MS-Mapping: Multi-session LiDAR Mapping with Wasserstein-based Keyframe Selection

Author Names Omitted for Anonymous Review. Paper-ID [add your ID here]

***Abstract*—Large-scale multi-session LiDAR mapping plays a crucial role in various applications but faces significant challenges in data redundancy and pose graph scalability. This paper present MS-Mapping, a novel multi-session LiDAR mapping system that combines an incremental mapping scheme with support for various LiDAR-based odometry, enabling high-precision and consistent map assembly in large-scale environments. Our approach introduces a real-time keyframe selection method based on the Wasserstein distance, which effectively reduces data redundancy and pose graph complexity. We formulate the LiDAR point cloud keyframe selection problem using a similarity method based on Gaussian mixture models (GMM) and tackle the real-time challenge by employing an incremental voxel update method. Extensive experiments on large-scale campus scenes and over $12.8\,\mathrm{km}$ of public and self-collected datasets demonstrate the efficiency, accuracy, and consistency of our map assembly approach. To facilitate further research and development in the community, we make our code[1] and datasets publicly available.**

I. Introduction

Large-scale multi-session LiDAR mapping has become increasingly crucial for a wide range of applications, such as surveying, autonomous driving, and multi-agent navigation. However, achieving high accuracy, consistency, and efficiency remains a significant challenge for existing approaches[5]. One of the primary challenges is the linear growth of LiDAR point cloud data as the mapping scale increases, leading to data redundancy and pose graph scalability. This often results in severe performance degradation of the mapping system and hinders long-term scalability. Moreover, the presence of redundant data further exacerbates the computational complexity and memory overhead, making it challenging to perform large-scale mapping on resource-constrained platforms. Researchers have proposed two main categories of methods to address these issues: rational keyframe selection and pose graph pruning or sparsification. However, the vast number of feature points in LiDAR point clouds makes optimization-based keyframe selection methods impractical, while pose graph sparsification approaches often suffer from accuracy loss and struggle to run in real-time. This paper stresses the following aspects:

- We propose MS-Mapping, a novel incremental multi-session LiDAR mapping algorithm that employs an incremental mapping scheme and flexibly supports various LiDAR-based odometry front-ends, enabling high-precision map assembly in large-scale environments.
- We formulate the LiDAR keyframe selection problem using GMM similarity for the first time, combined with an incremental voxel update method to achieve real-time updates of Gaussian parameters, thereby capturing the impact of keyframes on subtle changes in the map.
- We introduce a real-time keyframe selection method based on the Wasserstein distance to reduce data redundancy and pose graph complexity.

[1]https://github.com/JokerJohn/MS-Mapping

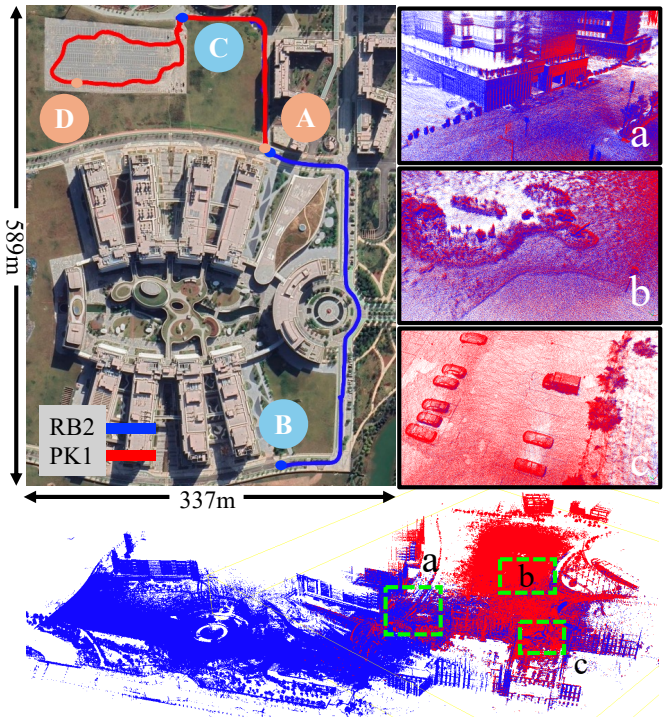

Fig. 1: **Multi-session mapping in the HKUSTGZ campus**. Top left: Projections of data sequences `RB2` (A→C→B) and `PK1` (A→C→D→A) on Google Maps. Bottom: Incremental mapping results using the MS-mapping algorithm with `PK1` (red point cloud) on `RB2` (blue point cloud). (a) Clearly visible flower bed in the overlapping area at the starting point. (b) Sharp grass details in the overlapping region. (c) Distinct car edges in the minimally overlapping area of `PK1`.

II. Problem Formulation

In this section, we assume that the reader is familiar with the basic theories of state estimation, such as maximum a posteriori (MAP) estimation and factor graphs[1].

Let $\mathcal{W}$ denote the common world frame. The transformation from the body frame (IMU frame) $\mathcal{B}$ to the world frame at time t_i is represented as: $\mathbf{T}_{b_i}^{w} = \begin{bmatrix} \mathbf{R}_{b_i}^{w} & \mathbf{t}_{b_i}^{w} \\ \mathbf{0}^T & 1 \end{bmatrix} \in SE(3)$, where $\mathbf{R}_{b_i}^{w} \in SO(3)$ is the rotation matrix and $\mathbf{t}_{b_i}^{w} \in \mathbb{R}^3$ is the translation vector. Given two data sequences $\mathcal{S}_1$ and $\mathcal{S}_2$, we

define their trajectories as $\mathbf{X}_1 = \{\mathbf{T}_1^{w_1}, \mathbf{T}_2^{w_1}, \ldots, \mathbf{T}_n^{w_1}\}$ and $\mathbf{X}_2 = \{\mathbf{T}_1^{w_2}, \mathbf{T}_2^{w_2}, \ldots, \mathbf{T}_m^{w_2}\}$, respectively. The corresponding point cloud frames are denoted as $\mathbf{P}_1 = \{{}^1P_1, {}^1P_2, \ldots, {}^1P_n\}$ and $\mathbf{P}_2 = \{{}^2P_1, {}^2P_2, \ldots, {}^2P_m\}$, where the left superscript indicates the sequence and the subscript represents the frame index. The merged pose graph is represented as $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, where $\mathbf{V} = \mathbf{X}_1 \cup \mathbf{X}_2$ is the set of pose nodes and $\mathbf{E}$ is the set of edges, including odometry edges and loop closure edges.

In multi-session mapping, we aim to incrementally expand the pose graph and construct a globally consistent map by optimizing the poses of $\mathcal{S}_2$ based on the existing pose graph of $\mathcal{S}_1$. Unlike recent works[4] that merge multiple data sequences simultaneously, our method incrementally merges two sequences at a time, simplifying the problem by focusing on pose graph scalability. We prioritize relative pose constraints[2] from LiDAR odometry and emphasize loop closure constraints to ensure global consistency and accuracy for multi-session data captured by sensors with similar noise levels over short periods. Given an initial guess $\mathbf{T}_{init}$ between $\mathcal{S}_1$ and $\mathcal{S}_2$, the problem can be formulated as estimating the optimal pose trajectory $\mathbf{X}_2$ that minimizes the overall error of the merged pose graph $\mathbf{G}$:

$$\arg\min_{\mathbf{X}_2} \sum_{i \in \mathcal{E}_{odo}^{\mathbf{G}}} \rho(\mathbf{e}_{odo_i}^{\mathbf{G}}) + \sum_{j \in \mathcal{E}_{lc}^{\mathbf{G}}} \rho(\mathbf{e}_{lc_j}^{\mathbf{G}}) + \sum_{k \in \mathcal{E}_{prior}^{\mathbf{G}}} \rho(\mathbf{e}_{prior_k}^{\mathbf{G}}), \tag{1}$$

where $\mathcal{E}_{odo}^{\mathbf{G}}$, $\mathcal{E}_{lc}^{\mathbf{G}}$, and $\mathcal{E}_{prior}^{\mathbf{G}}$ denote the sets of odometry edges, loop closure edges, and prior edges in the merged pose graph $\mathbf{G}$, respectively, $\mathbf{e}_{odo_i}^{\mathbf{G}}$, $\mathbf{e}_{lc_j}^{\mathbf{G}}$, and $\mathbf{e}_{prior_k}^{\mathbf{G}}$ are the corresponding edge errors, and $\rho(\cdot)$ is a robust kernel function.

III. Wasserstein-based Keyframe Selection

A. Keyframe Selection Formulation

We formulate the keyframe selection problem as a map distribution similarity task. Intuitively, if a point cloud frame P, when added to the map M, can bring significant map changes, it will be considered as a keyframe. Let M_1 and M_2 denote the map distributions before and after the update of the new frame, respectively. The keyframe selection problem can be transformed into measuring the dissimilarity between two distributions, defined as, $\text{KF} = \arg\max_P \textbf{Dis}(M_1, M_2)$ The map distribution can be modeled as a Gaussian Mixture Model: $M = \sum_{i=1}^{K} w_i \cdot \mathcal{N}(\mu_i, \Sigma_i)$, where K is the number of components, w_i is the weight of the i-th component, and $\mathcal{N}(\mu_i, \Sigma_i)$ is the Gaussian distribution with mean μ_i and covariance Σ_i. When a new point cloud frame P is available, it is first transformed into the map coordinate system using the estimated pose $T \in SE(3)$, $P_M = T \cdot P$. To ensure the reliability of the keyframe selection process, we make two assumptions. The pose T and the map M are accurately estimated, and the map M completely covers the range of the transformed frame P_M. The first assumption ensures a consistent scale between M_1 and M_2, while the second assumption avoids the interference of newly explored areas on the map information.

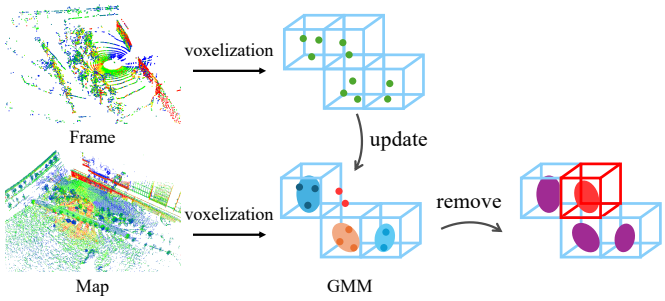

Fig. 2: Incremental update and maintenance of the GMM map. After initializing the GMM map, each new point cloud frame is transformed into the map coordinate using its pose. The Gaussian parameters of the voxels in the GMM are then updated point by point according to their indices. Voxels outside the radius range are removed. This process reflects the influence of the new frame on the overall map distribution.

B. Wasserstein Distance for Map Dissimilarity

To measure the dissimilarity between M_1 and M_2, we propose to use the Wasserstein distance, which is a theoretically optimal transport distance between two probability distributions. The L2 Wasserstein distance between two Gaussian distributions $\mathcal{N}_1(\mu_1, \Sigma_1)$ and $\mathcal{N}_2(\mu_2, \Sigma_2)$ is defined as:

$$\begin{aligned} W_2(\mathcal{N}_1, \mathcal{N}_2) = &\sqrt{|\mu_1 - \mu_2|_2^2 + } \\ &\text{tr}\left(\Sigma_1 + \Sigma_2 - 2\left(\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2}\right)^{1/2}\right) \end{aligned} \tag{2}$$

The first term, $|\mu_1 - \mu_2|_2^2$, represents the squared Euclidean distance between the means, capturing the spatial distance between the distribution centers and indicating the overall shift in the map distribution. The second term involves the trace of the covariance matrices and measures the dissimilarity in the shape and spread of the distributions, quantifying the difference in the local geometric structure of the map. This property makes the Wasserstein distance well-suited for detecting informative keyframes that contribute to the overall map quality. Figure 2 illustrates the incremental GMM map update process. The supporting materials[2] also provide a detailed mathematical description of the GMM update process and the algorithmic workflow of Wasserstein-based keyframe selection.

IV. Experiments

A. Hardware and Software Setup

To evaluate the proposed MS-mapping system, we utilize a diverse set of real-world datasets collected from both vehicle-mounted and handheld sensor setups. The primary datasets used are the FusionPortableV2 dataset[3] and our self-collected data from the HKUSTGZ campus. We have designed a custom sensor suite for data acquisition, which consists of a Pandar XT32 LiDAR with a measurement noise of approximately $\pm 1\,\text{cm}$ within a range of $40\,\text{m}$. The supporting materials also provide detailed information on the sensor suite

[2]https://github.com/JokerJohn/MS-Mapping

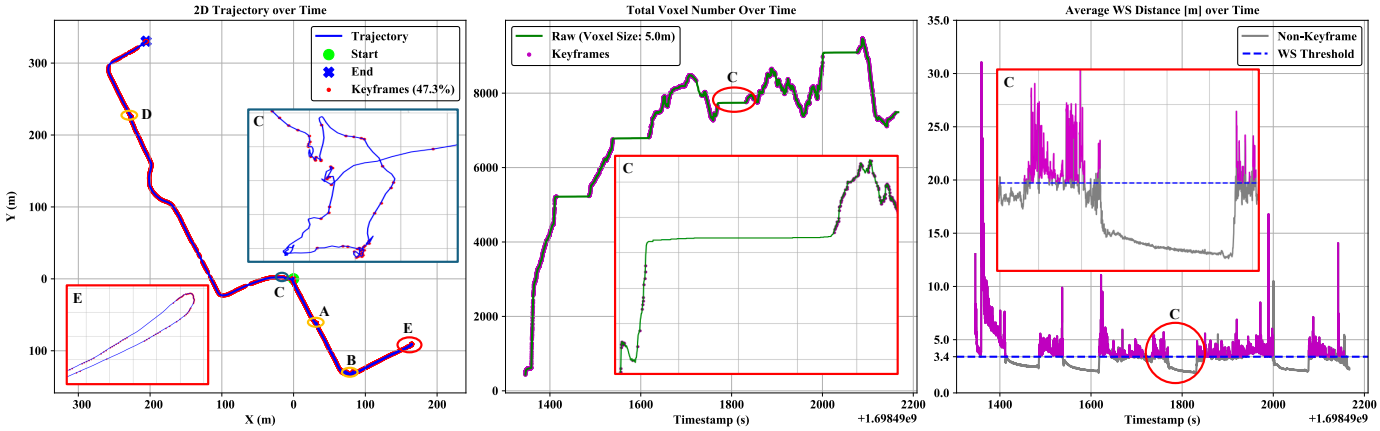

Fig. 3: Keyframe selection and temporal analysis on the RB2 dataset. **Left:** Original trajectory with selected keyframes (red). **Middle:** Voxel count over time, with keyframes marked in brown. **Right:** Average WS distance per frame, with keyframes exceeding the blue threshold line.

and the projection of dataset trajectories on Google Maps used in this study.

B. Baseline and Evaluation Metrics

We implemented the baseline algorithms combining odometry and traditional loop closure detection, based on the same principles as FAST_LIO_SAM[3] (FS) but with optimized performance. Additionally, we provide a manual alignment (MA) method using CloudCompare[4] software to register two dense maps, which is widely used in the surveying field but requires extensive manual adjustment. We evaluated localization accuracy using Absolute Trajectory Error (ATE) and assessed map accuracy using Accuracy (AC) and Chamfer Distance (CD) [2]. During the registration process, we set the KNN search distance to $1.0\,\mathrm{m}$ and consider point pairs with a distance smaller than $0.5\,\mathrm{m}$ as inliers. Furthermore, we utilize the Mean Map Entropy (MME) to evaluate the local quality of the generated maps.

C. Keyframe Selection Experiments

To evaluate our keyframe selection (WS) method, we compared it with the radius-based method, which selects keyframes based on translation or rotation thresholds (e.g., $0.1\,\mathrm{m}$ or $0.1\,\mathrm{rad}$), common in SLAM systems. Figure 4 shows that the WS method consistently outperforms the radius-based method in map accuracy on the RB2 dataset at the same keyframe rate. The Chamfer Distance (CD) validates the WS method's effectiveness in preserving essential map information. When the Keyframe Rate (KFR) is around 0.7, AC and CD converge, suggesting comprehensive scene description at this sparsity level. Visualization and temporal analysis on the RB2 dataset (Figure 3) further validate the WS method. The left plot shows the original trajectory with selected keyframes in red, achieving a KFR of 47.3%. Regions A-D, particularly region C where the robot is stationary, have fewer keyframes. Region E, a turning area, shows more frequent keyframe selection

[3]https://github.com/engcang/FAST-LIO-SAM
[4]https://www.danielgm.net/cc/

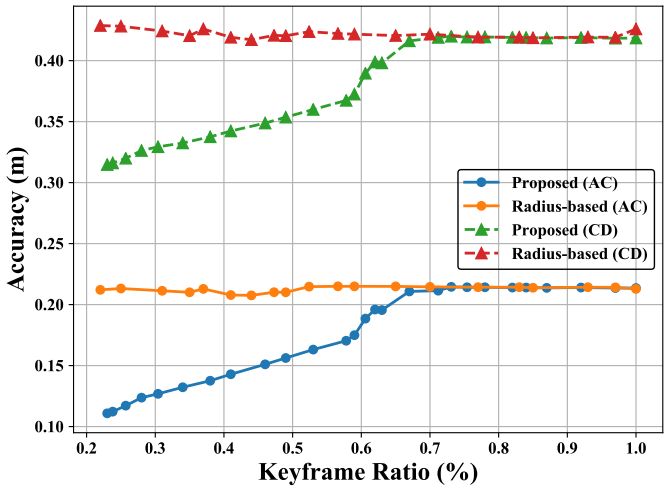

Fig. 4: Map accuracy comparison between the proposed and radius-based methods on the RB2 dataset across varying keyframe ratios.

due to significant map changes. The middle plot shows the voxel count over time, with keyframe instances marked in brown. Region C exhibits a nearly constant voxel count, indicating minimal map updates during stationary periods. Frequent keyframe selection before and after stationary regions suggests significant map updates during movement. The right plot presents the average WS distance for each frame, with keyframes identified when the WS distance exceeds the blue threshold line, and gray areas representing non-keyframes.

D. Overall Experiments

We conducted incremental mapping on the large-scale CP0 scene using two additional data sequences, comparing localization and map accuracy under different KFR to validate the proposed system's accuracy. Given the large scale of the CP0 base map and the partial coverage of the ground truth map, we evaluated only the covered portion, resulting in a relatively lower overall accuracy with an AC close to $40\,\mathrm{cm}$. Table I presents the results. On the RB2 dataset, the WS method shows superior localization and mapping accuracy compared to the

TABLE I: Performance comparison of multi-session mapping algorithms under different keyframe ratios (KFR).

Dataset	Description	Method	KFR (%)	Localization	Mapping		
				ATE (cm) ↓	AC (cm) ↓	CD (cm) ↓	MME ↓
CP0	Outdoor vehicle, 10.8 km, 6413 s	-	-	-	39.52	79.25	-6.87
RB2	Outdoor handheld, 1.06 km, 822 s	MA	-	-	21.39	37.83	-6.59
		FS	0-40	57.95	21.01	42.43	-6.92
			40-70	60.94	21.50	42.22	-6.92
			70-100	66.29	21.41	41.92	-6.94
		MS	0-40	**53.96**	**13.76**	**33.25**	**-6.95**
			40-70	59.97	17.49	37.25	-6.91
			70-100	59.78	19.59	39.88	-6.91
PK1	Degenerated parking lot, handheld, 0.97 km, 502 s	MA	-	-	21.70	42.94	-6.60
		FS	0-40	61.33	16.88	35.75	-6.91
			40-70	**57.52**	15.46	33.28	-6.90
			70-100	85.01	18.12	36.20	-6.93
		MS	0-40	76.12	**11.15**	**31.46**	**-7.11**
			40-70	77.12	15.29	33.47	-7.08
			70-100	85.01	17.09	36.55	-6.86

Bold values indicate the best accuracy, while underlined values represent the second-best accuracy.
ATE, CD, and AC are measured in centimeters (cm), while MME has no unit.

FS and MA methods, despite similar MME values. For the PK1 dataset, which includes regions with degraded LiDAR performance, the WS method exhibits relatively lower localization accuracy but outperforms the FS and MA methods in map accuracy and MME. The final map accuracy significantly surpasses that of the base map, indicating that multi-session mapping effectively uses new data to improve the accuracy of previously mapped areas. Figure 1 illustrates the incremental mapping on the RB02 dataset using PK01.

E. Run-time Analysis

Table II presents the quantitative results, showing the average performance consumption of each module tested on multiple datasets. The Graph Optimization (GO) module is the most time-consuming part of the system, even with the use of the incremental update algorithm ISAM2. When the KFR is low and the pose graph scale is small, the Loop Closure (LC) module consumes minimal time, running at a low frequency of 1 Hz in a separate thread. According to the table, for datasets of approximately 2 km, the MS method, despite taking nearly 30 ms to extract keyframes, shows much higher pose graph optimization efficiency compared to the FS method. This reflects the simplified complexity of graph optimization, with the system maintaining a frequency of around 12 Hz. Under similar KFR conditions, the FS method can only maintain a frequency close to 8 Hz.

TABLE II: Per-frame average execution time [ms] comparison of key modules on two datasets

Dataset	Method	KFR	KF ↓	LC ↓	GO ↓	Total ↓
RB2-PK1	MS1	40.04	32.47	0.16	36.75	80.28
	MS2	80.35	31.54	1.17	34.31	72.89
	FS1	45.20	0	1.77	133.84	134.01
	FS2	84.49	0	0.89	101.23	121.04

KFR represents the keyframe ratio multiplied by 100.
MS1 and MS2 represent map radius of 300 m and 200 m, respectively.
FS1 and FS2 adopt different distance thresholds.

V. Conclusion

We presented MS-Mapping, a novel multi-session LiDAR mapping system that addressed the challenges of data redundancy and pose graph scalability through an incremental mapping scheme and a real-time keyframe selection method. Experiments demonstrated the efficiency and accuracy of our approach. However, the keyframes selected by the Wasserstein distance may not be spatially uniform. Future work could investigate more advanced keyframe selection strategies to ensure better spatial distribution.

References

[1] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. *Georgia Institute of Technology, Tech. Rep*, 2:4, 2012.

[2] Xiangcheng Hu et al. Paloc: Advancing slam benchmarking with prior-assisted 6-dof trajectory generation and uncertainty estimation. *IEEE/ASME Transactions on Mechatronics*, pages 1–12, 2024. doi: 10.1109/TMECH.2024.3362902.

[3] Hexiang Wei et al. Fusionportablev2: A unified multi-sensor dataset for generalized slam across diverse platforms and scalable environments, 2024.

[4] Peng Yin et al. Automerge: A framework for map assembling and smoothing in city-scale environments. *IEEE Transactions on Robotics*, 39(5):3686–3704, 2023. doi: 10.1109/TRO.2023.3290448.

[5] Zuhao Zou et al. Lta-om: Long-term association lidar–imu odometry and mapping. *Journal of Field Robotics*, 2024.