Traj-LO: In Defense of LiDAR-Only Odometry Using an Effective Continuous-Time Trajectory

Xin Zheng, *Graduate Student Member, IEEE*, and Jianke Zhu, *Senior Member, IEEE*

***Abstract*—LiDAR Odometry is an essential component in numerous robotic applications. However, current LiDAR-only methods have poor performance in aggressive situations. Unlike the mainstreamed approaches that focus on improving accuracy by the additional inertial sensors, this letter demonstrates that LiDAR is enough to achieve the similar capability through a continuous-time perspective. Firstly, the measurements of LiDAR are regarded as streaming points continuously captured at separate time. Secondly, the LiDAR movement is parameterized by a simple yet effective continuous-time trajectory. Therefore, our proposed method named Traj-LO can recover the spatial-temporal consistent movement of LiDAR by tightly coupling the geometric information from LiDAR points and kinematic constraints from trajectory smoothness. This framework is generalized for different kinds of LiDAR as well as multi-LiDAR systems. Extensive experiments on the public datasets verify the robustness and effectiveness of our proposed LiDAR-only approach, even in scenarios where the kinematic state exceeds the IMU's measuring range.**

***Index Terms*—SLAM, localization, range sensing.**

I. Introduction

LIDAR Odometry (LO) plays a crucial role in robot navigation and path planning within GPS-denied environments [1] due to its exceptional range sensing ability. Over the past decade, increasing research efforts have been devoted to the aspects like feature selection [2], [3], map representation [4], [5], and sensor fusion [6], [7] to enhance the precision, efficiency, and robustness of LO systems.

Compared to LiDAR-Inertial Odometry (LIO) systems that leverage additional information from Inertial Measurement Units (IMUs) [6], [7], the accuracy of currently dominant LiDAR-only methods [5], [8], [9], [10], [11] still exhibits a notable gap in practical applications [12], [13]. This is mainly due to the fact that high-frequency propagated IMU states provide more accurate motion compensation. Moreover, IMUs enhance system robustness by directly enforcing kinematic constraints, through either pre-integration [14] or Kalman Filters [7]. Unfortunately, a stable LIO heavily depends on the precise calibration of IMU parameters, and IMU data are sensitive to temperature variations and mechanical shocks. Furthermore, recent research [15] indicates that even the state-of-the-art LIO systems [6], [7] struggle in scenarios where the kinematic state exceeds the IMU measurement range, as depicted in Fig. 1. These challenges motivate us to revisit the potential of a LiDAR-only approach in extreme situations, which are typically considered insurmountable.

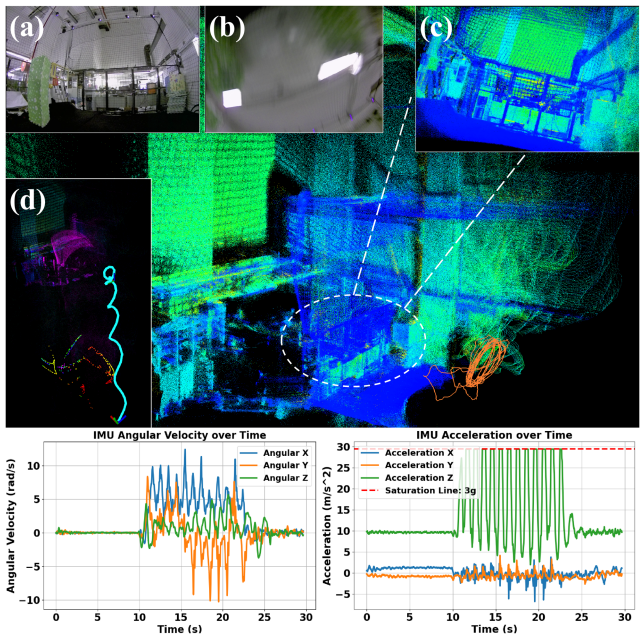

Fig. 1. Upper figure shows the mapping result of an indoor scenario (a) in case acceleration exceeds IMU measuring range 3G. Orange line is LiDAR trajectory. (b) Camera sensor is totally blurred in such fast motion. (d) FAST-LIO [7] drift significantly (c) while our method is still valid only using LiDAR.

Indeed, LiDAR is a streaming sensor with continuous movements in order to capture scene structures. Yet, the standard LO paradigm [8] typically represents the trajectory as a sequence of discrete-time poses, which severs the inherent connection between geometry and kinematics. Ideally, the geometry of continuous scene structures should reflect the spatial-temporal movement of LiDAR, while kinematics has motion-coherent constraints [16] on consecutively observed points. Hence, our

Manuscript received 21 September 2023; accepted 22 December 2023. Date of publication 10 January 2024; date of current version 18 January 2024. This letter was recommended for publication by Associate Editor M. Hanheide and Editor S. Behnke upon evaluation of the reviewers' comments. This work was supported by National Natural Science Foundation of China under Grant 62376244. *(Corresponding author: Jianke Zhu.)*

The authors are with the College of Computer Science, Zhejiang University, Hangzhou 310027, China (e-mail: xinzheng@zju.edu.cn; jkzhu@zju.edu.cn).

Our implementation is available at https://github.com/kevin2431/Traj-LO.

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3352360, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3352360

proposed odometry approach adopts a continuous-time trajectory to tightly couple geometric information and kinematic prior hiding behind streaming points.

The key of continuous-time trajectory is the ability to query the reference LiDAR pose of each measured point by its corresponding timestamp. Therefore, the previously neglected temporal information can be employed to iteratively adjust point position without the need of motion compensation. This continuous-time registration technique [10], [17] ensures not only spatial consistency but also temporal coherence. Moreover, the continuous-time trajectory introduces an indirect kinematic constraint through its smoothness, which is vital for preventing divergence during registration. To overcome the inefficiency of current methods [18], [19], [20], our trajectory is composed of multiple linear segments. This enables us to approximate continuous movement via the simplest linear interpolation within each segment. Furthermore, the trajectory is represented in the compact SE(3) parameterization, which greatly facilitates the derivation of analytic Jacobians. Additionally, the continuous-time trajectory inherently accommodates the fusion of multiple asynchronous LiDARs into a unified optimization framework, which presents a natural advantage in scenarios with multiple LiDAR sensors.

In summary, the main contributions of this letter are: 1) the LiDAR-only odometry approach with a simple yet effective continuous-time trajectory, which is robust in aggressive motions and adaptable to various kinds of LiDAR, including multi-LiDARs; 2) spatiotemporal consistent registration by leveraging previously overlooked temporal information from millions of points and the inherent smoothness of continuous trajectory; 3) extensive experiments on diverse datasets to demonstrate that our LiDAR-only method matches the performance of state-of-the-art LIO approaches [6], [7] and even surpasses them in extreme scenarios.

II. Related Works

A. LO and LIO Using Discrete-Time Trajectory

LOAM [8] is the seminal work on LiDAR odometry, which divided the points into line and planar features based on the roughness of each point on its respective ring. Then, a modified Iterative Closest Point (ICP) algorithm [21] incorporating both point-to-line and point-to-plane metrics was utilized to estimate poses at a scan rate of 10 Hz. Many subsequent works [2], [3], [5], [9], [11] aim to enhance the performance on the KITTI Odometry benchmark [22]. Notably, F-LOAM [9] introduced a lightweight scheme with efficient optimization. Recently, KISS-ICP [11] achieved the remarkable performance by adopting simple point-to-point ICP, complemented by a constant velocity model to mitigate motion distortion [12], [13]. However, existing LiDAR-only approaches struggle with the aggressive motions.

Practically, LIO aided by additional IMU measurements is recognized as a more robust solution. Shan et al. [6] formulated LIO by a factor graph framework to integrate various constraints, including loop closure, wheel odometry, and GPS, into the LIO system. Xu et al. [7] fused LiDAR and IMU measurements within an iterative error-state Kalman Filter framework, which corrected distorted points through back-propagation and directly registered raw points onto an incremental kd-tree map for fast neighborhood searching.

The above methods represent LiDAR movement by a series of discrete-time poses. The points collected within the period of consecutive poses have to be transformed into the position at a specific timestamp. This is called motion compensation, which is usually done before registration.

B. LO and LIO Using Continuous-Time Trajectory

Indeed, the LiDAR points are streaming data captured continuously. Previous methods of computing rigid transformation between two discrete-time poses suffer from motion distortions, where the reference LiDAR center of each measured point is not exactly the same as body movement. In general, the continuous-time trajectory [23] has the ability to query any state of sensors' spatial-temporal movement at any given timestamp. The challenge is how to approximate the entire trajectory with finite states while preserving accuracy and ensuring efficiency.

One major class is based on Gaussian Processes (GP) [18], [24], [25], [26]. Tong et al. [24] employed GP to model the continuous movement in SLAM, where Gauss-Newton method is used for state updating. Barfoot et al. [25] explored the sparsity by incorporating time-varying prior, which makes GP regression more efficient. Anderson and Barfoot [27] conducted batch continuous-time trajectory estimation in SE(3) space, while Dong et al. [26] extended sparse GP regression to the general Lie Group. More recently, Wu et al. [18] incorporated Doppler velocity measurements from FMCW LiDAR into GP-based trajectory estimation. However, it only achieves 2 Hz, which is the common limitation of GP methods.

Another solution is the parametric trajectory, especially B-spline [28]. Quenzel and Behnke introduced a surfel-based B-spline approach, also known as MARS [19], which validated across both regular driving and aerial scenarios. CLINS [20] integrated IMU data into trajectory estimation, although it struggled to meet real-time requirements. Its successor, CLIC [29], extended the B-spline representation to the LiDAR-inertial-camera system.

In contrast to B-splines which are weighted sums of basis functions, linear interpolation offers a more straightforward parametric approach to describing trajectories. However, linear interpolation at a low-frequency scan rate becomes inaccurate when facing aggressive motion. CT-ICP [10] simply addressed this issue by attributing the errors to scan discontinuities and mitigating them through the additional motion constraints. It has proven effective in ordinary driving scenarios but is unreliable for handheld devices and aerial vehicles. Therefore, other approaches such as Zebedee [30], ElasticSLAM [31], and Wildcat [32] adopted a series of high-frequency control poses with smaller temporal intervals to reduce linear errors caused by irregular motions. Nonetheless, these methods heavily rely on IMU measurements for robust estimation and also require a tedious cubic B-spline fitting procedure to refine the trajectory.

III. Methodology

A. Preliminary

To facilitate subsequent derivations, we firstly clarify the notations in the following. We define the continuous-time trajectory as $\mathbf{T}(t)$ over the interval $[t_0, t_K)$. For a specific pose at timestamp t, we denote it as $\mathbf{T}_t$. All poses are represented in world coordinates, where the world frame is the starting location

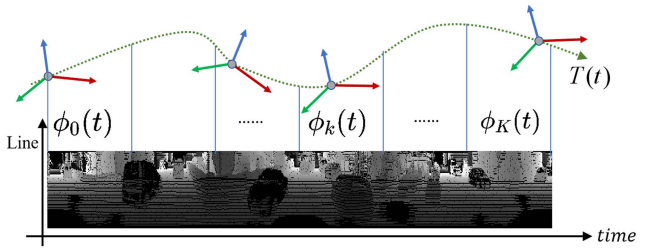

Fig. 2. Trajectory $\mathbf{T}(t)$ shows the continuous movement of LiDAR over interval $[t_0, t_K)$. The bottom is a range image [5] of point cloud collected by a 64-line LiDAR in a 360° circular motion. Only points in the same column are measured simultaneously.

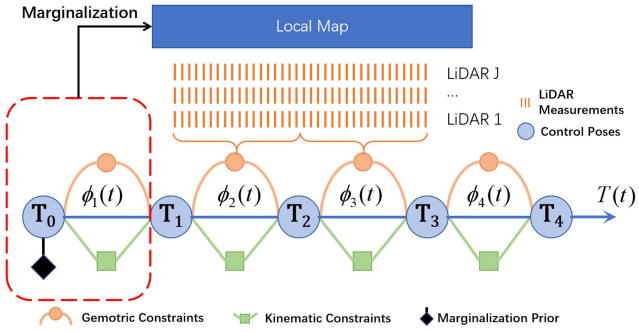

Fig. 3. Trajectory $\mathbf{T}(t)$ consisted with 4 segments, where each segment $\phi_k(t)$ is modeled by linear interpolation in Section III-B. There are three kinds of constraints, including geometry, kinematics, and marginalization.

of LiDAR. The concrete LiDAR pose $\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix}$ lies on $\mathrm{SE}(3)$ manifold [33], where $\mathbf{t} \in \mathbb{R}^3$ represents the translation and $\mathbf{R} \in \mathrm{SO}(3)$ denotes the rotation. The introduction of $\mathrm{SE}(3)$ enables to formulate the 3D rigid transformation into matrix multiplication.

As in [33], we employ the right plus $\oplus$ and minus $\ominus$ to establish a connection between the vector $\boldsymbol{\tau} \in \mathbb{R}^6$ in tangent space $\mathfrak{se}(3)$ and transformation matrix $\mathbf{T} \in \mathrm{SE}(3)$. Incrementing a transformation matrix by a vector is denoted as $\mathbf{T} \oplus \boldsymbol{\tau} = \mathbf{T}\mathrm{Exp}(\boldsymbol{\tau})$, while difference between two transformation matrix is given by $\mathbf{T}_1 \ominus \mathbf{T}_2 = \mathrm{Log}(\mathbf{T}_2^{-1}\mathbf{T}_1)$. $\mathrm{Exp} : \mathbb{R}^6 \to \mathrm{SE}(3)$ and $\mathrm{Log} : \mathrm{SE}(3) \to \mathbb{R}^6$ are the mapping functions, which inherently circumvent singularities.

B. Continuous-Time Trajectory Parameterization

In this letter, we introduce the continuous-time trajectory parameterization which serves as the foundation. As depicted in Fig. 2, LiDAR is a streaming sensor typically with continuous movement. Points in different columns of the range image are indeed collected at distinct LiDAR poses. By employing a continuous-time trajectory, we can perform registration using the precise LiDAR pose without requiring prior motion compensation. The challenge is how to represent continuous-time trajectories accurately and efficiently.

To tackle this issue, we propose a straightforward yet effective scheme. Specifically, the simplest linear interpolation [10] of the trajectory over the temporal window $[t_0, t_K)$ can be modeled by the beginning pose $\mathbf{T}_0$ and end pose $\mathbf{T}_K$. Unfortunately, linear approximation fails to obtain satisfactory results in case of aggressive motions such as handheld devices or aerial vehicles. Indeed, the body's movement becomes more linear as $t_K - t_0 \to 0$. However, a short interval implies fewer accumulated points, which may potentially lead to under-constrained situations during the registration process.

To account for rapid motion, we divide the temporal window into K equidistant small-resolution segments $\{[t_{k-1}, t_k)\}_{k=1}^{K}$. In each segment $[t_{k-1}, t_k)$, the movement is still parameterized with two control poses, the initial $\mathbf{T}_{k-1}$ and final $\mathbf{T}_k$. We represent the continuous-time trajectory over this segment as function $\phi_k(t)$. For a measured point with timestamp t_i in this segment, we retrieve its associated reference LiDAR pose as $\mathbf{T}_{t_i} = \phi_k(t_i)$

$$\phi_k(t_i) = \mathbf{T}_{k-1} \oplus (\alpha_i \boldsymbol{\tau}_k), \tag{1}$$

where $\alpha_i = (t_i - t_{k-1})/\Delta t_k$, and $\boldsymbol{\tau}_k = \mathbf{T}_k \ominus \mathbf{T}_{k-1}$. The interval length of each segment $\Delta t_k = t_k - t_{k-1}$ is a hyperparameter that depends on the motion profile. Then, the entire trajectory $\mathbf{T}(t)$ over temporal window $[t_0, t_K)$ consist of K piecewise functions $\{\phi_1(t), \ldots, \phi_k(t), \ldots, \phi_K(t)\}$. The trajectory $\mathbf{T}(t)$ during the k-th temporal interval $[t_{k-1}, t_k)$ corresponds to the k-th segment function, where $\mathbf{T}(t) = \phi_k(t)$. In case of aggressive motion, we select a shorter interval Δt_k to satisfy linear approximation in each segment. Meanwhile, the abundant measurements collected over the entire window provide the necessary geometric constraints for robust trajectory estimation.

Finally, the continuous-time trajectory $\mathbf{T}(t)$ within the temporal window $[t_0, t_K)$ actually depends on $K+1$ control poses $\{\mathbf{T}_0, \ldots, \mathbf{T}_k, \ldots, \mathbf{T}_K\}$. Unlike the discrete-time trajectory that only indicates the LiDAR position at $K+1$ moments, our proposed approach employs these $K+1$ control poses to model the entire continuous-time trajectory. At the first glance, our piecewise continuous-time trajectory is quite simple to implement. By properly handling three parts in Section III-C, our proposed LiDAR-only odometry using such parameterization performs on par with the state-of-the-art LIO methods [6], [7].

C. LiDAR Odometry Using Continuous-Time Trajectory

We present the LiDAR odometry approach using proposed trajectory parameterization. Fig. 3 illustrates the whole pipeline. To facilitate the real-time requirement, our method works in a sliding window fashion, which consists of three main components, including continuous-time registration, kinematic constraint, and marginalization.

To derive the probabilistic formulation of LiDAR odometry, we introduce the control input $\mathbf{u}(t)$ and map $\mathcal{M} \doteq \{\mathbf{q}_m\}_{m=1}^{M}$. The continuous-time variable $\mathbf{u}(t)$ affects the system's dynamics, while $\mathcal{M}$ is constructed by LiDAR points before the current temporal window. Each map point $\mathbf{q}_m$ is defined in the world coordinates, which is discussed in Section III-E. The estimated trajectory $\mathbf{T}(t)$ over interval $[t_0, t_K)$ depends on a series of LiDAR measurements $\mathcal{Z}$. Our target is to seek the maximum joint posterior distribution $p(\mathbf{T}(t) \mid \mathbf{u}(t), \mathcal{M}, \mathcal{Z})$ over this interval.

Since the control input $\mathbf{u}(t)$ does not influence the LiDAR measurements for the given $\mathbf{T}(t)$, we rewrite the posterior using Bayes' rule as follows:

$$p(\mathbf{T}(t) \mid \mathbf{u}(t), \mathcal{M}, \mathcal{Z}) \propto p(\mathcal{Z} \mid \mathbf{T}(t), \mathcal{M}) p(\mathbf{T}(t) \mid \mathbf{u}(t)).$$

1) Continuous-Time Geometric Constraint: The first part $p(\mathcal{Z} \mid \mathbf{T}(t), \mathcal{M})$ encodes the geometric relationship. Instead of just focusing on a specific LiDAR, our method is designed to accommodate various types of LiDARs, such as multi-line spinning and non-repetitive LiDARs, as well as different configurations of LiDAR systems, whether single or multiple. Given the potential involvement of multiple LiDARs in our system, we assume the continuous trajectory $\mathbf{T}(t)$ rigidly attached to a reference coordinate system that typically corresponds to the first LiDAR.

According to our trajectory parameterization, we organize all measurements $\mathcal{Z}$ into K sets, denoted as $\mathcal{Z} \doteq \{\mathcal{Z}_k\}_{k=1}^{K}$. Each set $\mathcal{Z}_k$ includes measurements collected within the interval $[t_{k-1}, t_k)$ from J LiDARs. The specific form is

$$\mathcal{Z}_k \doteq \left\{ \left\{ {}_k\mathbf{z}_i^j \right\}_{i=1}^{N_k^j} \right\}_{j=1}^{J}, \quad {}_k\mathbf{z}_i^j = \left\{ {}_k\mathbf{p}_i^j, {}_kt_i^j \right\}. \tag{2}$$

where N_k^j is the point number measured by j-th LiDAR in k-th segment. Note that each measurement ${}_k\mathbf{z}_i^j$ consists of a pair, i.e., the point's 3D position ${}_k\mathbf{p}_i^j = ({}_kx_i^j, {}_ky_i^j, {}_kz_i^j)$ and its corresponding timestamp ${}_kt_i^j$. Additionally, the extrinsic calibration parameter for each LiDAR relative to the reference coordinate system is represented by $\mathbf{T}_{L_j}^B$. These parameters are pre-calibrated and remained to be fixed throughout the optimization process.

Assuming that all measurements are independent, $p(\mathcal{Z} \mid \mathbf{T}(t), \mathcal{M})$ can be factorized as follows

$$\prod_{k=1}^{K}\prod_{j=1}^{J}\prod_{i=1}^{N_k^j} p\left(\left\{ {}_k\mathbf{p}_i^j, {}_kt_i^j \right\} \mid \mathbf{T}({}_kt_i^j), \mathcal{M} \right). \tag{3}$$

Existing LOs [9], [11] that utilize the discrete-time trajectories often assume that all points are measured simultaneously at a specific timestamp. Thus, the accuracy of these systems is highly dependent on the quality of motion compensation.

In contrast, our approach implements a spatial-temporal consistent registration. We not only consider the 3D point positions but also pick up the neglected temporal information, which allows for iterative updates of the point positions. Since we can query the reference LiDAR pose for each point using the continuous-time trajectory, it is unnecessary to take consideration of motion distortion. Moreover, we use the point-to-plane criteria for registration without selecting feature. This makes our approach to be effective for different types of LiDAR. The geometric error of ${}_k\mathbf{z}_i^j$ is

$$\mathbf{e}_{{}_k\mathbf{z}_i^j} = {}_k\mathbf{n}_i^{j\top} \left(\phi_k \left({}_kt_i^j \right) \cdot \mathbf{T}_{L_j}^B \cdot {}_k\mathbf{p}_i^j - {}_k\mathbf{q}_i^j \right). \tag{4}$$

where $\phi_k({}_kt_i^j)$ is the pose of reference coordinate system computed by (1). ${}_k\mathbf{q}_i^j$ is the nearest neighbor of ${}_k\mathbf{p}_i^j$ in the map $\mathcal{M}$. The normal vector ${}_k\mathbf{n}_i^j$ is computed by Principal Component Analysis choosing five closet points in $\mathcal{M}$ like [7], [10].

Ideally, the point-to-plane observation should be close to zero. We assume that $\mathbf{e}_{{}_k\mathbf{z}_i^j} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_r)$ accounts for the LiDAR measurement error. Therefore, we have:

$$p\left(\left\{ {}_k\mathbf{p}_i^j, {}_kt_i^j \right\} \mid \mathbf{T}\left({}_kt_i^j \right), \mathcal{M} \right) = \mathcal{N}\left(\mathbf{e}_{{}_k\mathbf{z}_i^j}, \mathbf{Q}_r \right) \tag{5}$$

2) Trajectory Smoothness Constraint: The second part $p(\mathbf{T}(t) \mid \mathbf{u}(t))$ encodes the kinematic model. It is typically utilized in LIOs [6], [7] but is often neglected in LO. This is because the control input $\mathbf{u}(t)$ can directly connect to the IMU data. Indeed, kinematics is an inherent property of the system, even without the assistance of IMU. In this letter, we propose an indirect motion constraint from trajectory smoothness, which is vital for the convergence of continuous-time registration only depending on LiDAR input.

Given that our trajectory parameterization has divided the temporal window into several small intervals, the velocity within each segment is approximately constant. Then, we define the generalized velocity $\varpi_k(t)$ in SE(3) over the interval $[t_{k-1}, t_k)$ as $\varpi_k(t) = (\mathbf{T}_k \ominus \mathbf{T}_{k-1})/\Delta_k$. In the physical world, a continuous trajectory should exhibit smoothness, where the velocity between consecutive segments tends to be coherent. Thus, we introduce a pseudo-velocity measurement $\check{\varpi}_k = (\check{\mathbf{T}}_{k-1} \ominus \check{\mathbf{T}}_{k-2})/\Delta_{k-1}$ from previous segment and ensure $\varpi_k(t) - \check{\varpi}_k$ to be close to zero. $\check{\mathbf{T}}_{k-1}$ and $\check{\mathbf{T}}_{k-2}$ are fixed values obtained after the convergence of the previous sliding window optimization, rather than variables in the current optimization window. Since the intervals of each segment are equidistant, we directly define the smoothness error through control poses as:

$$\mathbf{e}_{v_k} = \mathbf{T}_k \ominus \mathbf{T}_{k-1} - \check{\mathbf{T}}_{k-1} \ominus \check{\mathbf{T}}_{k-2}. \tag{6}$$

Let $\mathbf{e}_{v_k} \sim \mathcal{N}(0, \mathbf{Q}_v)$ follows a zero-mean Gaussian distribution, $p(\mathbf{T}(t) \mid \mathbf{u}(t))$ over the entire trajectory is split into

$$p(\mathbf{T}(t) \mid \mathbf{u}(t)) = \prod_{k=1}^{K} p(\phi_k(t), \mathbf{u}_k(t))$$

where $p(\phi_k(t), \mathbf{u}_k(t)) = \mathcal{N}(\mathbf{e}_{v_k}, \mathbf{Q}_v)$.

3) Joint Optimization: Our target is to find the optimal continuous-time trajectory $\mathbf{T}(t)$ depending on $K+1$ control poses $\{\mathbf{T}_k\}_{k=0}^{K}$ over the temporal window $[t_0, t_K)$. Thus, we minimize the negative logarithm of the posterior likelihood as below

$$\{\mathbf{T}_k{}^*\}_{k=0}^{K} = \underset{\mathbf{T}(t)}{\operatorname{argmin}}(-\log(p(\mathbf{T}(t) \mid \mathbf{u}(t), \mathcal{M}, \mathcal{Z})). \tag{7}$$

Since all distributions are Gaussian, it can be transferred into a non-linear least squares problem that minimizes the following energy function

$$\{\mathbf{T}_k{}^*\}_{k=0}^{K} = \underset{\mathbf{T}(t)}{\operatorname{argmin}}\, E_{reg} + E_{kine} + E_{marg}, \tag{8}$$

$$E_{reg} = \sum_{k=1}^{K}\sum_{j=1}^{J}\sum_{i=1}^{N_k^j} \mathbf{e}_{{}_k\mathbf{z}_i^j}^{\top} \mathbf{Q}_r^{-1} \mathbf{e}_{{}_k\mathbf{z}_i^j}, \; E_{kine} = \sum_{k=1}^{K} \mathbf{e}_{v_k}^{\top} \mathbf{Q}_v^{-1} \mathbf{e}_{v_k}.$$

We have provided the formulation for error terms $\mathbf{e}_{{}_k\mathbf{z}_i^j}$ and $\mathbf{e}_{v_k}$, while E_{marg} represents the marginalization energy resulting from the sliding window optimization.

4) Marginalization: To ensure real-time performance, we maintain exactly K segments in the optimization window. Once a new segment $\phi_{K+1}(t)$ is obtained, we shift out the oldest segment $\phi_1(t)$. Since the interval of each segment is typically shorter than the duration of a single scan, the K segments within the sliding window may be from different scans. Instead of simply discarding measurements collected over $[t_0, t_1)$, we employ the marginalization priors to retain information that falls outside the next temporal window.

As described in [34], the computation of the marginalization priors $\mathbf{H}_m$ and $\mathbf{b}_m$ occurs after the optimization in the old

window has converged. We derive $\mathbf{H}_m$ and $\mathbf{b}_m$ using Schur complement, considering only the energy terms that depend on the marginalized variables. For more detailed procedures, please refer to [34]. The control poses in the new optimization window can be split into two parts $\mathbf{s} = \{\mathbf{s}_m, \mathbf{s}_n\}$, where $\mathbf{s}_m$ is related to marginalized poses, and $\mathbf{s}_n$ is irrelevant. The marginalization energy is

$$E_m = \frac{1}{2}(\mathbf{s}_m \ominus \bar{\mathbf{s}}_m)^\top \mathbf{H}_m (\mathbf{s}_m \ominus \bar{\mathbf{s}}_m) + \mathbf{b}_m^\top (\mathbf{s}_m \ominus \bar{\mathbf{s}}_m),$$

and only influences $\mathbf{s}_m$. $\bar{\mathbf{s}}_m$ is the fixed linearization point when applying Schur complement. In our presented trajectory representation, segment $\phi_1(t)$ depends on two control poses, $\mathbf{T}_0$ and $\mathbf{T}_1$. Since $\mathbf{T}_1$ is relevant to the next segment $\phi_2(t)$, only $\mathbf{T}_0$ will be removed during the marginalization. Besides, the energy term depends on $\mathbf{T}_0$, which is related to $\mathbf{T}_1$. Thus, $\mathbf{s}_m$ in new window includes the poses $\mathbf{T}_1$ from the old window. In addition, we apply the first-estimate Jacobians [35] to keep the consistency of the reduced system. For the poses connected to marginalization prior, we derived Jacobin at their fixed linearization point $\bar{\mathbf{s}}_m$, while the residuals are calculated at their current state $\mathbf{s}_m$.

D. Analytic Jacobian Derivation

We make use of Gauss-Newton optimization to solve the nonlinear least square minimization in(8). By the first-order Taylor expansion, the general error term around the linearized point $\mathbf{s}$ is approximated as $\mathbf{e}(\mathbf{s} \oplus \boldsymbol{\xi}) \simeq \mathbf{e}(\mathbf{s}) + \frac{\partial \mathbf{e}}{\partial \mathbf{s}}\boldsymbol{\xi}$, where $\mathbf{s}$ is $K+1$ poses with its stacked increment vector $\boldsymbol{\xi} \in \mathbb{R}^{6(K+1)}$. $\partial \mathbf{e}/\partial \mathbf{s}$ is the Jacobian. The optimal increment is found by solving the normal equation $\mathbf{H}\boldsymbol{\xi} = -\mathbf{b}$, where the Hessian matrix $\mathbf{H} = \sum(\frac{\partial \mathbf{e}}{\partial \mathbf{s}})^\top \mathbf{W}^{-1} \frac{\partial \mathbf{e}}{\partial \mathbf{s}}$ and $\mathbf{b} = \sum(\frac{\partial \mathbf{e}}{\partial \mathbf{s}})^\top \mathbf{W}^{-1}\mathbf{e}$. $\mathbf{W}$ is the related covariance. Each control pose of our continuous trajectory iteratively updates as $\mathbf{T}_k \leftarrow \mathbf{T}_k \oplus \boldsymbol{\xi}_k$, where $\mathbf{s} = \begin{bmatrix}\mathbf{T}_0 & \dots & \mathbf{T}_K\end{bmatrix}$ and $\boldsymbol{\xi} = \begin{bmatrix}\boldsymbol{\xi}_0 & \dots & \boldsymbol{\xi}_K\end{bmatrix}$.

To speed up optimization, we derive analytic Jacobian rather than inefficient automatic differentiation [10]. Indeed, each error term is only related to several poses. Our derivation uses the right forms in [33] so that the concrete Jacobians of continuous-time registration term $\mathbf{e}_{{}_k\mathbf{z}_i^j}$ is

$$\frac{\partial \mathbf{e}_{{}_k\mathbf{z}_i^j}}{\partial \mathbf{s}} = \frac{\partial \mathbf{e}_{{}_k\mathbf{z}_i^j}}{\partial \phi_k({}_kt_i^j)} \begin{bmatrix} \frac{\partial \phi_k({}_kt_i^j)}{\partial \mathbf{T}_{k-1}} & \frac{\partial \phi_k({}_kt_i^j)}{\partial \mathbf{T}_k} \end{bmatrix}, \tag{9a}$$

$$\frac{\partial \mathbf{e}_{{}_k\mathbf{z}_i^j}}{\partial \phi_k({}_kt_i^j)} = {}_k\mathbf{n}_i^{j\top} \begin{bmatrix} \mathbf{R}_{{}_kt_i^j} & -\mathbf{R}_{{}_kt_i^j} \left[\mathbf{T}_{L_j}^B \cdot {}_k\mathbf{p}_i^j\right]_\times \end{bmatrix}, \tag{9b}$$

$$\frac{\partial \phi_k({}_kt_i^j)}{\partial \mathbf{T}_{k-1}} = (1 - {}_k\alpha_i^j)\mathbf{J}_r((_k\alpha_i^j - 1)\boldsymbol{\tau}_k)\mathbf{J}_l^{-1}(\boldsymbol{\tau}_k), \tag{9c}$$

$$\frac{\partial \phi_k({}_kt_i^j)}{\partial \mathbf{T}_k} = {}_k\alpha_i^j \mathbf{J}_r({}_k\alpha_i^j \boldsymbol{\tau}_k)\mathbf{J}_r^{-1}(\boldsymbol{\tau}_k). \tag{9d}$$

$\mathbf{R}_{{}_kt_i^j}$ is the rotation part of reference coordinate pose $\mathbf{T}_{{}_kt_i^j} = \phi_k({}_kt_i^j)$ at ${}_kt_i^j$. $[\cdot]_\times$ denotes the skew symmetric matrix. $\mathbf{J}_r(\cdot)$ and $\mathbf{J}_l(\cdot)$ are the right- and left- Jacobians [33] of SE(3), respectively. The Jacobians of kinematic constraints are

$$\frac{\partial \mathbf{e}_{v_k}}{\partial \mathbf{T}_{k-1}} = -\mathbf{J}_l^{-1}(\boldsymbol{\tau}_k), \quad \frac{\partial \mathbf{e}_{v_k}}{\partial \mathbf{T}_k} = \mathbf{J}_r^{-1}(\boldsymbol{\tau}_k). \tag{10}$$

E. Map Management

The bottleneck of LiDAR odometry is thousands of nearest-neighbor searches during point registration. In order to accelerate this intensive and repetitive operation, our map adopts a spatial hashing structure from [10], [11]. The voxel size depends on the environment. Each voxel stores up to 20 points and we explore the closest 7 voxels during nearest-neighbor searching. Generally, we assume the LiDAR odometry starts from a stationary state, and use the points collected within the first 0.3 s for map initialization. The voxel index of each point is computed by its position in world coordinates. The map points are unchangeable during registration. Once the optimization is converged, the map will be updated by the points leaving the window. If the maximum capacity of voxel is reached, we drop all newly collected points. To reduce the memory consumption of the map, points located more than 100 m away from the current LiDAR center are removed.

IV. Experiment

In this section, we present the details of our experiments on AMD Ryzen 9 5900X CPU. To show the effectiveness and generalizability, we evaluate our approach named 'Traj-LO' on three ordinary datasets and another with extreme motion where IMUs are saturated. The comparative methods include state-of-the-art LOs and LIOs.

The selection of hyperparameters, specifically the segment number and interval length, is critical for accurate trajectory estimation. Our criteria for choosing these parameters aim to hold linear assumptions within each segment while ensuring sufficient geometric constraints. To balance the need for a larger Field of View (FoV) and computational efficiency, we choose 4 segments in the temporal window, with each interval Δt_k at 0.03 s. The noise $\mathbf{Q}_r = \sigma_r \mathbf{I}_1$ and $\mathbf{Q}_v = \sigma_v \mathbf{I}_6$ are diagonal matrices, where $\sigma_r = 0.1$ and $\sigma_v = 0.05$. The indoor voxel size is 0.4 m and the outdoor is 0.8 m.

A. Dataset

KITTI odometry dataset [22] provides driving scenarios, in which 3D point scans are collected using the Velodyne HDL-64E S2. This dataset offers a wide range of scenarios from urban city to highway traffic. However, points in all 22 sequences are motion-corrected where temporal information is discarded. Due to its popularity in the robotic community, we still report the evaluation results on this benchmark.

NTU VIRAL [13] is a visual-inertial-ranging-LiDAR dataset for autonomous aerial vehicles. It has two 16-channel Ouster LiDARs, which separately equip on the horizontal and vertical direction. These LiDARs produce point clouds at 10 Hz rate accompanied by a 9-axis IMU at 385 Hz. The point cloud contains the timestamp of each point relative to the scan's start time, which is crucial for continuous-time estimation. The ground truth is obtained by a Leica Nova MS60 in several challenging indoor and outdoor conditions.

Hilti 2021 dataset [12] contains indoor sequences of offices, labs, and construction environments and outdoor sequences of construction sites and parking areas. The LiDAR configuration is an Ouster OS0-64 which collected points in 360° FoV at 10 Hz, and another LiDAR unit is Livox MID70 with a non-repetitive scan pattern in 70° circular FoV at 10 Hz. The devices are mounted on a handheld platform, providing millimeter-accurate

TABLE I
RTE RESULTS (%) ON KITTI ODOMETRY BENCHMARK

Approach	00	01	02	03	04	05	06	07	08	09	10	Online
FLOAM[9]	0.71	**0.71**	0.73	0.98	0.57	0.62	0.33	0.47	1.04	0.88	1.02	0.72
KISS-ICP[11]	0.52	0.72	0.53	**0.65**	**0.35**	0.30	**0.26**	0.33	**0.81**	0.49	0.54	0.61
CT-ICP[10]	**0.49**	0.76	**0.52**	0.72	0.39	**0.25**	0.27	0.31	**0.81**	0.49	**0.48**	0.59
Ours	0.50	0.81	**0.52**	0.67	0.40	**0.25**	0.27	**0.30**	**0.81**	**0.45**	0.55	**0.58**

The bold values represent the best results obtained for each sequence in the KITTI Dataset.

ground truth from the Hilti PLT 300 automated Total Station or a MoCap system. The temporal information of each point is reserved.

Point-LIO dataset [15] is collected by Livox Avia. We select two sequences for evaluation. The outdoor sequence features a spinning motion on a rotating platform, while the indoor involves a circular swinging motion with the LiDAR attached to one end of a rope. Both sequences suffer that the kinematic state exceeds the IMU measuring range, causing most LIOs [6], [7] to fail. The interval Δt_k is 0.01 s and voxel size is 0.2 m for indoor sequence.

B. Comparison on Public Datasets

We compare Traj-LO with the LiDAR-only methods, including FLOAM [9], KISS-ICP [11] and CT-ICP [10].

1) Kitti: At first, a vertical angle of 0.205° is used to rectify the calibration errors in raw point clouds. Since the points are motion-corrected, continuous registration is disabled. We maintain motion constraints between 4 consecutive scans. For quantitative analysis, we use the KITTI relative translation error, and the results are reported in Table I. It can be seen that FLOAM obtains the worst performance among the four methods. Across the 11 training sequences, KISS-ICP, CT-ICP and our Traj-LO achieve similar average accuracy. On the other 11 testing sequences, we have obtained the best online result on the KITTI benchmark with a score of 0.58% translation error and 0.0014 deg/m rotation error.

2) NTU With LO: Moreover, we provide a comprehensive evaluation on more challenging aerial dataset. It records the specific timestamps of each LiDAR point, which makes it possible to examine the advantage of continuous-time approaches. The results of absolute trajectory error (ATE) compared with ground truth aAblationre listed in Table II, which is computed by evo package.[1]

Our proposed Traj-LO approach achieves the best performance and outperforms other methods at a large margin except for tnp sequences. FLOAM nearly drifts on all sequences, since its original public code does not compensate for motion distortion. Although KISS-ICP claims its constant velocity model is on par or even slightly better with the IMU, it drifts on most sequences. For the more complicated NTU dataset, the constant velocity assumption may not be valid. CT-ICP achieves an average accuracy at the centimeter level, similar to our Traj-LO when excluding failed sequences, which outperforms the two previous discrete-time methods.

3) NTU and Hilti With LIO: Traj-LO has demonstrated a significant advantage over the existing LiDAR-only methods. Therefore, we compare it against the widely used LIO methods like LIO-SAM [6] and FAST-LIO [7]. Surprisingly, Traj-LO performs the best on the handheld Hilti dataset in Table III and achieves competitive results on the aerial NTU dataset in Table II. Note that our approach achieves such promising results using only LiDAR points.

[1][Online]. Available: https://github.com/MichaelGrupp/evo

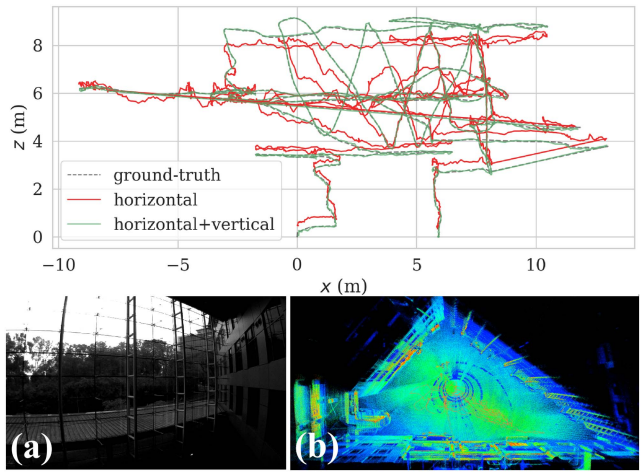

Fig. 4. Tnp sequences occur within a Manhattan-like environment (a), where the majority of the 16-channel laser beams from the horizontal LiDAR sensor intersect with three vertical walls. The top figure illustrates the trajectory of the tnp_01 sequence in the xz plane. (b) Presents the mapping results achieved after integrating data from both the horizontal and vertical LiDAR sensors.

C. Evaluation on Multi-LiDAR System

Our continuous-time trajectory representation is naturally capable of integrating multiple LiDARs into a unified framework. As demonstrated in Table II, Traj-LO outperforms the existing B-spline-based LO method MARS [19], when utilizing both horizontal and vertical LiDARs. Furthermore, our approach exhibits exceptional performance in high-speed spms sequences that is traditionally considered as a strength of LIO approaches such as CLIC [29] and SLICT [36].

A notable advantage of multi-LiDAR systems is their ability to provide supplementary geometric information in scenes, where a single LiDAR may degrade. As illustrated in Fig. 4, tnp sequences lack sufficient features to constrain motion in the vertical direction. Traj-LO can leverage the additional vertical LiDAR to enhance system observability, which significantly reduces large errors along the z-direction.

D. Evaluation on Non-Repetitive LiDAR

Non-repetitive scanning LiDARs with a small FoV are popular as complements to multi-line spinning LiDARs. In the experiment, we evaluate different methods on the Hilti dataset with a Livox MID70. To cover the 70° circular FoV with limited laser heads, the laser direction has to change frequently using Risley prism. This sensor introduces more severe motion distortions [17]. As shown in Table III, continuous-time methods outperform discrete-time methods even without IMU. However, the overall performance of Livox MID70 is lower than that of multi-line Ouster LiDAR, which offers a wider FoV.

TABLE II
ATE (M) ON THE NTU VIRAL DATASET

	Approach	Sensor[1]	eee			nya			sbs			rtp			tnp			spms		
			01	02	03	01	02	03	01	02	03	01	02	03	01	02	03	01	02	03
LO	FLOAM [9]	L1	4.486	8.238	1.133	1.447	1.292	1.498	0.976	2.010	1.079	10.775	4.637	2.218	2.354	2.249	1.566	×	×	×
LO	KISS-ICP [11]	L1	2.220	1.570	1.014	0.628	1.500	1.272	0.917	1.312	1.030	3.663	1.970	2.382	2.305	2.405	0.799	8.493	×	5.451
LO	CT-ICP [10]	L1	7.763	0.125	11.171	0.100	0.101	0.073	×	0.084	1.545	×	0.081	0.086	0.073	0.071	0.045	×	×	×
LO	MARS[3][19]	L1+L2	0.247	0.103	0.093	0.056	0.062	0.083	0.137	0.126	0.159	×	0.233	0.138	0.073	0.068	0.067	×	×	19.865
LO	**Ours**	L1	0.055	0.039	0.035	0.047	0.052	0.050	0.048	0.039	0.039	0.050	0.058	0.057	0.505	0.607	0.101	0.121	×	0.103
LO	**Ours**	L1+L2	0.051	0.033	0.036	0.052	0.045	0.045	0.042	0.042	0.041	0.045	0.061	0.044	0.049	0.040	0.049	**0.056**	0.163	**0.063**
LIO	LIO-SAM [6]	L1+I	0.032	0.050	0.077	0.041	0.056	0.067	0.054	0.043	0.044	0.085	0.073	0.066	0.065	0.127	0.052	0.207	×	0.074
LIO	FAST-LIO [7]	L1+I	**0.029**	**0.019**	**0.023**	0.031	0.031	0.036	0.031	0.045	**0.029**	**0.042**	0.060	**0.052**	0.043	0.037	0.045	**0.056**	**0.050**	0.075
LIO	SLICT[3][36]	L1+L2+I	0.032	0.025	0.028	**0.023**	**0.023**	**0.016**	**0.030**	**0.029**	0.034	0.045	**0.047**	0.050	**0.029**	**0.020**	**0.038**	0.061	0.10	0.066
LIO	CLIC[3][29]	L1+L2+I	0.040	0.021	0.031	0.030	0.037	0.034	0.033	0.037	0.044	0.072	0.239	0.064	0.060	0.061	0.053	0.123	×	0.211

[1] L1: horizon OS1-16, L2: vertical OS1-16, I: external IMU.
[2] The best results overall are in **blod**, while the best results in each category are underlined. '×' denotes divergence.
[3] The results of MARS and SLICT are obtained from the paper [36] while CLIC are from [29].

TABLE III
ATE (M) ON THE HILTI SLAM CHALLENGE DATASET

	Approach	Sensor[1]	RPG	Base1	Base4	Lab	Cons2	Camp2
LO	FLOAM [9]	L1	2.775	0.914	0.287	0.182	11.515	8.946
LO	FLOAM [9]	L2	-	-	-	-	-	-
LO	KISS-ICP [11]	L1	0.187	**0.294**	0.119	0.073	0.835	5.052
LO	KISS-ICP [11]	L2	3.726	6.850	0.293	×	21.650	3.609
LO	CT-ICP [10]	L1	0.188	0.321	0.210	0.052	0.087	0.077
LO	CT-ICP [10]	L2	0.197	×	0.126	×	13.359	6.516
LO	**Ours**	L1	0.172	0.301	0.064	**0.027**	0.065	**0.045**
LO	**Ours**	L2	0.218	0.314	0.064	×	0.135	0.049
LO	**Ours**	L1+L2	**0.170**	0.297	0.049	0.050	**0.063**	0.060
LIO	LIO-SAM [6]	-	-	-	-	-	-	-
LIO	FAST-LIO [7]	L1+I	0.182	0.309	**0.033**	0.035	0.066	0.087
LIO	FAST-LIO [7]	L2+I	0.282	0.313	0.693	×	0.198	0.063
LIO	CLIC [29]	L1+I	0.394	0.340	0.202	0.240	0.327	0.397
LIO	CLIC [29]	L2+I	-	-	-	-	-	-

[1] L1: OS0-64, L2: Livox MID70, I: IMU embedded in L1.
[2] The best results overall are in **blod**, while the best results in each category are underlined. '×' denotes divergence, and '-' denotes invalid results.

TABLE IV
ATE (M) ACROSS VARIOUS SEQUENCE SETTINGS

	nya01-seg1[1]	nya01-seg4	sbs01-seg1	sbs01-seg4
w	0.052	0.047	0.049	0.048
w/o	0.493	0.093	0.081	0.055

[1] seg1: segment number $K = 1$, seg4: segment number $K = 4$.

TABLE V
PERFORMANCE EVALUATION ON NYA01 (395 S)

	FLOAM	KISS-ICP	CT-ICP	Our-seg1	Our-seg4
Time (s)	25.6	47.7	152.8	49.7/110.6[1]	116.2/173.2
Memory (M)	100	53	223	208	208

[1] -/*: '-' is the time consumption with marginalization while '*' is without.

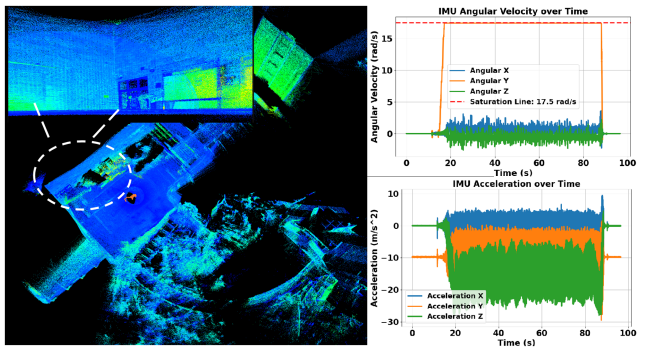

Fig. 5. It is the mapping result of our method on the rotating platform where the angular velocity exceeds the measurement range of 17.5 rd/s. FAST-LIO [7] fails in this situation.

E. Extreme Scenarios Beyond IMU Measuring Range

Besides, we evaluate Traj-LO in more challenging scenarios using the Point-LIO dataset. Figs. 1 and 5 show the mapping results along with the recorded IMU data over time. Remarkably, our LiDAR-only odometry method exhibits minimal drift even in case of exceptionally aggressive motion. This unexpected performance can be attributed to the spatial-temporal information contained within the millions of streaming LiDAR points, which surpasses the capabilities of conventional IMUs with the frequencies between 100–400 Hz. Furthermore, MEMS IMUs tend to introduce noises, especially in aggressive motions, whereas the range measurements obtained from individual LiDAR points are considerably more accurate. By making use of previously overlooked temporal data in registration, LiDAR-only odometry demonstrates a level of capability that exceeds the initial expectations.

F. Evaluation on Marginalization

Gauge freedom constraint [37] is essential to prevent divergence in multi-state optimization. Instead of incorporating a position constraint like CT-ICP [17], we employ a marginalization term to introduce a gauge prior. To demonstrate the efficacy of this approach, we conducted comparative experiments on both an indoor sequence (nya01) and an outdoor sequence (sbs01), across various settings on the number of segments. The results, as depicted in Table IV, clearly show that the marginalization prior effectively reduces trajectory error under different conditions. Notably, the impact of marginalization is more sound with a smaller number of segments. This is likely due to the reduced geometric constraints inherent in setups with fewer segments.

G. Runtime Analysis

Traj-LO is optimized to operate efficiently in real-time. To demonstrate this, we compared the processing time and memory consumption of our method with other LiDAR-only approaches using the nya01 dataset. As described in Table V, our total processing time remains below the duration of the nya01 sequence.

When the number of segments is set to one, Traj-LO aligns with the CT-ICP. Yet, our method exhibits significantly lower computational time, primarily due to the efficient analytic Jacobians. Moreover, the incorporation of marginalization not only improves accuracy but also contributes to faster convergence. As for memory consumption, the maximum memory requirement is capped at 208 MB, since our map is constrained to store points within a fixed radius.

V. Conclusion

This letter has introduced a LiDAR-only odometry using a simple yet effective continuous-time trajectory. By coupling geometric information and kinematic prior hiding behind streaming points, the performance of our approach was on par with start-of-the-art LIOs and even suppressed them in extreme scenarios. Moreover, the proposed odometry was designed to accommodate various types of LiDARs as well as multi-LiDAR systems. At present, our method is primarily designed for odometry task. In future work, we intend to explore the capabilities of a complete LiDAR-only SLAM system that incorporates this continuous-time trajectory along with a more representative map structure.

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] J. E. Deschaud, "IMLS-SLAM: Scan-to-model matching based on 3D data," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 2480–2485.

[3] 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. Automat.*, 2021, pp. 11633–11640.

[4] J. Behley and C. Stachniss, "Efficient surfel-based SLAM using 3D laser range data in urban environments," in *Proc. Robot.: Sci. Syst.*, 2018, pp. 1–10.

[5] X. Zheng and J. Zhu, "Efficient LiDAR odometry for autonomous driving," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 8458–8465, Oct. 2021.

[6] 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.*, 2020, pp. 5135–5142.

[7] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2053–2073, Aug. 2022.

[8] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst.*, 2014, pp. 1–9.

[9] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM: Fast LiDAR odometry and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 4390–4396.

[10] P. Dellenbach, J.-E. Deschaud, B. Jacquet, and F. Goulette, "CT-ICP: Real-time elastic lidar odometry with loop closure," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 5580–5586.

[11] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, "KISS-ICP: In Defense of Point-to-Point ICP–Simple, Accurate, and Robust Registration If Done the Right Way," *IEEE Robot. Automat. Lett.*, vol. 8, no. 2, pp. 1–8, Feb. 2023.

[12] M. Helmberger, K. Morin, B. Berner, N. Kumar, G. Cioffi, and D. Scaramuzza, "The hilti SLAM challenge dataset," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7518–7525, Jul. 2022.

[13] T.-M. Nguyen, S. Yuan, M. Cao, Y. Lyu, T. H. Nguyen, and L. Xie, "NTU viral: A visual-inertial-ranging-lidar dataset, from an aerial vehicle viewpoint," *Int. J. Robot. Res.*, vol. 41, no. 3, pp. 270–280, 2022.

[14] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration theory for fast and accurate visual-inertial navigation," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, Feb. 2017.

[15] D. He, W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang, "Point-LIO: Robust high-bandwidth light detection and ranging inertial odometry," *Adv. Intell. Syst.*, vol. 5, 2023, Art. no. 2200459.

[16] 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.

[17] X. Zheng and J. Zhu, "ECTLO: Effective continuous-time odometry using range image for LiDAR with small FoV," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 9102–9109.

[18] Y. Wu et al., "Picking up speed: Continuous-time lidar-only odometry using doppler velocity measurements," *IEEE Robot. Automat. Lett.*, vol. 8, no. 1, pp. 264–271, Jan. 2023.

[19] J. Quenzel and S. Behnke, "Real-time multi-adaptive-resolution-surfel 6D LiDAR odometry using continuous-time trajectory optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 5499–5506.

[20] J. Lv, K. Hu, J. Xu, Y. Liu, X. Ma, and X. Zuo, "CLINS: Continuous-time trajectory estimation for LiDAR-inertial system," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 6657–6663.

[21] P. J. Besl and N. D. McKay, "Method for registration of 3-D shapes," *Proc. SPIE*, vol. 1611, pp. 586–606, 1992.

[22] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.

[23] P. Furgale, T. D. Barfoot, and G. Sibley, "Continuous-time batch estimation using temporal basis functions," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 2088–2095.

[24] C. H. Tong, P. Furgale, and T. D. Barfoot, "Gaussian process gauss–newton for non-parametric simultaneous localization and mapping," *Int. J. Robot. Res.*, vol. 32, no. 5, pp. 507–525, 2013.

[25] T. D. Barfoot, C. H. Tong, and S. Särkkä, "Batch continuous-time trajectory estimation as exactly sparse Gaussian process regression," in *Proc. Robot.: Sci. Syst.*, 2014, pp. 1–10.

[26] J. Dong, M. Mukadam, B. Boots, and F. Dellaert, "Sparse Gaussian processes on matrix lie groups: A unified framework for optimizing continuous-time trajectories," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 6497–6504.

[27] S. Anderson and T. D. Barfoot, "Full STEAM ahead: Exactly sparse Gaussian process regression for batch continuous-time trajectory estimation on SE(3)," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 157–164.

[28] C. Sommer, V. Usenko, D. Schubert, N. Demmel, and D. Cremers, "Efficient derivative computation for cumulative b-splines on lie groups," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11148–11156.

[29] J. Lv, X. Lang, J. Xu, M. Wang, Y. Liu, and X. Zuo, "Continuous-time fixed-lag smoothing for LiDAR-inertial-camera slam," *IEEE/ASME Trans. Mechatronics*, vol. 28, no. 4, pp. 2259–2270, Aug. 2023.

[30] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3-D range sensor with application to mobile mapping," *IEEE Trans. Robot.*, vol. 28, no. 5, pp. 1104–1119, Oct. 2012.

[31] C. Park, P. Moghadam, J. L. Williams, S. Kim, S. Sridharan, and C. Fookes, "Elasticity meets continuous-time: Map-centric dense 3D LiDAR SLAM," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 978–997, Apr. 2022.

[32] M. Ramezani et al., "Wildcat: Online continuous-time 3D LiDAR-inertial SLAM," *CoRR*, vol. abs/2205.12595, 2022.

[33] J. Sola, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," *CoRR*, vol. abs/1812.01537, 2018.

[34] N. Demmel, D. Schubert, C. Sommer, D. Cremers, and V. Usenko, "Square root marginalization for sliding-window bundle adjustment," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13260–13268.

[35] G. P. Huang, A. I. Mourikis, and S. I. Roumeliotis, "A first-estimates jacobian EKF for improving slam consistency," in *Proc. Exp. Robot.: 11th Int. Symp.*, 2009, pp. 373–382.

[36] 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 Robot. Automat. Lett.*, vol. 8, no. 4, pp. 2102–2109, Apr. 2023.

[37] Z. Zhang, G. Gallego, and D. Scaramuzza, "On the comparison of gauge freedom handling in optimization-based visual-inertial state estimation," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 2710–2717, Jul. 2018.