2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Multi-LIO: A Lightweight Multiple LiDAR-Inertial Odometry System

Qi Chen†, Guanghao Li†, Xiangyang Xue, Jian Pu*

***Abstract*— The integration of multiple LiDAR sensors has the potential to significantly enhance odometry systems by providing comprehensive environmental measurements. However, current multiple LiDAR-inertial odometry frameworks face challenges in real-time processing due to the voluminous data generated. This paper introduces a real-time, computationally efficient multiple LiDAR-inertial odometry system (Multi-LIO) that outperforms existing state-of-the-art solutions in accuracy and scalability. Utilizing a novel parallel strategy for state updates and a voxelized map format, Multi-LIO optimizes computational efficiency. Furthermore, we introduce a point-wise uncertainty estimation method to augment the accuracy of scan-to-map registration, particularly in large-scale and complex scenarios. We validate our system's performance through extensive experiments on various challenging sequences. Multi-LIO emerges as a robust, scalable, and extensible solution, adaptable to various LiDAR configurations.**

I. INTRODUCTION

Over the past few decades, the advent of autonomous vehicles has led to the widespread use of mapping systems for constructing dense three-dimensional (3D) maps. Single LiDAR (-inertial) odometry systems such as LOAM [1], LIO [2], and LIO-SAM [3] provide accurate odometry results and 3D point cloud maps through multi-line mechanical LiDAR, which offers a 360-degree field of view (FoV), enabling the creation of dense 3D maps and accurate vehicle localization. However, the high cost and fragility of multi-line mechanical LiDAR have limited its widespread adoption in mapping systems. Some systems like Fast-LIO2 [4] have adapted to work with less expensive LiDARs with small FoV, such as solid-state LiDAR, though reducing hardware costs, often need to improve accuracy and stability due to the limited FoV. Multiple LiDAR systems could solve the FoV issue, though studies in this area are nascent and underexplored.

Addressing the FoV limitations and seeking to enhance odometry accuracy, our work explores multiple LiDAR setups. However, adapting the existing state-of-the-art odometry systems such as M-LOAM [5] and MA-LIO [6] to our 7-LiDAR system has revealed significant challenges in real-time processing and accuracy, with performance occasionally lagging behind single LiDAR systems like Fast-LIO2 [4]. We pinpoint one of the root causes of these issues as inefficient map formats. Specifically, we find that inefficiencies in map formats directly impact the odometry's effectiveness, while accurate uncertainty estimation in measurements, facilitated by an appropriate map format, can substantially elevate accuracy. This relationship underscores the necessity for a refined approach to manage map data and uncertainty estimation to overcome the prevalent limitations in multiple LiDAR odometry systems.

Moreover, the computational demands of processing extensive data from multiple LiDARs in real-time necessitate a reevaluation of system design, particularly the state update step in optimization-based odometry, which is notably time-consuming. One of the solutions is to employ geometry feature extraction from each scan, as demonstrated by M-LOAM [5]. The approach from M-LOAM reduced the number of measurements, but the feature extraction itself was computationally expensive. Given these considerations, we argue that treating all downsampled LiDAR points as surface features as [4] and parallel all the time-consuming computation offer a more viable solution for developing efficient multiple LiDAR odometry systems.

To address these challenges, we introduce Multi-LIO, a lightweight odometry system that extends the work presented in [7]. Our system tightly couples multiple LiDARs with an Inertial Measurement Unit (IMU) to provide robust and efficient odometry. Remarkably, Multi-LIO can estimate the point-wise uncertainty during scan-to-map registration and significantly increase the accuracy of odometry, which allows the system to construct an accurate 3D dense map of a complex campus environment, covering an area of approximately $1,000,000\,m^2$ through 6 Livox Horizon LiDARs and 1 Livox Tele LiDAR in real-time, achieved without the need for loop-closure strategies or Global Positioning System (GPS) support. Our main contributions are the following:

- We introduce Multi-LIO, a low-latency system that mitigates computational load in LiDAR-inertial odometry. Optimized for real-time 3D mapping, it supports diverse LiDAR setups and has shown reduced drift error in large, complex scenes through rigorous experiments.
- Our Gaussian voxel map, built upon voxelized Gaussian distributions, integrates seamlessly with an efficient method for point-wise uncertainty estimation in the scan-to-map registration process. This cohesive framework significantly improves registration accuracy, particularly in the complex landscapes of large-scale environments where managing uncertainty is crucial.
- We develop a parallelization strategy incorporated within the iterated error state Kalman filter, significantly accelerating the system's state updates. This enhancement is instrumental in achieving efficient mapping in expansive, large-scale environments.

* Corresponding author
† These two authors.contribute equally to this work
All authors are with Fudan University, Shanghai 200433, China {qichen21, ghli22}@m.fudan.edu.cn, {xyxue, jianpu}@fudan.edu.cn.

II. Related Works

A. Map format of LiDAR (-Inertial) Odometry

The choice of map format in an odometry system can critically influence its overall performance. Systems such as Lego-LOAM [8], LIO-SAM [3], LINS [9], Fast-LIO [10], and DLIO [11] employed keyframes stored in memory, this approach may necessitates the continual construction of new KD-trees for nearest neighbor search, which could be computationally demanding. SUMA++ [12] and SUMA [13] leveraged surfel-based maps to achieve accurate odometry in expansive urban environments. Fast-LIO2 [4], Swarm-LIO [14] utilized an incremental KD-tree (ikd-tree) [15] for efficient nearest-neighbor searching and real-time map updating, effectively addressed the imbalance issue associated with incremental point updated to the KD-tree. R3live [16] presented a voxel-point map format and achieved real-time performance in multimodal mapping. To accelerate the nearest neighbor search and map update processes, Faster-LIO [7] proposed iVox and iVox-PHC, voxel-based maps implemented by hash tables. LiTAMIN [17] and LiTAMIN2 [18] demonstrated the computational efficiency of Normal Distributions Transform (NDT) registration when using a map composed of ellipsoids. Another innovative approach was the adaptive-size voxel map from [19], which not only stored plane information but also offered a method to estimate the uncertainty of observations. Last but not least, SLAMesh [20] introduced a mesh-based map format that significantly reduced memory consumption while enhancing the efficiency of nearest-neighbor searches through its mesh search algorithm.

B. Multiple LiDAR (-Inertial) Odometry

Odometry systems relying on a single LiDAR often grapple with substantial drift due to a limited FoV. To mitigate this, M-LOAM [5] employed a tightly coupled multiple LiDAR odometry system to calibrate extrinsic parameters and generate accurate odometry results. Compared with the M-LOAM, MILIOM [21] enhanced odometry accuracy by tightly coupling an IMU with multiple LiDARs, showing improvements. SLICT [22] introduced an octree-based global map of multi-scale surfels and a corresponding optimization method to elevate the mapping system's efficiency and accuracy. RailLoMer [23], GM-Livox [24], and M-LIO [25] adopted tightly-coupled approach, integrating multiple LiDARs with an Inertial Navigation System (INS) to deliver precise odometry, particularly in degenerated and large-scale environments. However, these optimization-based methods often come at the cost of computational efficiency. Addressing this, MA-LIO [6] offered a real-time, filter-based system that focused on the temporal and spatial discrepancies inherent in multiple LiDAR-inertial setups. This framework also estimated real-time point-wise uncertainty, enhancing odometry accuracy in complex urban landscapes.

III. Method

Our system is a tightly coupled multiple LiDAR-inertial odometry framework. As depicted in Fig.1, the architecture

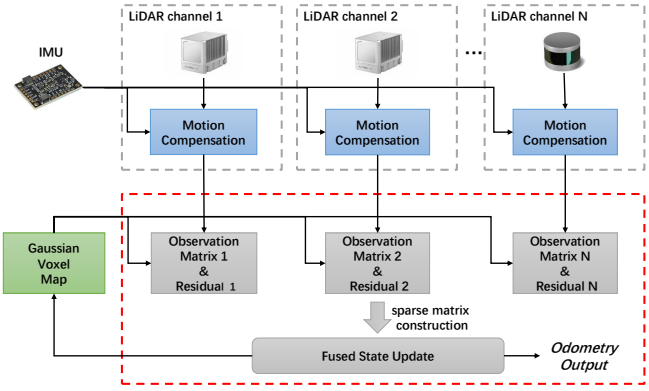


Fig. 1: An overview of Multi-LIO, a filter-based multiple LiDAR-inertial odometry system derived from [4]. Utilizing our novel decentralized Extended Kalman Filter, the system seamlessly integrates multiple LiDAR scans with a singular IMU to generate stable, unified, precise odometry results.

comprises multiple LiDAR channels, a Gaussian voxel map, a motion compensation module, and a state fusion update module. Their respective roles are as follows:

- LiDAR channels: Each LiDAR channel handles the preprocessing for a single LiDAR, which aligns with the Fast-LIO2 [4] methodology.
- Gaussian voxel map: This module primarily handles the nearest-neighbor search based on hash functions and the real-time updating of the voxel map.
- Motion compensation module: This component integrates acceleration and angular velocity from IMU data and predicts prior poses at each IMU timestamp for LiDAR scan motion compensation.
- State fusion update module: While receiving the scans from the IMU processor, the State Fusion Update Module performs scan-to-map registration and updates the state by applying paralleled iterated error state Kalman filter.

A. Notation and State Model

To rigorously elucidate the mathematical underpinnings of our proposed techniques, we employ the operations $\boxplus$ and $\boxminus$ to represent operation on a manifold $\mathcal{M}$, as delineated in [4].

Our system consists of a state variable $\boldsymbol{x}$ and its corresponding covariance $\boldsymbol{P}$. The IMU frame, denoted as I, serves as the body frame, while the first body frame treats the global frame, denoted as G. The state variable $\boldsymbol{x}$ is defined as follows:

$$\boldsymbol{x} \triangleq [{}^{G}\boldsymbol{R}_I^T \; {}^{G}\boldsymbol{p}_I^T \; {}^{G}\boldsymbol{R}_{L_n}^T \; {}^{G}\boldsymbol{p}_{L_n}^T \; \dots \; {}^{G}\boldsymbol{v}^T \; \boldsymbol{b}_g^T \; \boldsymbol{b}_a^T \; {}^{G}\boldsymbol{g}^T]^T \in \mathcal{M}, \tag{1}$$

here in state variable $\boldsymbol{x}$, $\boldsymbol{T}_I = [{}^{G}\boldsymbol{R}_I, {}^{G}\boldsymbol{p}_I]$ and $\boldsymbol{T}_{L_n} = [{}^{G}\boldsymbol{R}_{L_n}, {}^{G}\boldsymbol{p}_{L_n}]$ represent the transformation from the body frame or the $(n)^{th}$ LiDAR frame to the global frame. The terms ${}^{G}\boldsymbol{v}$, ${}^{G}\boldsymbol{g}$, and ${}^{G}\boldsymbol{b}_{a/g}$ denote the velocity, gravity, and bias in the body frame, respectively.

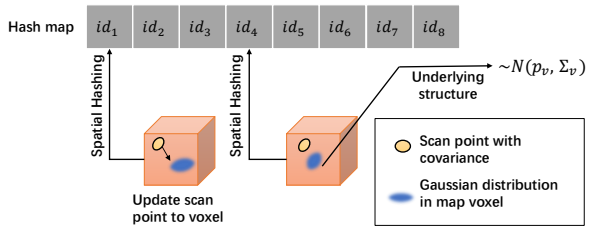
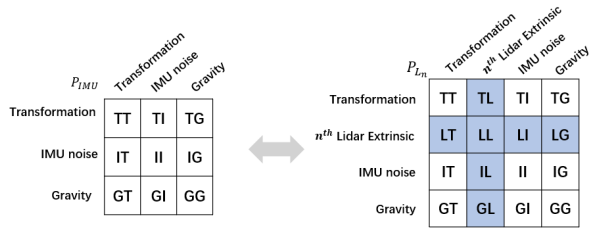
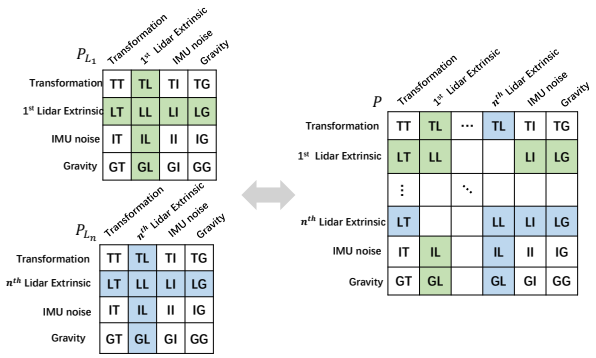


Fig. 2: Map structure of Gaussian voxel map. It employs an unordered map with a hash function for efficient storage and retrieval of voxels.



(a) Bidirectional conversion between $\boldsymbol{P}_{IMU}$ and $\boldsymbol{P}_{L_n}$



(b) Bidirectional conversion between $\boldsymbol{P}$ and $\boldsymbol{P}_{L_n}$

Fig. 3: Covariance conversion method between sparse covariance matrices, where matrices share the values in the same color and name blocks. (a) The conversion between the covariance matrix $\boldsymbol{P}_{IMU}$ saved in the IMU processor and the covariance matrix $\boldsymbol{P}_{L_n}$ saved in $(n)^{th}$ LiDAR channel. (b) Bidirectional conversion LiDAR's covariance matrices and the fused covariance matrix P. The fused matrix is then utilized to refresh the covariance matrices of the preceding variables.

B. Gaussian voxel map

1) Map Structure: As depicted in Fig.2, we employ an unordered map with a hash function for efficient voxel storage and retrieval. Furthermore, we store a Gaussian distribution within each voxel, characterized by its mean $\boldsymbol{p}_v$, covariance matrix $\boldsymbol{\Sigma}_v$, and the number of points n_v updated to the voxel. This approach eliminates the need for iterative point searching within the voxel during nearest-neighbor queries.

2) Map Update: When updating a point characterized by its mean $\boldsymbol{p}_{\text{new}}$ and covariance matrix $\boldsymbol{\Sigma}_{\text{new}}$ to the map, where $\boldsymbol{\Sigma}_{\text{new}}$ is estimated using the method in [19], the Gaussian voxel map first calculates the hash key corresponding to $\boldsymbol{p}_{\text{new}}$. It then locates the relevant voxel and updates its $\boldsymbol{p}_v$ and $\boldsymbol{\Sigma}_v$ as outlined in:

$$\begin{aligned} \boldsymbol{p}_v^{\text{new}} &= \frac{n_v \boldsymbol{p}_v + \boldsymbol{p}_{\text{new}}}{n_v + 1}, \\ \boldsymbol{\Sigma}_v^{\text{new}} &= \frac{n_v(\boldsymbol{\Sigma}_v + \boldsymbol{\delta p}_v \boldsymbol{\delta p}_v^T) + (\boldsymbol{\Sigma}_{\text{new}} + \boldsymbol{\delta p}_{\text{new}} \boldsymbol{\delta p}_{\text{new}}^T)}{n_v + 1}, \end{aligned} \tag{2}$$

where $\boldsymbol{\delta p}_v = \boldsymbol{p}_v - \boldsymbol{p}_v^{\text{new}}$ and $\boldsymbol{\delta p}_{\text{new}} = \boldsymbol{p}_{\text{new}} - \boldsymbol{p}_v^{\text{new}}$. In contrast to the original iVox map update procedure, our proposed method only necessitates updating the voxel's mean and covariance. This results in a reduced RAM requirement compared to both ikdtree and iVox.

C. Motion Compensation

In our motion compensation module, we compute only the partial state variable $\boldsymbol{x}_{IMU}$ related to our IMU, excluding the LiDAR's external extrinsic from the initial state variable $\boldsymbol{x}$. The discrete state transition model at the $(i)^{th}$ IMU measurement is defined as:

$$\boldsymbol{x}_{IMU_{i+1}} = \boldsymbol{x}_{IMU_i} \boxplus (\Delta t f(\boldsymbol{x}_{IMU_i}, \boldsymbol{u}_i, \boldsymbol{w}_i)), \tag{3}$$

where Δt, $\boldsymbol{u}_i$, $\boldsymbol{w}_i$ represent the sample period, input, and noise of our IMU measurement, respectively. $f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})$ from [4] defines to the time derivative of each element in $\boldsymbol{x}_{IMU}$. During the time interval between the $(k)^{th}$ and $(k-1)^{th}$ LiDAR scans, the IMU is utilized to predict the system's state. Given the $(i)^{th}$ measurement u_i and the $(i)^{th}$ prior state $\hat{\boldsymbol{x}}_{IMU_i}$ along with its covariance $\hat{\boldsymbol{P}}_{IMU_i}$, the subsequent prior state $\hat{\boldsymbol{x}}_{IMU_{i+1}}$ and its covariance $\hat{\boldsymbol{P}}_{IMU_{i+1}}$ can be computed as follows:

$$\begin{aligned} \hat{\boldsymbol{x}}_{IMU_{i+1}} &= \hat{\boldsymbol{x}}_{IMU_i} \boxplus (\Delta t f(\hat{\boldsymbol{x}}_{IMU_i}, \boldsymbol{u}_i, 0)), \\ \hat{\boldsymbol{P}}_{IMU_{i+1}} &= \boldsymbol{F}_{\tilde{x}_{IMU_i}} \hat{\boldsymbol{P}}_{IMU_i} \boldsymbol{F}_{\tilde{x}_{IMU_i}}^T + \boldsymbol{F}_{w_i} \boldsymbol{Q}_i \boldsymbol{F}_{w_i}^T, \end{aligned} \tag{4}$$

where $\hat{\boldsymbol{P}}_{IMU_0} = \bar{\boldsymbol{P}}_{IMU_{k-1}}$ and $\hat{\boldsymbol{x}}_{IMU_0} = \bar{\boldsymbol{x}}_{IMU_{k-1}}$. $\boldsymbol{Q}_i$ signifies the covariance of $\boldsymbol{w}_i$, while the Jacobians $\boldsymbol{F}_{\tilde{x}_{IMU_i}}$ and $\boldsymbol{F}_{w_i}$ denote the derivatives of $\delta(\boldsymbol{x}_{IMU_{i+1}} \boxminus \hat{\boldsymbol{x}}_{IMU_{i+1}})$ with respect to each subscript.

After the IMU has predicted and recorded all prior states, we proceed to update the covariance matrices $\boldsymbol{P}_{L_n}$ stored in the $(n)^{th}$ LiDAR channel, which is accomplished by employing the most recent covariance matrix $\boldsymbol{P}_{IMU}$, as illustrated in Fig.3(a).

D. State Fusion Update

To refine the odometry state at the $(k)^{th}$ time step, we reformulate the state estimation problem as a Maximum A Posteriori (MAP) problem, following the approach described in [4]. The optimization problem is:

$$\min_{\tilde{\boldsymbol{x}}_k^\kappa} \quad \|\boldsymbol{x}_k \boxminus \hat{\boldsymbol{x}}_k\|^2_{\hat{\boldsymbol{P}}_k^{-1}} + \sum_{j=1}^{m} \left\| \boldsymbol{z}_{k,j}^\kappa + \boldsymbol{H}_{k,j}^\kappa \tilde{\boldsymbol{x}}_k^\kappa \right\|^2_{\boldsymbol{R}_{k,j}^{-1}}, \tag{5}$$

here, $\boldsymbol{z}_k = [\boldsymbol{z}_{k,1}^\kappa, \ldots, \boldsymbol{z}_{k,m}^\kappa]$ represents the point-to-plane residuals. The Jacobian matrix $\boldsymbol{H}_k = [\boldsymbol{H}_{k,1}^{\kappa T}, \ldots, \boldsymbol{H}_{k,n}^{\kappa T}]$ is associated with $\boldsymbol{z}_k$ with respect to $\tilde{\boldsymbol{x}}_k^\kappa$. The fused covariance matrix $\boldsymbol{P}_k$ is constructed as described in Fig.3(b). The uncertainty of all measurements is denoted as $\boldsymbol{R}_k = \text{diag}(\boldsymbol{R}_{k,1}, \ldots, \boldsymbol{R}_{k,m})$.

1) Uncertainty of measurement model: To enhance the performance of odometry, accurately estimating the covariance matrix $\boldsymbol{R}_k$ is crucial. Initially, we perform scan-to-map matching by transforming each point to the map and searching for the nearest five voxels, as described in [7]. To compute a specific $\boldsymbol{R}_{k,j}$, we adopt the approach from [19] and make certain approximations for computational efficiency. We first fuse the Gaussian distributions stored in each voxel to obtain a fused Gaussian distribution with mean $\boldsymbol{p}_{k,j}$ and covariance matrix $\boldsymbol{\Sigma}_{k,j}$. We then apply the least squares method to the centers of these five voxels to estimate a plane characterized by its normal vector $\boldsymbol{n}_{k,j}$ and the center of the plane $\boldsymbol{q}_{k,j} = \boldsymbol{p}_{k,j}$. The covariance of this plane, $\boldsymbol{\Sigma}_{n_{k,j},q_{k,j}}$, is approximated as $\boldsymbol{\Sigma}_{k,j}$. Following [19], the uncertainty $\boldsymbol{R}_{k,j}$ for the $(j)^{th}$ matching pair is defined as:

$$\begin{aligned} \boldsymbol{R}_{k,j} &= \boldsymbol{J}_{k,j} \boldsymbol{\Sigma}_{\boldsymbol{n}_{k,j},\boldsymbol{q}_{k,j},\boldsymbol{p}_j^L} \boldsymbol{J}_{k,j}^T, \\ \boldsymbol{\Sigma}_{n_{k,j},\boldsymbol{q}_{k,j},\boldsymbol{p}_j^L} &= \begin{bmatrix} \boldsymbol{\Sigma}_{k,j} & 0 \\ 0 & \boldsymbol{\Sigma}_{p_j^L} \end{bmatrix}, \end{aligned} \tag{6}$$

here, $\boldsymbol{p}_j^L$ is the LiDAR point, and its corresponding covariance $\boldsymbol{\Sigma}_{p_j^L}$ is calculated as per [19]. The Jacobian matrix $\boldsymbol{J}_{k,j}$ is with respect to the point-to-plane residual $\boldsymbol{z}_{k,j}$ and is a function of $\boldsymbol{n}_{k,j}$, $\boldsymbol{q}_{k,j}$, and $\boldsymbol{p}_j^L$.

2) Parallel LiDAR measurements update: Consider a system equipped with n LiDARs, where the total number of measurements from all LiDAR scans is m. To optimize the $(\kappa+1)^{th}$ state of the odometry, the system employs the paralleled iterated error state Kalman filter algorithm as described in [4] to solve Equation 5. The state update equations are given by:

$$\begin{aligned} \boldsymbol{K}_k^\kappa &= \left(\boldsymbol{H}_k^{\kappa T} \boldsymbol{R}_k^{-1} \boldsymbol{H}_k^\kappa + (\boldsymbol{J}^\kappa)^T \hat{\boldsymbol{P}}_k^{-1} \boldsymbol{J}^\kappa \right)^{-1} \boldsymbol{H}_k^{\kappa T} \boldsymbol{R}_k^{-1}, \\ \hat{\boldsymbol{x}}_k^{\kappa+1} &= \hat{\boldsymbol{x}}_k^\kappa \boxplus \left(-\boldsymbol{K}_k^\kappa \boldsymbol{z}_k^\kappa - (\boldsymbol{I} - \boldsymbol{K}_k^\kappa \boldsymbol{H}_k^\kappa)(\boldsymbol{J}^\kappa)^{-1} (\hat{\boldsymbol{x}}_k^\kappa \boxminus \hat{\boldsymbol{x}}_k) \right). \end{aligned} \tag{7}$$

During the implementation of Equation 7, we find that directly calculating $\boldsymbol{H}_k^{\kappa T} \boldsymbol{R}_k^{-1} \boldsymbol{H}_f^\kappa$ using dense matrix methods is computationally expensive, which is because the dimension of $\boldsymbol{H}_k^\kappa$ is $m(6n+18)$, implying that increasing the number of LiDARs not only augments the dimension of the measurements m but also the dimension of the states that need to be updated.

We apply a parallel computing pipeline to address the computational bottleneck as illustrated in Fig.4. This pipeline parallelize the computation of the sparse matrices $\boldsymbol{z}_k^\kappa$, $\boldsymbol{H}_k^\kappa$, $\boldsymbol{H}_k^{\kappa T} \boldsymbol{R}_k^{-1} \boldsymbol{H}_k^\kappa$, and $\boldsymbol{x}_k$.

The above process keeps repeating until convergence is achieved, i.e., $\| \hat{\boldsymbol{x}}_k^\kappa \boxminus \hat{\boldsymbol{x}}_k^{\kappa+1} \| < \epsilon$. Upon convergence, the optimal fused state and fused covariance estimates are given by:

$$\overline{\boldsymbol{x}}_k = \hat{\boldsymbol{x}}_k^{\kappa+1}, \quad \overline{\boldsymbol{P}}_k = (\boldsymbol{I} - \boldsymbol{K}_k^\kappa \boldsymbol{H}_k^\kappa)(\boldsymbol{J}^\kappa)^{-1} \hat{\boldsymbol{P}}_k (\boldsymbol{J}^\kappa)^{-T}. \tag{8}$$

After obtaining the fused state and covariance matrix, the system updates each LiDAR channel's covariance matrix $\boldsymbol{P}_{L_n}$ and $\boldsymbol{P}_{IMU}$ stored in the IMU processor, as depicted in Fig.3. The final step in one iteration of the odometry process is to update all LiDAR points and their corresponding covariances to the map. Given a point $\boldsymbol{p}_j$ and its covariance $\boldsymbol{\Sigma}_j$ from the $(n)^{th}$ LiDAR in the $(k)^{th}$ state. We first transform $\boldsymbol{p}_j$ and $\boldsymbol{\Sigma}_j$ to global frame refer to [19], then we use Eq.2 to update the point and covariance to the map.

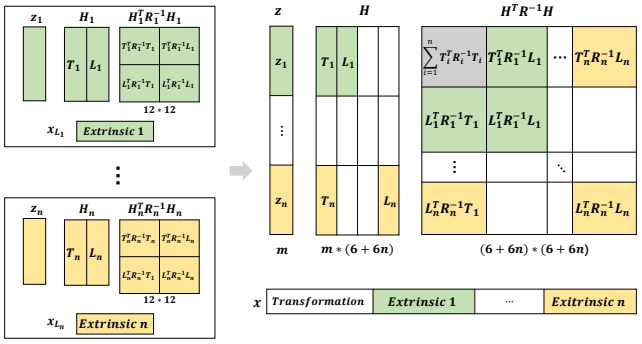


Fig. 4: The parallel computation pipeline for sparse matrices $\boldsymbol{z}, \boldsymbol{H}, \boldsymbol{H}^T \boldsymbol{R}^{-1} \boldsymbol{H}$, and $\boldsymbol{x}$. In the $(n)^{th}$ channel, $\boldsymbol{z}_k, \boldsymbol{H}_k$, and $\boldsymbol{H}_k^T \boldsymbol{R}_k^{-1} \boldsymbol{H}_k$ are first computed. $\boldsymbol{T}_n$ and $\boldsymbol{L}_n$ represent derivatives of $\boldsymbol{z}_n$ related to body frame transformation and $(n)^{th}$ LiDAR-to-body frame extrinsics, respectively. Sparse matrices for state updates are generated, excluding IMU-related state variables as their derivatives are zero.

TABLE I: Dataset description

Dataset	LiDAR Information	IMU	Environment
Hilti	1 OS0-64 1 Livox Horizon	200Hz	Indoor & Outdoor
City01-03	1 Livox Avia 1 Livox Tele 1 OS2-128	100Hz	Urban
Ours	6 Livox Horizon 1 Livox Tele	200Hz	Campus

IV. EXPERIMENTS

A. Dataset and Evaluation

1) Dataset Overview: To rigorously assess our system's capabilities, we undertake experiments across three distinct datasets: the Hilti SLAM Dataset 2021 [26], City01-03 [6], and our private dataset. Our dataset allows for a thorough evaluation of the system's stability and precision. Tab.I provides comprehensive details about these datasets.

2) Evaluation: We deploy our system on a 16GB RAM computer with an AMD Ryzen 5 3600X CPU and set the resolution of our Gaussian voxel map to 0.5m. We compare our system against multiple state-of-the-art algorithms configured per the guidelines outlined in [6] for benchmarking purposes. We employ EVO [27] as our evaluation tool to calculate the Root Mean Square Error ($RMSE$) of the Absolute Trajectory Error (ATE), which furnishes a quantitative comparison of system accuracy. The ATE_t is denoted as the translation part of ATE while ATE_r is the rotation part of ATE.

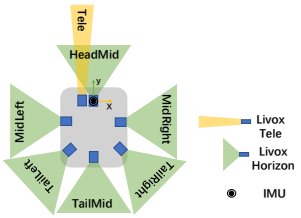
B. Results

1) Hilti SLAM Dataset 2021: Tab.II delineates the Absolute Trajectory Translation Error (ATE_t) results derived

TABLE II: ATE results for the public dataset, the best results are in **bold** and the second-best are in underline.

	Hilti SLAM 2021					City01-03					
	Basement	Campus	Construction	Lab	UZH	City01		City02		City03	
	ATE_t	ATE_t	ATE_t	ATE_t	ATE_t	ATE_t	ATE_r	ATE_t	ATE_r	ATE_t	ATE_r
M-LOAM [5]	0.115	0.386	2.647	0.064	0.276	33.907	8.792	72.382	4.683	33.801	6.657
LOCUS 2.0 [28]	0.120	0.087	0.290	0.040	0.177	23.998	5.521	58.211	4.722	21.753	4.773
Fast-LIO2 [4] (Ouster)	0.046	0.063	0.088	0.026	0.184	9.970	4.575	35.308	7.473	6.951	4.194
MA-LIO [6]	0.036	0.046	0.063	0.024	0.177	6.538	3.491	6.707	3.522	5.470	3.522
Ours (w/o uncertainty)	0.017	0.040	0.063	0.100	0.183	9.074	2.071	7.937	2.392	7.740	2.802
Ours (full)	**0.012**	**0.036**	**0.046**	**0.020**	**0.173**	**4.682**	**2.067**	**3.984**	**2.392**	**5.114**	**2.800**

(a) Platform



(b) Configuration of LiDARs

Fig. 5: The platform of our system. (a) The vehicle we used to deploy our system. All sensors are installed on the vehicle's top and calibrated by MLCC [29]. (b) Our system's hardware configuration consists of 6 Livox Horizon and 1 Livox Tele. Here, we use the IMU inside HeadMid as IMU input.

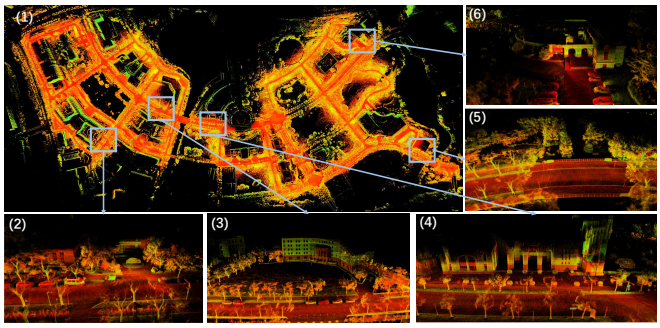
Fig. 6: Trajectories of our private dataset. The dataset consists of three scenarios: *Campus01*, *Campus02*, and *Campus03*. The *Campus01* trajectory has a length of 2 kilometers, the *Campus02* trajectory spans 7.9 kilometers, and the *Campus03* trajectory measures 704 meters.

from the Hilti dataset. Among all evaluated odometry systems, our Multi-LIO system achieve the best performance among the presented methods. The experiments conducted on the Hilti dataset substantiate that our system adeptly handles degenerate cases commonly encountered in confined indoor settings, such as laboratories and basements. Simultaneously, the system maintains high accuracy in scenarios characterized by more dynamic movements, as evidenced by mapping results of *Construct* and *UZH*.

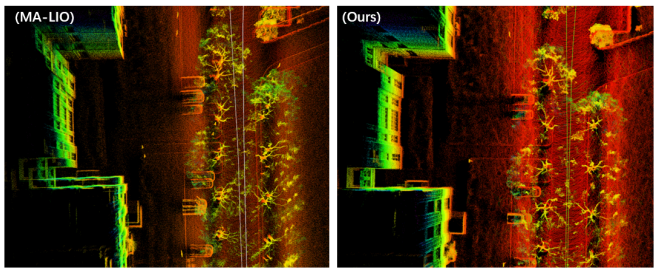
2) City01-03 Dataset: City01-03 dataset from [6] poses a complex challenge: LiDARs are mounted atop a vehicle that traverses the environment at high speeds, and the recorded scenarios are replete with dynamic objects and degenerate zones, including tunnels. Tab.II presents the results for the Absolute Trajectory Translation Error (ATE_t) gleaned from the City01-03 dataset. Our system outperforms other state-of-the-art methods in large and complex scenarios.

TABLE III: ATE_t results for private dataset.

	Campus01 (2.0km)	Campus02 (7.9km)	Campus03 (704m)
Fast-LIO2(HeadMid)	6.851	12.150	2.059
MA-LIO M3	5.023	5.558	2.060
Ours M3 (w/o uncertainty)	5.456	3.060	7.740
Ours M3 (full)	4.566	2.824	2.051
Ours M7 (full)	**2.132**	**2.050**	**1.752**



(a) Mapping result of Multi-LIO (ours)



(b) Comparison between MA-LIO and Multi-LIO (ours)

Fig. 7: Mapping results of Campus02. (a) The point cloud map produced by our system, (1) is the whole point cloud, (2)-(6) are the details of the point cloud. (b) The mapping comparison result between [6] and our system.

3) Private Dataset: Our dataset utilizes only Livox LiDARs mounted atop the vehicle as depicted in Fig. 5. Our private dataset has three distinct records. *Campus01* serves as a relatively brief record without loop closure to evaluate the drift error inherent in the odometry. *Campus02* is recorded for mapping the entire campus, allowing for a comprehensive assessment of the system's accuracy and stability. Lastly, *Campus03* is a record containing several degraded areas to test the system's performance in degenerate dynamic scenarios. Fig.6 shows the satellite trajectory maps for the three scenarios.

For the experiments conducted on our private dataset, we

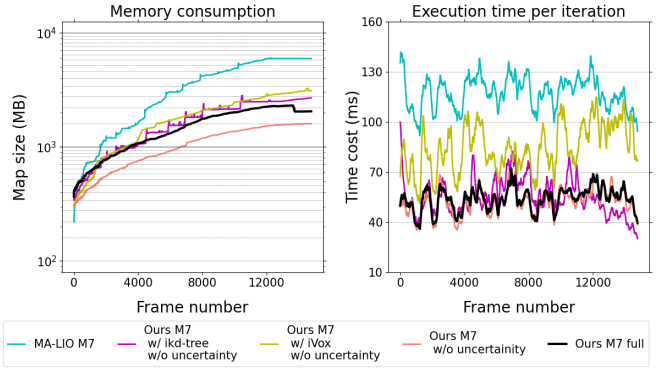


Fig. 8: Comparison of time and memory usage across methods. The left plot shows memory usage while the right plot displays runtime.

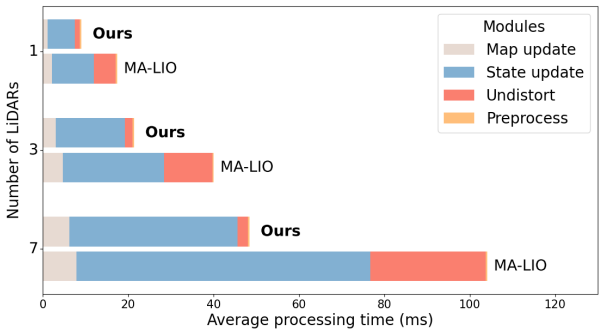


Fig. 9: Comparison of modular execution time per frame for different methods under varying numbers of LiDARs.

employ two multiple LiDAR hardware configurations, M3 and M7. M3 is for a fair comparison that uses 3 Livox LiDARs (HeadMid, MidLeft, MidRight in Fig.5(b)). In M7, we use all available LiDARs mounted on the vehicle to test our system's efficiency rigorously. We generate the 3-DOF ground truth for the dataset via GPS, and the mapping results are presented in Fig.7 and Tab.III, which indicates our method has the best performance.

C. Efficiency of System

To substantiate the efficiency of our Multi-LIO system, we focus our evaluation on our longest record, *Campus02*, which comprises 14,792 frames per LiDAR, and all experiments are conducted on the same device with the same configuration. The left plot of Fig.8 illustrates the map sizes associated with various prevalent map-saving methodologies employed in odometry systems, which shows our method costs less memory than ikd-tree [15] and iVox [7].

The right plot of Fig.8 delineates the time expenditure per iteration for the *Campus02*. Owing to the utilization of a voxel-based map, the time cost of Multi-LIO does not increase as the scenario scale increases. In addition to the above, we have conducted a detailed analysis of the modular execution time per frame for different methods under varying numbers of LiDARs. Fig.9 presents the results of running time. Remarkably, our system can run at 50Hz when configured with three Livox Horizon LiDARs and at 20Hz with seven Livox LiDARs. Furthermore, to gauge the efficiency of Multi-LIO in handling mechanical LiDARs, we also assess the time cost per frame on the *City01*, which consists of 13,089 frames per LiDAR. Our system demonstrates an average time cost of 40ms per frame, which is approximately twice as fast as MA-LIO [6], underscores the superior efficiency of our system.

D. Ablation Study

We conduct an ablation study to assess the efficacy of our proposed methodologies. Results presented in Tab.II and Tab.III underscore a pivotal observation: when Multi-LIO operates without the feature of uncertainty estimation, opting instead for a constant-value uncertainty (as shown in the rows "w/o uncertainty"), the system experiences a marked degradation in odometry accuracy, particularly in large-scale or degenerated scenarios such as City01-03. This empirical evidence accentuates the indispensable role that our proposed uncertainty estimation technique plays in fortifying the robustness of the odometry calculations. Also, the right plot in Fig.8 demonstrates that our uncertainty estimation process does not consume excessive time, thereby maintaining the system's overall efficiency.

Additionally, our Gaussian voxel map substantially contributes to the system's computational efficiency. As evidenced in the left plot of Fig.8, our map representation tends to be more memory-efficient than other efficient map types. Furthermore, Fig.9 illustrates that the Gaussian voxel map considerably expedites the nearest-neighbor search process, particularly when juxtaposed with alternative map representations.

V. CONCLUSIONS

This paper introduces the Multi-LIO, a streamlined, multiple LiDAR-inertial odometry system designed for robust stability and pinpoint accuracy in complex, large-scale outdoor campuses and intricate urban environments. Our comprehensive experimental results demonstrate that Multi-LIO can map expansive areas with a significantly reduced drift error and a more efficient processing time, thereby outperforming existing multiple LiDAR-inertial odometry systems. Bolstered by our innovative parallel computational strategy implemented in the paralleled iterated error state Kalman filter, our system exhibits scalability and flexibility, effortlessly accommodating various hardware configurations. The distinguishing cornerstone of Multi-LIO lies in its advanced map representation techniques coupled with a meticulously designed uncertainty estimation methodology specifically for scan-to-map registration. As part of our future research endeavors, we aim to augment Multi-LIO by integrating an array of cameras, enhancing the readability and richness of the point cloud map.

ACKNOWLEDGMENT

This work was partly supported by the Shanghai Research and Innovation Functional Program under Grant 17DZ2260900, Shanghai Municipal Science and Technology Major Project(19511120700, 2018SHZDZX01), and ZJLab.

REFERENCES

[1] 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.

[2] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3d lidar inertial odometry and mapping," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3144–3150.

[3] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and R. Daniela, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2020, pp. 5135–5142.

[4] 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.

[5] J. Jiao, H. Ye, Y. Zhu, and M. Liu, "Robust odometry and mapping for multi-lidar systems with online extrinsic calibration," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 351–371, 2021.

[6] M. Jung, S. Jung, and A. Kim, "Asynchronous multiple lidar-inertial odometry using point-wise inter-lidar uncertainty propagation," *IEEE Robotics and Automation Letters*, 2023.

[7] 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.

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

[9] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, "Lins: A lidar-inertial state estimator for robust and efficient navigation," in *IEEE International Conference on Robotics and Automation*. IEEE, 2020, pp. 8899–8906.

[10] 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.

[11] 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.

[12] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma++: Efficient lidar-based semantic slam," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4530–4537.

[13] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments." in *Robotics: Science and Systems*, vol. 2018, 2018, p. 59.

[14] F. Zhu, Y. Ren, F. Kong, H. Wu, S. Liang, N. Chen, W. Xu, and F. Zhang, "Swarm-lio: Decentralized swarm lidar-inertial odometry," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3254–3260.

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

[16] 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.

[17] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "Litamin: Lidar-based tracking and mapping by stabilized icp for geometry approximation with normal distributions," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5143–5150.

[18] ——, "Litamin2: Ultra light lidar-based slam using geometric approximation applied with kl-divergence," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 619–11 625.

[19] C. Yuan, W. Xu, X. Liu, X. Hong, and F. Zhang, "Efficient and probabilistic adaptive voxel mapping for accurate online lidar odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8518–8525, 2022.

[20] J. Ruan, B. Li, Y. Wang, and Y. Sun, "Slamesh: Real-time lidar simultaneous localization and meshing," *arXiv preprint arXiv:2303.05252*, 2023.

[21] T.-M. Nguyen, S. Yuan, M. Cao, L. Yang, T. H. Nguyen, and L. Xie, "Miliom: Tightly coupled multi-input lidar-inertia odometry and mapping," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5573–5580, 2021.

[22] T.-M. Nguyen, D. Duberg, P. Jensfelt, S. Yuan, and L. Xie, "Slict: Multi-input multi-scale surfel-based lidar-inertial continuous-time odometry and mapping," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2102–2109, 2023.

[23] Y. Wang, W. Song, Y. Lou, F. Huang, Z. Tu, and S. Zhang, "Simultaneous localization of rail vehicles and mapping of environment with multiple lidars," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8186–8193, 2022.

[24] Y. Wang, Y. Lou, W. Song, and Z. Tu, "A tightly-coupled framework for large-scale map construction with multiple non-repetitive scanning lidars," *IEEE Sensors Journal*, vol. 22, no. 4, pp. 3626–3636, 2022.

[25] S. Das, N. Mahabadi, M. Fallon, and S. Chatterjee, "M-lio: Multi-lidar, multi-imu odometry with sensor dropout tolerance," in *2023 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2023, pp. 1–7.

[26] M. Helmberger, K. Morin, B. Berner, N. Kumar, D. Wang, Y. Yue, G. Cioffi, and D. Scaramuzza, "The hilti slam challenge dataset," 2021.

[27] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.

[28] A. Reinke, M. Palieri, B. Morrell, Y. Chang, K. Ebadi, L. Carlone, and A.-A. Agha-Mohammadi, "Locus 2.0: Robust and computationally efficient lidar odometry for real-time 3d mapping," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9043–9050, 2022.

[29] X. Liu, C. Yuan, and F. Zhang, "Targetless extrinsic calibration of multiple small fov lidars and cameras using adaptive voxelization," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.