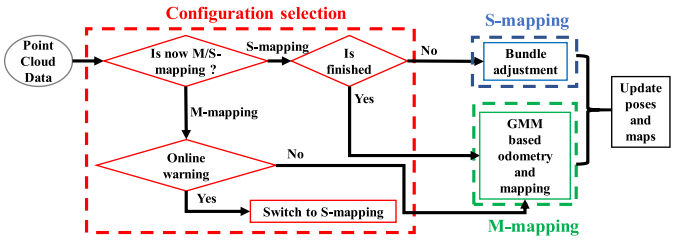


Fig. 4. PCD is initially fed into the configuration selection module, which determines whether MS-Mapper should be in the M-mapping or S-mapping mode. Once S-mapping local maps are generated by BA, M-mapping uses scan-to-map registration by GMM to estimate poses.

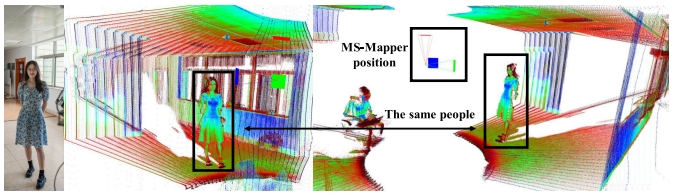


Fig. 5. MS-Mapper in S-mapping can make the LiDAR spin a circle path. Then, it can build a very dense and accurate point cloud map.

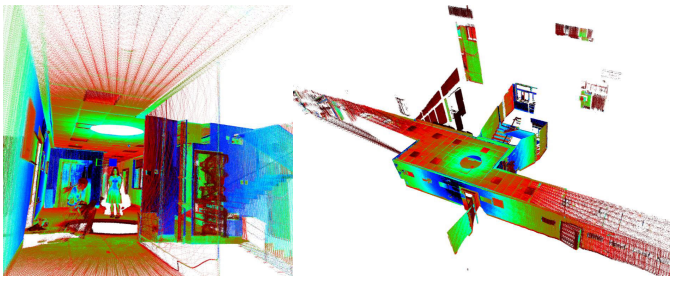


Fig. 6. S-mapping builds a room, whose details are very distinct.

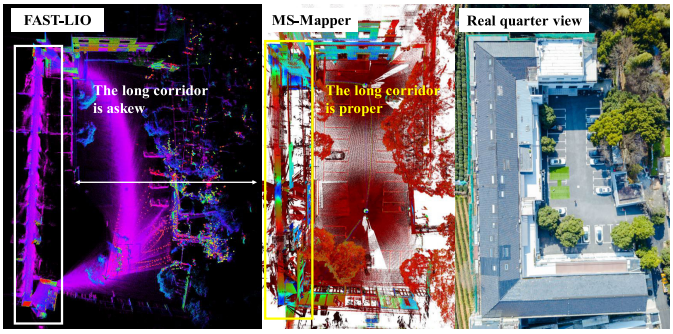


Fig. 7. Compared to the real quarter view, the SLAM method has some drifts in the long corridor. MS-Mapper estimates pose well.

The hardware design of MS-Mapper encompasses two configurations: stationary mapping (S-mapping) and mobile mapping (M-mapping). In the S-mapping configuration, the LiDAR is vertically positioned and rotates gradually around its center via a motor, while the base remains stationary. The M-mapping configuration necessitates LiDAR horizontal positioning by a steering gear, with all components static except for the mobile base. As depicted in Fig. 2, S-mapping primarily addresses pose drift correction, while M-mapping provides a broad field of view akin to LiDAR SLAM, ensuring comprehensive coverage mapping.

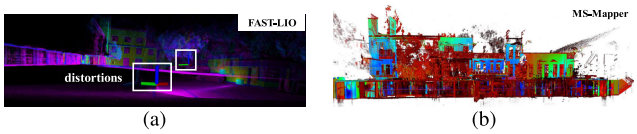
The MS-Mapper operational process is shown in Fig. 3. The workflow initiates with MS-Mapper stopped at a fixed location, starting a round of S-mapping. When this S-mapping is finished, MS-Mapper switches to M-mapping for motion. Upon receiving a signal from the configuration selection module, MS-Mapper stops its movement and reverts to S-mapping again.

The core reason that MS-Mapper is better than traditional SLAM methods is S-mapping. It employs S-mapping to generate accurate local maps. These local maps can be directly aligned, effectively mitigating drift errors during the motion process. Consequently, compared to the traditional SLAM methods, MS-Mapper exhibits significantly lower drift in mapping.

The proposed mapping algorithm pipeline is illustrated in Fig. 4. The configuration selection module determines the switching between S-mapping and M-mapping. The configuration selection module incorporates an online pose estimation degeneration warning algorithm. It considers the impact of different point residuals on pose estimation. It determines the opportune moment for a configuration change. It decouples the current residuals into six dimensions: three for rotation and three for translation. It judges whether the current residual constraints are enough to estimate the pose. If not, then it suggests a signal to set a S-mapping on the user interface (UI).

During the S-mapping phase, a residual selection-based bundle adjustment (BA) method is employed to register all point clouds. MS-Mapper in S-mapping follows a spinning path, facilitating the creation of a dense and precise point cloud map, as illustrated in Figs. 5 and 6. Conversely, in the M-mapping phase, a dynamic Gaussian mixture model (GMM) map representation is adopted, converting the dense point cloud into Gaussian models to enhance computational efficiency. Additionally, it can dynamically adjust and reduce the number of redundant Gaussian models.

An experiment illustrating the effectiveness of MS-Mapper is presented in Fig. 7. A visual comparison between Fig. 8(a) and (b) highlights distortions in the SLAM method FAST-LIO, which results in a misalignment during the long corridor. Conversely, MS-Mapper demonstrates accurate pose



(a) (b)

Fig. 8. Because of S-mapping and maps fusion, MS-Mapper can correct pose drifts to avoid stratification. (a) SLAM. (b) MS-Mapper.

TABLE I
COMPARISON WITH SLAM

Method Names	Advantages	Disadvantages
ALOAM	automatic pose estimation	drifts after long a time running
BALM	multi-frame bundle adjustment	more plane patterns better
FAST-LIO	filter framework based fast processing	drifts in large environment
LIO-SAM	has loop closing function	costs much time and drifts in a long corridor
LIO-Mapping	IMU aided with high accuracy	drifts without a loop closing
MS-Mapper(**Ours**)	low drift in no loop closing environment, long corridor	need to stop a while for stationary mapping

estimation and produces a high-quality map. The efficacy of MS-Mapper can be attributed to S-mapping. S-mapping generates multiple local maps, and the fusion of these maps corrects pose estimation errors. Similar conclusions are also drawn from our other experiments.

Before the summary, it should be indicated that the SLAM loop-closing function is not almighty. Some capturing traces are straight and do not have circle paths for SLAM methods to meet old places. Even though there are circle capturing paths, the loop-closing thread is only trying to disperse the final jump-error to history poses [2], rather than eliminating error. After the loop-closing optimization, a globally consistent map is built. Although there are no jump errors in the whole map, this map still has a gap to the real world. This phenomenon requires a higher pose estimation accuracy mapping system. Our MS-Mapper exactly aims to offer this ability, because we have S-mapping for the registration. In our system, errors only come from LiDAR measurement self and the S-mapping registration process. By utilizing our designed good algorithms, MS-Mapper errors can be controlled at a satisfactory level.

The primary contributions of this study are summarized as follows.

1) *New Hardware Structure:* We have designed and manufactured MS-Mapper, a mapping system that integrates the strengths of TLS and SLAM into a novel LiDAR mapping system. A corresponding workflow, with S-mapping and M-mapping switching to accommodate its unique structure, has also been developed.
2) *Configuration Selection:* We introduce an online warning algorithm for detecting pose estimation degeneration and determining optimal moments for configuration changes. This algorithm relies on residual decoupling, assessing whether current constraints are sufficient. If not, it signals for setting a S-mapping through the UI.
3) *Dynamic GMM Tree Representation:* The proposed dynamic GMM tree representation reduces data association complexity and addresses issues related to local minimum models. Particularly suitable for pose estimation tasks with ultradense point clouds, this representation enhances the robustness of the mapping system.

Collectively, these contributions enhance the performance and capabilities of LiDAR mapping systems, enabling accurate pose estimation, improved map quality, and effective configuration selection.

II. RELATED WORK

We first introduce some famous LiDAR SLAM literature. Then, we summarize their benefits and limitations in Table I.

ALOAM [3] proposes a local mapping approach for 3-D LiDAR SLAM. However, it may drift after a long time running, because ALOAM does not have the loop closing function. Building upon ALOAM, LeGO-LOAM [4] utilizes the ground assumption to enhance accuracy. IMLS-SLAM [5] defines boundaries as implicit surfaces and applies moving least squares to resample points, similar to ALOAM's local map registration. MULLS [6] employs different types of residuals (point-to-point, point-to-plane, and point-to-line) with varying weights to balance pose state dimensions. LIO-FILO [7] combines the Kalman filter and factor graph optimization to reach higher accuracy. WeCoSLAM [8] fuses LiDAR, IMU, Camera, and GPS to mapping in challenging environments. ORBLiDARSLAM [9] projects the point cloud into an image and uses the ORB feature for registration. BALM [10] uses a multipoint cloud for registration through BA.

These above SLAM methods aim to enhance SLAM robustness to noise by adopting diverse algorithms or fusing with other sensors. Our MS-Mapper is different, and we aim to design a new structure to solve the pose drift problem.

LO-Degeneracy [11] aims to prevent degeneracy in pose estimation by updating the state along an orthogonal, well-conditioned direction. By taking the dot product of the coefficient matrix with its transpose, the objective problem is transformed into a linear form, enabling the identification of the degeneracy direction. LION [12] incorporates an observability metric to self-assess performance and determine if the pose estimation is geometrically ill-constrained. DLO [13] introduces a key-frame selection method that maximizes the overlap region between the current observation and the local map. DLIO [14] builds upon DLO and incorporates an IMU to aid frame-to-frame estimation. FAST-LIO [15] and FAST-LIO2 [16] use the filter framework and model the LIO process as solving an extended Kalman filter problem, demonstrating the equivalence of the filter and Gauss–Newton methods under similar residual problem formulations. LIO-mapping [17] and LIO-SAM [18] are designed as tightly coupled nonlinear optimization problems, and they use factor graphs to update. Mid-omni [19] utilizes an omnidirectional camera in conjunction with a Livox LiDAR Mid-360 to create maps. It operates by intermittently running and stopping for visual data capture.

These above SLAM methods motivate us how to detect the pose estimation degeneracy. LO-degeneracy [11], LION [12], FAST-LIO [15], and LIO-SAM [18] emphasize the importance of registration between the current observation and the map. Hence, we identify that the registration residual contains sufficient information to assess the present pose estimation state.

We provide a summary of primarily representative research in Table I, elucidating their distinctive properties. ALOAM, BALM, FAST-LIO, and LIO-mapping lack loop-closing

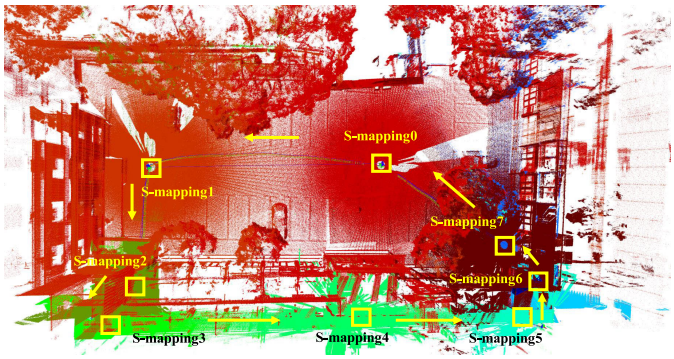


Fig. 9. Map is built by MS-Mapper. Different station locations from S-mapping0 to S-mapping7 are shown. Their corresponding point clouds are drawn in different colors.

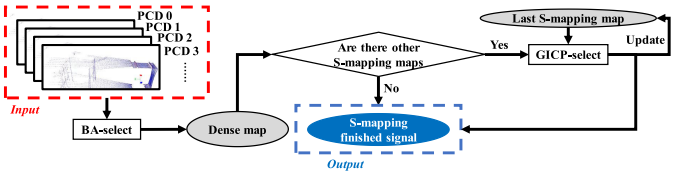


Fig. 10. S-mapping pipeline uses multipoint cloud frames as input. These PCDs are aligned together to form a dense map by the BA-select algorithm. Then, we judge whether are there other S-mapping maps. If not, the S-mapping process is finished. If yes, we use GICP-select to align the current dense map to the last S-mapping map, which can estimate the current map's pose, and correct the errors from the previous M-mapping.

capabilities, leading to potential drift after prolonged operation. While LIO-SAM incorporates loop closing, it cannot utilize this feature to correct drift in noncircular capturing traces.

In contrast, our work, MS-Mapper, employs S-mapping to generate precise local maps. These maps can be directly aligned, effectively mitigating drift errors during the motion process. Consequently, compared to the aforementioned SLAM methods, MS-Mapper exhibits significantly lower drift in mapping. It is important to note that MS-Mapper requires intermittent pauses for mapping, representing a tradeoff for its enhanced accuracy.

III. Concrete Practice

The pipeline of MS-Mapper is delineated in Fig. 4, comprising three primary components: S-mapping, M-mapping, and configuration selection. The resulting map generated by MS-Mapper is illustrated in Fig. 9, showcasing various station locations from S-mapping0 to S-mapping7. Notably, S-mapping0 represents the beginning location, while S-mapping7 denotes the final location. Each station's point cloud is visually represented using distinct colors, offering a comprehensive depiction of the entire environment captured through the S-mapping process.

A. S-Mapping

The S-mapping pipeline is shown in Fig. 10, which uses multipoint cloud frames as input. These point cloud data (PCD) are aligned together to form a dense map by the BA-select algorithm. Then, we judge whether are there other

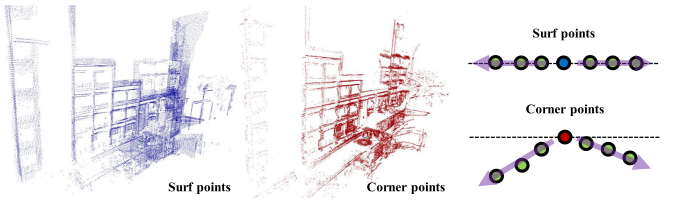


Fig. 11. In the feature detection process of MS-Mapper, each point utilizes its left and right neighboring points for PCA fitting. When the current point lies on a plane, its neighboring points should manifest only one dominant eigenvalue, and their eigenvectors should be parallel. Conversely, if the current point is situated on an edge, the eigenvectors of its neighboring points will form a dihedral angle. Through the analysis of these properties, MS-Mapper can accurately detect surf and corner feature points.

S-mapping maps. If not, the S-mapping process is finished. If yes, we use generalized iterative closest point (GICP)-select to align the current dense map to the last S-mapping map, which can estimate the current map's pose, and correct the errors from the previous M-mapping.

In MS-Mapper, the registration process utilizes feature points instead of raw measurements. Unlike the method employed in LOAM [3], which calculates neighbor curvatures, MS-Mapper introduces a novel surf/corner feature detection method depicted in Fig. 11. This method leverages the sparse distribution of LiDAR points along different laser lines and harnesses the information within each scan line.

For each point, principal component analysis (PCA) fitting is performed using its left and right neighboring points. If the current point lies on a plane, its neighboring points should exhibit only one dominant eigenvalue, and their eigenvectors should be parallel. Conversely, if the current point is situated on an edge, the eigenvectors of its neighboring points will form a dihedral angle. Through the analysis of these properties, MS-Mapper can accurately detect surf and corner feature points.

There are inherent limitations in step-motor accuracy, hardware synchronization, and assembly errors. Combining raw point clouds directly may lead to vagueness and distortions, especially when the S-mapping configuration relies solely on angle feedback. This issue becomes particularly pronounced at the start and end positions of the mapping process.

To mitigate this challenge, the BA technique is employed. In this approach, all points from the point cloud are organized into a voxel map with a grid size of 0.1 m. For each grid, a singular value decomposition (SVD) is performed to calculate the mean and covariance of the points within that grid. The obtained eigenvalues from the SVD are denoted as λ_k, where $\lambda_1 < \lambda_2 < \lambda_3$.

The primary objective of the BA is to minimize the value of λ_1 by adjusting the LiDAR poses. This adjustment aims to make the point cloud formed by the LiDAR measurements thinner. A thinner point cloud contributes to a more accurate representation of the environmental surface. This adjustment process is critical in mitigating distortions induced by pose estimation errors.

Assume that there are total M times measured LiDAR poses $\mathbf{T}^i = \begin{bmatrix} \mathbf{R}^i & \mathbf{t}^i \\ \mathbf{0} & 1 \end{bmatrix} \in \mathrm{SE}(3)$ and $i = 1, \ldots, M$. For the

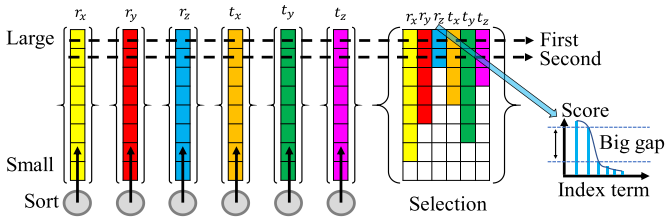


Fig. 12. To evaluate the influence of each residual on the pose result, we project all residuals into six dimensions and analyze their derivatives to represent their impact on the pose. The size of the Jacobian matrix defines the score of each residual. Residuals exhibit sensitivity to changes in the pose state. This means that a slight disturbance in the pose state leads to significant changes in result errors. This phenomenon is valuable for our registration process. These residuals provide stronger constraints. Therefore, we rank all projected residuals based on their scores in parallel and select the top subset with the highest scores for registration.

i index LiDAR measurement, the feature residual pairs are $P^i = \{\mathbf{p}_j^i\}$, $Q^i = \{\mathbf{q}_j^i\}$, and $j = 1, \ldots, N_i$. The BA objective is to minimize point-to-plane distances by adjusting M LiDAR poses $\mathbf{T}^i$

$$\mathbf{T}^i = \arg\min_{\mathbf{T}^i \in \mathrm{SE}(3)} \sum_{i=1}^{M} \sum_{j=1}^{N_i} \left[\mathbf{n}_j^{i\,T} \left(\mathbf{T}^i \mathbf{p}_j^i - \mathbf{q}_j^i \right) \right]^2. \tag{1}$$

The solution to (1) follows a methodology akin to that employed in Eigen factors [20], BALM [10], PA [21], and PiSLAM [22]. The optimization process utilizes the Ceres optimization library [23]. The key lies in specifying the optimization residual form and its Jacobian form to Ceres, along with providing initial values. Ceres then performs optimization and returns the refined values. To get Jacobian, we define the center point $\bar{\mathbf{p}}^i$ and the covariance matrix $\mathbf{A}^i$ as

$$\bar{\mathbf{p}}^i = \frac{1}{N_i} \sum_{j=1}^{N_i} \left(\mathbf{T}^i \mathbf{p}_j^i \right)$$

$$\mathbf{A}^i = \frac{1}{N_i} \sum_{j=1}^{N_i} \left[\mathbf{T}^i \left(\bar{\mathbf{p}}^i - \mathbf{p}_j^i \right) \right]. \tag{2}$$

Our objective is to minimize the thickness of $\mathbf{A}^i$. Assume that $\mathbf{A}^i$ has eigenvalues λ_k^i corresponding to eigenvectors $\mathbf{u}_k^i$ and $k = 1, 2, 3$. Then, the Jacobian is

$$\frac{\partial \lambda_k^i}{\partial \mathbf{p}_j^i} = \frac{2}{N_i} \left[\mathbf{T}^i \left(\bar{\mathbf{p}}^i - \mathbf{p}_j^i \right) \right]^T \mathbf{u}_k^i \mathbf{u}_k^{i\,T}. \tag{3}$$

Equation (3) is the Jacobian form, which is sent to the optimization library. These residuals are collected in each grid.

Overall, the optimization objective aims to minimize the mean point distribution $\bar{\mathbf{p}}^i$ and $\bar{\mathbf{q}}^i$, along with the covariance $\mathbf{\Sigma}^i$. These values are derived from the PCD. They are instrumental in quantifying the distribution and characteristics of points within each grid.

In accordance with our residual contribution theory [24], a subset of feature points is strategically chosen for the BA process, as illustrated in Fig. 12. This selection is guided by the concept that certain feature points contribute more significantly to the accuracy of point estimation. To determine this subset, we project all residuals into every six dimensions and employ derivatives to quantify their impact on the pose result. The size of the derivative represents the score of each residual, with some being notably sensitive to changes in the pose state. Residuals exhibiting this sensitivity imply that even a minor disturbance in the pose state can result in substantial changes in error. Such residuals are deemed valuable for registration due to their capacity to provide stronger constraints.

Consequently, we rank all projected residuals in parallel based on their scores and select the top subset with the highest scores for registration. By focusing on this refined subset of feature points, MS-Mapper achieves heightened accuracy compared to the original method.

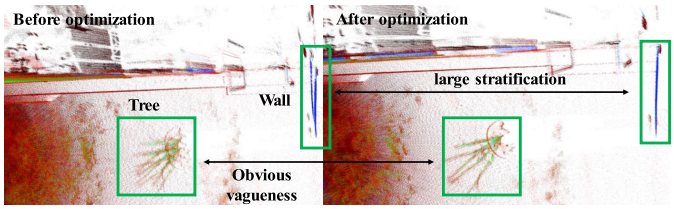


Fig. 13. Directly combining LiDAR measurements without optimization can lead to distortions and result in a vague point cloud representation. However, after applying our optimization process, these distortions are mitigated. The optimized point cloud exhibits improved clarity, especially in terms of wall structures where distortions are eliminated, and the outline of objects such as trees becomes more defined.

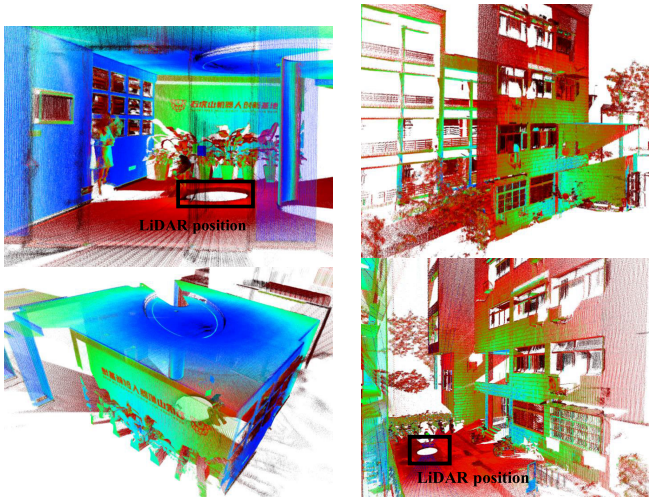


Fig. 14. Point cloud map is constructed through the S-mapping process. In this representation, the color of each point corresponds to the laser point intensity. Specifically, blue points indicate higher intensities, suggesting that the corresponding objects reflect more laser. Conversely, red points indicate a lower intensity, signifying that the corresponding objects absorb more laser. This color scheme provides a visual representation of the reflective properties of the objects within the point cloud map. (a) Room. (b) Building.

In Fig. 13, the left point cloud result is obtained by utilizing the encoder feedback, where the encoder angle is employed for aligning the point cloud. In this case, the point cloud result exhibits vagueness and distortions, particularly noticeable in the tree and wall structures. However, after applying the BA, the right point cloud becomes clearer, and distortions are significantly reduced. As depicted in Fig. 14(a) and (b), the resulting map showcases clear and well-defined features, with dense yet thin points. This refined map provides accurate

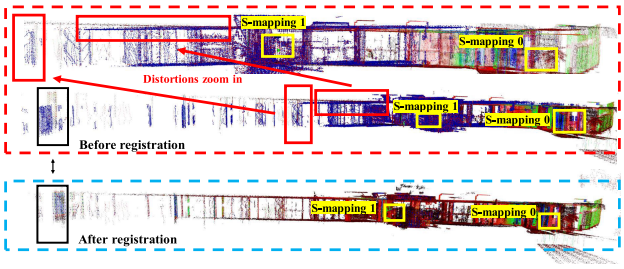


Fig. 15. Red dashed box shows accumulative errors from the mobile process between S-mapping0 and S-mapping1. The blue dashed box is our effect, which corrects the pose drift from moving.

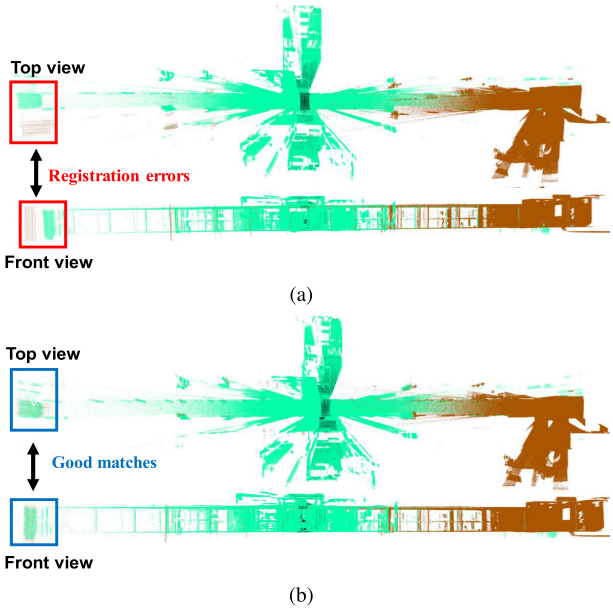


Fig. 16. GICP utilizes all valid residuals for optimization. While nearby points tend to match well, the influence of far-away residuals on the objective function is relatively small. In contrast, the GICP-select method is designed to homogenize all six dimensions and consider more selection opportunities for far-away residuals. This characteristic contributes to a more effective registration result compared to the original GICP, as it places greater emphasis on distant residuals in the optimization process. (a) GICP. (b) GICP-select (ours).

and informative prior information for both M-mapping and S-mapping during their registration processes.

The map fusion step is activated upon the completion of a new S-mapping map. If there exist old S-mapping maps, a registration process is initiated between the new and old maps. This direct registration aligns the new map to its proper location, correcting pose drift in the entire mapping process. The effectiveness of this correction is demonstrated in Fig. 15, where the misalignment between the old and new maps is reduced. This registration employs a modified version of the GICP method, known as GICP-select [25]. The original GICP may have registration errors in Fig. 16(a), GICP-select offers more opportunities to select far away residuals, which can achieve a higher accuracy alignment in Fig. 16(b).

B. M-Mapping

The M-mapping pipeline is shown in Fig. 17. It uses the current point cloud as input. There is a preprocess that makes the point cloud map into the Gaussian model map.

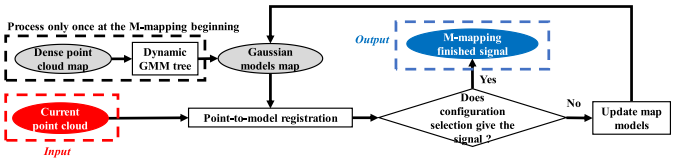


Fig. 17. M-mapping pipeline uses the current point cloud as input. There is a preprocess that makes the point cloud map into the Gaussian model map. Then, the registration can calculate the pose. The pose and point cloud are sent into the configuration selection module for judging. If not, then the current information is going to update map models. If yes, M-mapping should finish, and MS-Mapper switches to S-mapping.

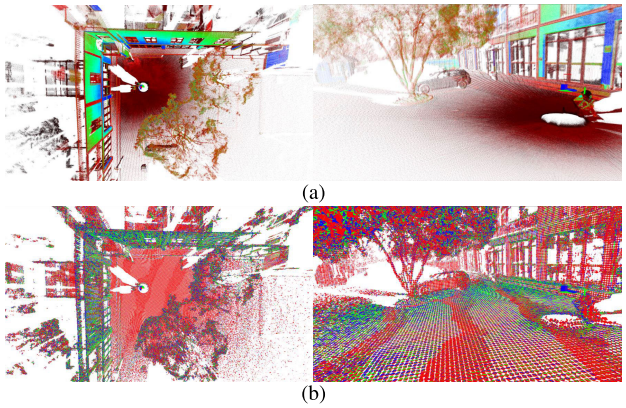


Fig. 18. Built map is ultradense, which makes the pose estimation cost a lot of time. The Gaussian model utilizes much less amount to represent the space geometry information. (a) Dense point cloud. (b) Gaussian models.

Then, the point-to-model registration can calculate the pose. The pose and point cloud are sent into the configuration selection module for judging. If not, then the current information is going to update map models. If yes, M-mapping should finish, and MS-Mapper switches to S-mapping.

In the M-mapping process, the current laser scans serve as input. However, the motion of the LiDAR during mobile capturing can introduce distortions in the captured point cloud, which is an inevitable challenge. To mitigate this issue, MS-Mapper employs a constant speed motion compensation technique as a preprocessing step, aiding in the undistortion of each frame.

For the M-mapping, the algorithm utilizes the current point cloud and performs a local map registration. However, the local map generated by the preceding S-mapping is extremely dense, imposing a substantial computational burden. To address this challenge, the dense point cloud map is transformed into a more compact representation comprising Gaussian models.

The original point cloud map contains approximately 48 million points and occupies around 768 MB of memory [Fig. 18(a)]. It is converted into the Gaussian model representation depicted in Fig. 18(b). This transformation reduces the total number of models to just 9563, which significantly decreases the data size. Despite this reduction, the essential geometric information of the environment is preserved. The Gaussian model representation not only expedites the data association process but also retains the visibility of spatial geometry information.

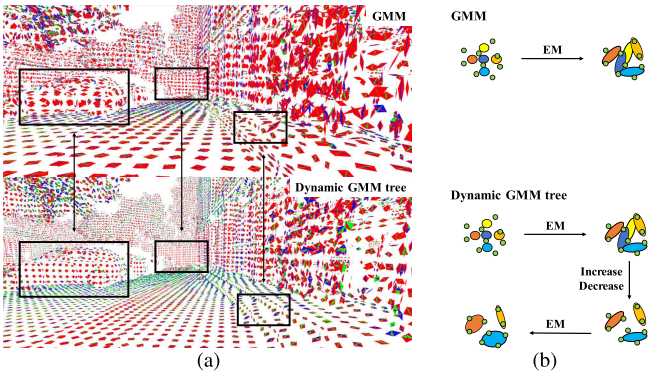


Fig. 19. Models on the car and stairs are badly fit, whereas others on the ground are clustered together. Our method removes bad models and diffuses aggregations. (a) Models fitting effect. (b) Work flow.

MS-Mapper introduces a dynamic GMM tree representation, a hybrid approach inspired by the normal distribution transform (NDT) [26], GMM [27], and coherent point drift (CPD) [28].

In the traditional GMM approach [27], the point cloud is segmented into fixed-length grids, and a predetermined number of Gaussian models are randomly or uniformly placed within these grids. The expectation-maximization (EM) algorithm is then applied to adjust the mean and covariance of the Gaussian models through several iterations. The goal of EM is to minimize the distances between points and the fit Gaussian distributions.

In Fig. 19(a), a comparison between the traditional GMM and the dynamic GMM tree representation reveals that the traditional GMM exhibits poor fitting on the car and stairs, with models on the ground clustered together. In contrast, the dynamic GMM tree representation eliminates poorly fit models and diffuses aggregations, resulting in a more accurate representation of the environment.

In contrast, the workflow of our dynamic GMM tree representation has more adjusting steps than the traditional GMM in Fig. 19(b). We also introduce a checking step to enhance the accuracy and flexibility of the model fitting process. The space is divided into larger grids, and an initial set of Gaussian models is assigned as leaves. After EM iterations, the weight vectors of the models are examined. The probability of each point belonging to a specific Gaussian is calculated. Gaussian models are then added to poorly fit points, increasing the number of children Gaussians. Only the E-step of EM is applied to these new models for refining their fitting. Each Gaussian undergoes SVD to determine if it represents a line or a plane. Line models are discarded and replaced by adjacent planes. The EM algorithm is employed again to further refine the models. Plane models are expanded to the next leaf level, treating ellipsoids as parent nodes for iterative refinement. Typically, two levels of children are sufficient for practical applications.

During the M-mapping registration process, the current sensor pose is optimized to minimize distances formed by current LiDAR points and their corresponding Gaussian distributions. Data associations are built using initial pose projection under the nearest principle. After calculation, new observed points are added to the local map, and all valid Gaussian models are updated based on the recalculation of mean and covariance. This update enhances the stability of Gaussian models.

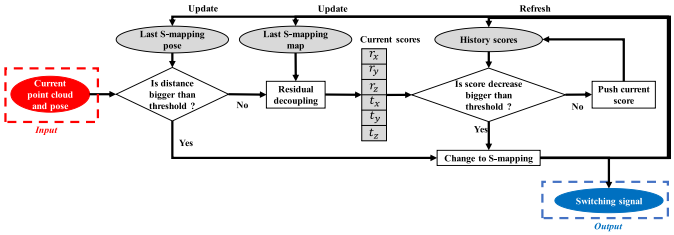


Fig. 20. Current residuals are decoupled into six dimensions, and the resulting scores are analyzed. If the current score exhibits a substantial gap disparity compared to the scores from historical frames, the configuration selection module issues a warning. This warning signals the potential need to switch to S-mapping, indicating a significant deviation in the current pose estimation compared to historical trends.

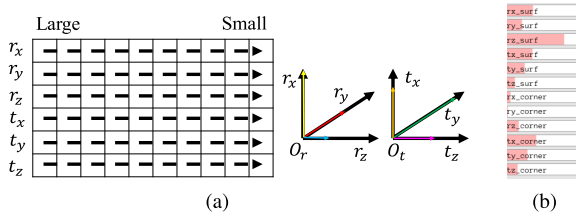


Fig. 21. All residuals are decomposed into six dimensions based on their contributions, with three dimensions for rotation and the remaining three for translation. These scores are dynamically displayed in our UI. For instance, in scenarios such as entering a long corridor, the score associated with the t_z-axis may experience a sudden decrease. (a) Sort. (b) UI.

C. Configuration Selection

In Fig. 20, the details of MS-Mapper's configuration selection module are illustrated. First of all, this module only works when MS-Mapper is in the M-mapping phase because S-mapping can automatically terminate after it finishes a circle scanning. The configuration selection module judges whether the current pose is far away from the last S-mapping position. If the distance is bigger than the threshold, such as 20 m in our experiments, the module gives a switching signal to suggest changing to S-mapping. If the distance is not bigger than the threshold, the configuration selection module involves associating the current point cloud with the last S-mapping map, which forms residuals. These residuals are then decoupled into six dimensions: three for rotation and three for translation. All residuals serve as constraints for the pose estimation. So, they are calculated as the current score indicating how tight one dimension is constrained. By comparing the current score with history scores, if the score decrease is bigger than the threshold, the configuration selection module gives a switching signal, and MS-Mapper should put a station here by changing it into S-mapping.

The real-time visualization of scores is depicted in Fig. 21(a) and (b). A sliding window of 30 frames is maintained to store the scores for each dimension, offering a history of the scores' behavior.

If the current score in any dimension exhibits a substantial deviation from its historical range, a warning signal advocating a switch to S-mapping is triggered. This warning indicates that

a fixed station should be established at the current location, suggesting that the pose estimation performance is experiencing degradation or unreliability. This mechanism is designed to detect opportunities where MS-Mapper requires S-mapping to enhance accuracy and reliability in pose estimation.

The residual decoupling process of MS-Mapper is detailed here. It involves calculating the sensitivity of the objective function to small disturbances in the pose state. This sensitivity is quantified by the Jacobian matrix in (4), which delineates how the objective function is impacted by perturbations in the pose parameters

$$\mathbf{J}_i = \begin{cases} \begin{bmatrix} (\mathbf{p}_i \times \mathbf{n}_i)^{\mathrm{T}} & \mathbf{n}_i^{\mathrm{T}} \end{bmatrix}_{1\times 6}, & \text{(point-to-plane)} \\ \begin{bmatrix} (\mathbf{n}_i^{\mathrm{T}}\mathbf{p}_i)\mathbf{I}_3 - \mathbf{p}_i\mathbf{n}_i^{\mathrm{T}} & \mathbf{n}^{\wedge} \end{bmatrix}_{6\times 3}, & \text{(point-to-line).} \end{cases} \tag{4}$$

The Jacobian matrix comprehensively considers the derivatives of point-to-plane and point-to-line distances with respect to the sensor pose. It quantifies the influence of small changes in the pose parameters on the objective function. The Jacobian matrix accounts for both the distances and directions between the LiDAR points and Gaussian models.

In specific scenarios, such as navigating a long corridor, where the LiDAR motion is predominantly along the z-axis, the score associated with the z-axis dimension may experience a sudden decrease. A small z-axis score indicates that very few points are measured in the z-dimension. Consequently, the pose estimation result becomes less reliable in the z-direction, exhibiting characteristics akin to drift.

IV. Experiments

In the evaluation of map accuracy (Section IV-A), multiple metrics, including point cloud thickness and model fitting error, are employed to assess the accuracy of the generated maps. For the evaluation of pose estimation accuracy (Section IV-B), metrics such as absolute pose error (APE) are utilized to gauge the accuracy and reliability of the pose estimation algorithm. The evaluation of time cost (Section IV-C) involves recording the time cost of MS-Mapper and comparing it with some conventional SLAM methods.

Section IV-A tries to demonstrate that dynamic GMM and selection-based BA are good for mapping results. Section IV-B wants to evaluate that MS-Mapper can work well in challenging environments. Section IV-C is going to illustrate that MS-Mapper can also work in real-time performance.

Fig. 22 shows the hardware of MS-Mapper. It is a step motor and steering gear-based mechanical structure, which can be carried by the vehicle and hand. The experiments were conducted using C++ and executed on a computer equipped with a six-core CPU (AMD 2600×), 48 GB of RAM, and an Nvidia RTX 2070 GPU. It is noteworthy that the GPU was exclusively employed for OpenGL rendering and not utilized for the core processing tasks.

A. Map Accuracy

In Fig. 23(a), noticeable stratification is observed in the ground points after completing a circular scan. However, the optimization process applied in Fig. 23(b) adjusts the sensor poses, eliminating map conflicts. The observed stratification

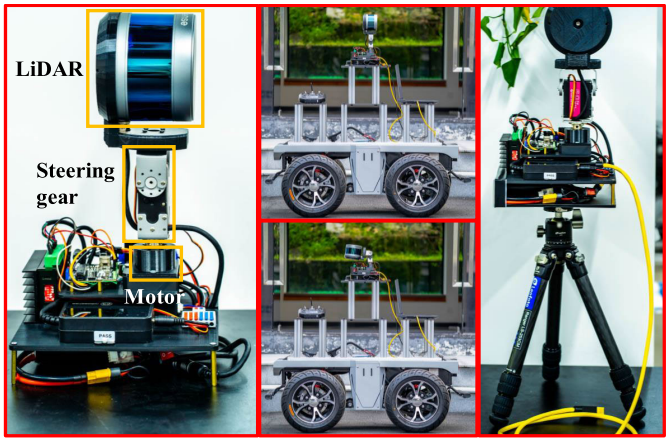


Fig. 22. Hardware of MS-Mapper is a step motor and steering gear-based mechanical structure, which can be carried by the vehicle and hand.

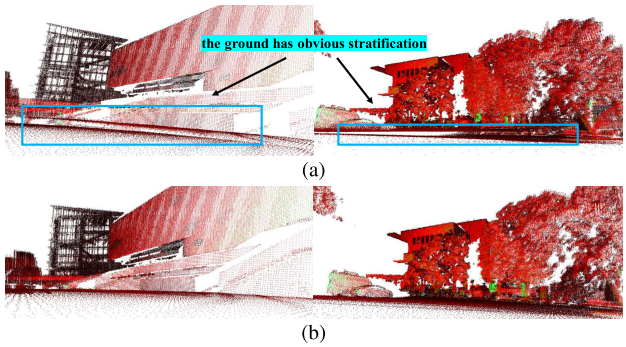


Fig. 23. Before BA, the ground points have obvious stratification. Because the assembling is hard to place the LiDAR in the center of the rotary axis, the continuous estimation has drifted. When a circle scan is finished, the optimization adjusts all sensor poses to eliminate map conflicts. (a) Before BA. (b) After BA.

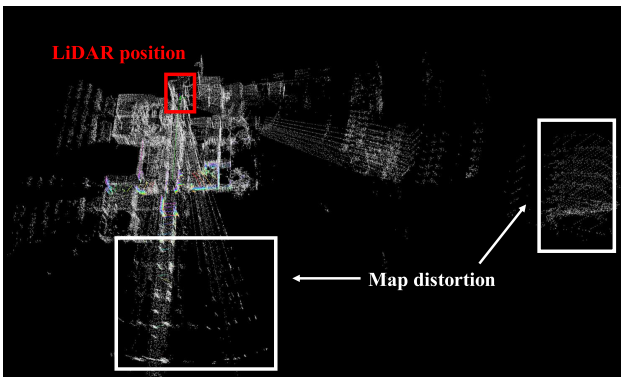


Fig. 24. If the MS-Mapper only uses the encoder angle feedback, it may cause map distortion, because of step-motor accuracy, hardware synchronization, and assembling errors.

is attributed to limitations in step-motor accuracy, hardware synchronization, and assembly errors, as illustrated in Fig. 24.

To quantitatively assess the map quality, the thickness of the planes is defined and measured using CloudCompare software, as shown in Fig. 25. This manual measurement enables an evaluation of the accuracy and consistency of the map's-plane representation. Table II presents measurements from five distinct real-world wall and floor locations, demonstrating the

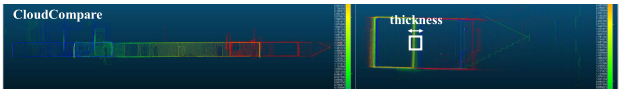


Fig. 25. We use CloudCompare software to measure the plane's thickness manually. The wall surface thickness can reflect the map's accuracy.

TABLE II
THICKNESS

	RMSE/cm					residual terms
S-mapping (withoutBA)	6.25	5.13	4.11	5.34	5.08	—
S-mapping (BA)	4.06	3.37	3.12	3.59	3.08	4863
S-mapping (BA-select)	**2.98**	**3.11**	**1.83**	**2.19**	**2.43**	**2151**

TABLE III
MODEL FITTING

environments		flat wall	engine hood	stairways	human body	tree leaves
GMM	terms	64	63	64	64	64
	error/cm	**0.64**	1.49	2.57	3.46	4.48
GMM (dynamic tree)	terms	23	33	41	87	112
	error/cm	0.89	**1.24**	**1.90**	**2.86**	**3.13**

efficacy of MS-Mapper in accurately capturing the geometry of planes in the environment.

In Table II, the first method, S-mapping (without BA), represents the continuous pose estimation without applying BA. The estimated thickness of the planes has larger errors, approximately 5 cm, compared to the other methods. In the second method, S-mapping (BA), the estimated thickness of the planes is reduced to approximately 3 cm. This indicates that the BA process significantly improves the accuracy and consistency of the map by adjusting the sensor poses. The third method, S-mapping (BA-select), discards large uncertainty residuals, and the thickness reaches about 2 cm, which is close to the raw LiDAR measurement error.

Table III presents quantitative results for evaluating the effectiveness of the dynamic GMM tree representation in MS-Mapper. The employed evaluation metric is the fitting error, defined as the point-to-model distance error. It reflects the accuracy of associating points with the models in the map.

To compute the fitting error, a 0.1-m cube is defined, and the points within this cube are associated with their corresponding models. The error summation is then calculated based on the probabilities of generating ellipsoids for each model.

It is important to note that this indicator is meaningful when the model counts are equal. If each point has a unique model, the error will be exactly zero. Therefore, the model counts and errors for the entire bottom layer nodes are recorded for comprehensive analysis.

In the case of a flat wall, our method tends to reduce redundant models, leading to a slightly larger distance error compared to the traditional GMM. This is because the flat wall offers a relatively simple and structured environment, where a smaller number of models can adequately represent the geometry. However, the reduction in models may result in a slightly higher error, as it sacrifices some level of detail.

As the environments become more unstructured, such as with human bodies and tree leaves, our method utilizes fewer models but achieves better fitting results. This phenomenon indicates that the dynamic GMM tree effectively captures and represents the complex geometry of these scenarios, resulting in smaller distance errors compared to the traditional GMM method. These environments demand more sophisticated modeling techniques to accurately capture intricate details.

Although our method achieves smaller errors in complex scenarios, it may require a higher number of models to capture complete details. However, it is crucial to consider that, in the context of pose estimation, these areas may not be ideal for accurate localization due to high complexity and occlusion. Therefore, the focus is often on obtaining a reasonable representation without sacrificing computational cost.

TABLE IV
LOOP CLOSURE ERROR

	APE/m				
Environments	Outdoor	Outdoor	Half	Indoor	Indoor
Names	College	Library	Technology Park	Institute1	Institute2
ALOAM	1.38	1.45	2.02	2.73	2.56
BALM	0.96	0.77	1.26	1.10	1.23
FAST-LIO	0.25	0.28	0.71	1.15	1.24
LIO-SAM	0.34	0.30	0.64	0.94	0.65
LIO-Mapping	0.29	0.31	0.52	0.88	0.74
MS-Mapper(**Ours**)	**0.16**	**0.20**	**0.25**	**0.24**	**0.27**

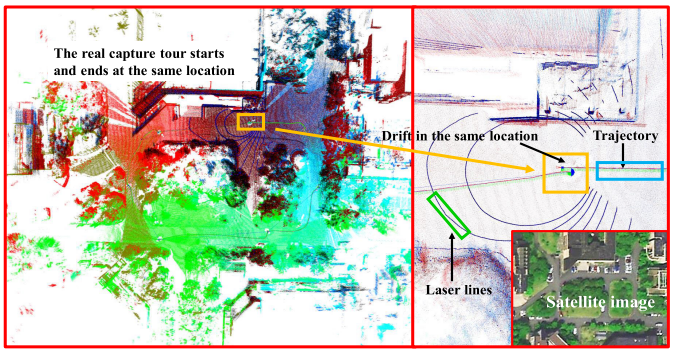


Fig. 26. MS-Mapper starts and ends at the same location. The translation error is defined as a loop-closure error, which reflects the pose estimation accuracy.

B. Pose Estimation Accuracy

Table IV provides a comparison of various LiDAR SLAM methods in terms of pose estimation error. The evaluation is carried out in both indoor and outdoor environments, with a focus on loop-closure error by starting and ending at the same location, as illustrated in Fig. 26. Pose outputs are recorded for each method, and the APE is used for evaluation. The following methods are included in the comparison: ALOAM, BALM, FAST-LIO, LIO-SAM, and LIO-mapping.

In the outdoor sequences, FAST-LIO utilizes IMU assistance and operates in open wide scenes. ALOAM and BALM lack a loop-closure function, leading to larger errors resulting from cumulative pose estimations. This observation aligns with the expectation that MS-Mapper performs better than SLAM-based methods. The ability of two adjacent S-mapping processes can strongly control drifts between them. It contributes to the improved performance of MS-Mapper.

In both indoor and outdoor environments, MS-Mapper demonstrates superior performance. The drift in FAST-LIO begins in narrow scenarios, such as long corridors shown

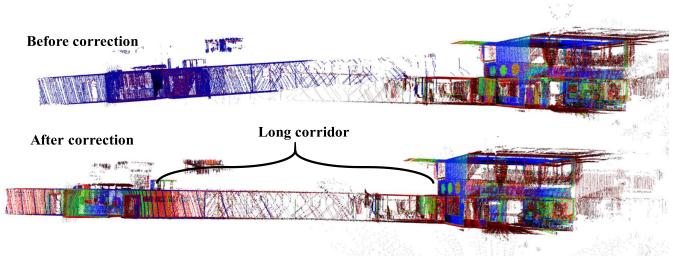


Fig. 27. Long corridor scene is easy to drift.

TABLE V
TIME COST WITH SLAM

	Mean Pose Calculation Time/ms				
Environments	Outdoor	Outdoor	Half	Indoor	Indoor
Names	College	Library	Technology Park	Institute1	Institute2
ALOAM*	279	303	268	172	183
BALM	125	139	117	92	98
FAST-LIO**	**113**	**122**	**107**	**91**	**94**
LIO-SAM**	138	146	130	112	124
LIO-Mapping**	141	153	134	118	126
	S-mapping/M-mapping				
MS-Mapper(**Ours**)	125/139	137/143	109/119	96/113	97/118

* We record the mapping thread pose publishing time rather than the odometry thread
** We record the point cloud processing time rather than IMU fused pose publishing

in Fig. 27. BALM occasionally outperforms FAST-LIO, likely due to the presence of structured walls and floors in the environment, providing effective constraints for BA. MS-Mapper maintains high accuracy and avoids drift in long corridors, thanks to the rich information obtained from a single S-mapping session.

We also compare MS-Mapper with LIO-SAM and LIO-mapping methods. For a fair comparison, the loop-closure method in LIO-SAM is disabled. Otherwise, there will be no error in the circular capturing trace. In outdoor environments, LIO-SAM and LIO-mapping outperform ALOAM, BALM, and FAST-LIO. This improvement can be attributed to the IMU's ability to correct motion distortion. The improvement is also achieved through factor graph optimization to receive higher accuracy compared to filter-based methods.

Overall, MS-Mapper exhibits promising results in various environments, surpassing SLAM-based methods in terms of pose estimation accuracy. Its capacity can effectively handle complex scenarios and provide accurate localization positions. It is a robust candidate for mapping applications.

C. Time Cost

Table V provides a comparison of the pose calculation time cost between MS-Mapper and various SLAM methods. The mean time for each sequence, from receiving the point cloud message to publishing the pose result, is recorded separately for MS-Mapper's S-mapping and M-mapping modes. For ALOAM, the time is recorded by the mapping thread to publish the pose. While for other IMU-aided SLAM methods, the point cloud processing time is recorded, instead of the IMU-fused pose publishing time.

Table V illustrates the time cost of pose calculation for MS-Mapper and various SLAM methods in different sequences. Notably, the time cost depends on the complexity and characteristics of the environment. In outdoor environments with more scattered point clouds, SLAM methods tend to extract more feature points, resulting in increased time costs. FAST-LIO, being a filter-based method, exhibits the lowest time cost. While LIO-SAM and LIO-mapping, involving factor graph optimization with IMU measurements, have relatively higher time costs. MS-Mapper, in its S-mapping mode, performs similar to BALM with a processing frequency of nearly 10 Hz. The M-mapping mode, which involves scan-to-map registration over a larger space, incurs a slightly higher time cost but remains within real-time performance.

V. CONCLUSION

In this article, we design a novel LiDAR mechanical structure called MS-Mapper, which combines the advantages of TLS and LiDAR SLAM. We also develop the corresponding algorithm to work in challenging environments, such as the long corridor or no loop-closing capturing trace. We propose an online warning algorithm for configuring changes. We also propose the dynamic Gaussian model representation to suit different environmental characteristics.

MS-Mapper employs S-mapping to generate accurate local maps. These local maps can be directly aligned, effectively mitigating drift errors during the motion process. Consequently, compared to the traditional SLAM methods, MS-Mapper exhibits significantly lower drift in mapping. It is important to note that MS-Mapper requires intermittent pauses for mapping, representing a tradeoff for its enhanced accuracy.

Our future objective is to investigate methods for processing the dense maps into mesh-based models. The dense maps are generated by MS-Mapper. This avenue of research holds promise for enhancing visualization, optimizing data storage, and supporting various mesh-based applications.

REFERENCES

[1] C. Cadena et al., "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.
[2] T. Barfoot, *State Estimation for Robotics*. Cambridge Univ. Press, 2017.
[3] J. Zhang and S. Singh, "LOAM: LiDAR odometry and mapping in real-time," in *Proc. Robot., Sci. Syst. X*, Jul. 2014, pp. 1–9.
[4] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 4758–4765.
[5] J.-E. Deschaud, "IMLS-SLAM: Scan-to-model matching based on 3D data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 2480–2485.
[6] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "MULLS: Versatile LiDAR SLAM via multi-metric linear least square," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11633–11640.
[7] J. Tang, X. Zhang, Y. Zou, Y. Li, and G. Du, "A high-precision LiDAR-inertial odometry via Kalman filter and factor graph optimization," *IEEE Sensors J.*, vol. 23, no. 11, pp. 11218–11231, Sep. 2023.
[8] V. Kachurka et al., "WeCo-SLAM: Wearable cooperative SLAM system for real-time indoor localization under challenging conditions," *IEEE Sensors J.*, vol. 22, no. 6, pp. 5122–5132, Mar. 2022.
[9] W. Ali, P. Liu, R. Ying, and Z. Gong, "A feature based laser SLAM using rasterized images of 3D point cloud," *IEEE Sensors J.*, vol. 21, no. 21, pp. 24422–24430, Nov. 2021.
[10] Z. Liu and F. Zhang, "BALM: Bundle adjustment for LiDAR mapping," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3184–3191, Apr. 2021.
[11] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 809–816.

[12] A. Tagliabue et al., "LION: LiDAR-inertial observability-aware navigator for vision-denied environments," 2021, *arXiv:2102.03443*.
[13] K. Chen, B. T. Lopez, A.-A. Agha-Mohammadi, and A. Mehta, "Direct LiDAR odometry: Fast localization with dense point clouds," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2000–2007, Apr. 2022.
[14] K. Chen, R. Nemiroff, and B. T. Lopez, "Direct LiDAR-inertial odometry," 2022, *arXiv:2203.03749*.
[15] W. Xu and F. Zhang, "FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.
[16] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO$_2$: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.
[17] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3D LiDAR inertial odometry and mapping," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 3144–3150.
[18] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled LiDAR inertial odometry via smoothing and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 5135–5142.
[19] Z. Miao et al., "Coarse-to-Fine hybrid 3D mapping system with co-calibrated omnidirectional camera and non-repetitive LiDAR," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1778–1785, Mar. 2023.
[20] G. Ferrer, "Eigen-factors: Plane estimation for multi-frame and time-continuous point cloud alignment," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1278–1284.
[21] L. Zhou, D. Koppel, and M. Kaess, "LiDAR SLAM with plane adjustment for indoor environment," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 7073–7080, Oct. 2021.
[22] L. Zhou, S. Wang, and M. Kaess, "P-LSAM: LiDAR smoothing and mapping with planes," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 5751–5757.
[23] S. Agarwal, K. Mierle, and T. C. S. Team, "Ceres solver," in *Proc. IEEE Eur. Conf. Comput. Vis. (ECCV)*, 2010, pp. 29–42.
[24] Z. Wan et al., "Enhance accuracy: Sensitivity and uncertainty theory in LiDAR odometry and mapping," *CoRR*, vol. abs/2111.07723, 2021.
[25] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Voxelized GICP for fast and accurate 3D point cloud registration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11054–11059.
[26] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2003, pp. 2743–2748.
[27] B. Eckart, K. Kim, A. Troccoli, A. Kelly, and J. Kautz, "Accelerated generative models for 3D point cloud data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5497–5505.
[28] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 12, pp. 2262–2275, Dec. 2010.

Zeyu Wan received the B.S. degree in automation from Nankai University, Tianjin, China, in 2018. He is now pursuing the Ph.D. degree with the College of Control Science and Engineering, Zhejiang University, Hangzhou, China.

His research interests include simultaneous localization and mapping (SLAM), 3-D reconstruction, machine learning, and robotics.

Changjian Jiang received the B.S. degree in robotics from Zhejiang University, Hangzhou, China, in 2023.

His research interests include 3-D reconstruction, simultaneous localization and mapping (SLAM), and robotics.

Xiaolong Wang received the B.S. degree in automation from Zhejiang University, Hangzhou, China, in 2022, where he is currently pursuing the M.S. degree in control science and engineering.

His research interests include simultaneous localization and mapping (SLAM), computer vision, and deep learning.

Ruilan Gao received the B.E. degree in automation from Zhejiang University, Hangzhou, China, in 2023, where he is now pursuing the Ph.D. degree with the College of Control Science and Engineering.

His research interests include bio-inspired simultaneous localization and mapping (SLAM), machine learning, and neurorobotics.

Yu Zhang received the B.S. degree in information engineering from Xi'an Jiaotong University, Xi'an, China, in 2003, and the M.S. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 2009.

He is now an Associate Professor at the College of Control Science and Engineering, Zhejiang University, Hangzhou, China. His research interests include visual navigation, intelligent control, computer vision, and intelligent autonomous systems.