Efficient and Probabilistic Adaptive Voxel Mapping for Accurate Online LiDAR Odometry

Chongjian Yuan , *Graduate Student Member, IEEE*, Wei Xu , *Graduate Student Member, IEEE*, Xiyuan Liu , *Graduate Student Member, IEEE*, Xiaoping Hong , *Member, IEEE*, and Fu Zhang , *Member, IEEE*

Abstract—This letter proposes an efficient and probabilistic adaptive voxel mapping method for LiDAR odometry. The map is a collection of voxels; each contains one plane feature that enables the probabilistic representation of the environment and accurate registration of a new LiDAR scan. We further analyze the need for coarse-to-fine voxel mapping and then use a novel voxel map organized by a Hash table and octrees to build and update the map efficiently. We apply the proposed voxel map to an iterated extended Kalman filter and construct a maximum a posteriori probability problem for pose estimation. Experiments on the open KITTI dataset show the high accuracy and efficiency of our method compared to other state-of-the-art methods. Experiments on indoor and unstructured outdoor environments with solid-state LiDAR and non-repetitive scanning LiDAR further verify the adaptability of our mapping method to different environments and LiDAR scanning patterns (see our attached video[1]). Our codes and dataset are open-sourced on Github[2]

Index Terms—Mapping, Localization, SLAM.

I. INTRODUCTION

IN RECENT years, with the development of 3D LiDAR technology, especially the emergence of low-cost high-density solid-state LiDARs [1], LiDAR(-inertial) odometry has been increasingly used in various applications such as autonomous vehicles [2], UAVs (Unmanned Aerial Vehicles) [3], [4] and 3D mobile mapping devices [5].[1] [2]

[1]https://youtu.be/HSwQdXg31WM

[2]https://github.com/hku-mars/VoxelMap

A fundamental requirement of LiDAR odometry is a proper representation of the environment (i.e., map) that can be maintained efficiently and can register new points effectively. A predominant mapping method in existing LiDAR(-inertial) odometry, such as LOAM [6], Lego-LOAM [7], LINS [8], LIO-SAM [9], and Lili-OM [10], is a point cloud map, which consists of raw or selected edge or plane points. When registering a new scan, each point in the scan is registered to a small plane fitted by a few nearest points in the map by iterated closet point (ICP) [11] or its variants (e.g., generalized ICP [12]). This line of work consummates in FAST-LIO2 [13], which develops an incremental kd-tree structure to organize the point cloud map efficiently.

As a direct form of LiDAR measurements, a point cloud map is simple to implement. However, a major drawback of the point cloud map is difficult to consider map uncertainty caused by LiDAR measurement noise. In ICP or its variants, the plane fitted by the nearest map points is usually regarded as the true deterministic plane that a new point lies on, while it should really be probabilistic due to measurements noises in the map points. While incorporating these noises in the fitted plane (and the subsequent point registration) is possible, it is less reliable (since the nearest points are usually few in order to avoid spurious planes) and time-consuming (since the uncertainty has to be recomputed if any nearest point is changed during the registration).

Consideration of the map uncertainty requires explicit parameterization of salient features (e.g., plane) in the environment, tracking these features across different LiDAR scans and estimating these feature parameters and their uncertainties. These tasks are challenging due to the various possible environments which contain features of different sizes. The problem is further complicated by the coarse-to-fine phenomenon widely in LiDAR measurements. For example, points in a scan or map usually have drastically different densities in the space due to varying LiDAR resolution, scanning types, and considerably uneven distribution of points in the space. The point density also varies over time due to the sequentially scanning nature of LiDAR sensors.

To address the above challenges, this letter proposes a novel online adaptive voxel mapping method, which constructs voxels of different sizes to adapt to the variation in environment structures and point density. A voxel tracks one salient feature (we use plane in this letter). The feature parameters and uncertainties are repeatedly estimated within the voxel based on received points. The parameterized feature allows us to consider the uncertainty of both the current point and the map accurately. Concretely, the contributions of this letter include:

1) An adaptive-size and coarse-to-fine voxel construction method, which is adaptable to environments of varying structures and robust to the sparsity and irregularity of LiDAR point cloud. The adaptive voxel map is organized in an octree-hash data structure to increase the efficiency of voxel construction, updates, and inquiries.
2) A true probabilistic map representation, where each feature (i.e., plane) contained in the voxel map accurately considers uncertainties arising from both point measurement noise and pose estimation error.
3) An open-source implementation of the proposed mapping method in a LiDAR(-inertial) odometry system and full validation of the design on real-world datasets of various environments (structured and unstructured) and LiDARs (multi-spinning LiDARs and non-conventional solid-state LiDARs). In particular, the superiority of our method over other state-of-art methods on KITTI dataset is demonstrated.

II. RELATED WORKS

On the one hand, the plane feature used in our voxel map is most similar to the surfel used in Suma [14] and Suma++[15], which also register new points by optimizing the point to plane residuals and explicitly maintain and estimate the surfel in the map. The surfel in [14], [15] is a very small planar feature, which is also the case of our plane features in cluttered environments. However, due to the adaptive sizes in our voxel maps, our plane features could be very large in structured environments. This adaptive strategy makes our method suitable for environments with varying structures [16] while ensuring high computation efficiency. Another difference between our plane features with the surfels in [14] and [15] lies in the surfel/plane association. In [14], [15], both surfels in the map and new LiDAR points are projected (i.e., rendered) to a virtual image plane to perform point-to-plane association. Such a surfel rendering process is very time-consuming and typically requires GPU acceleration for online applications. In contrast, our plane association is made by efficient Hash table inquiry.

The computation issue in the surfel rendering could be mitigated by modeling the surfel map as a Gaussian mixture model and performing surfel association by Expectation Maximization, as shown in MRSLaserMap [17] and [18]. The works in [17], [18] also adopted a multi-resolution surfel map, where the valid surfels are merged from the finest resolution to coarser resolution. This fine-to-coarse map method originates from MRSMap [19], which was inspired from the radial nature of the LiDAR scanner, and is usually sensitive to the sparsity and irregularity of the LiDAR point cloud. In contrast, our method constructs voxel maps in a coarse-to-fine manner and is more robust to LiDAR point cloud's irregularity and adaptable to different environments.

Instead of deterministic surfels, the works in [20], [21] incorporate LiDAR measurements noises into surfels, leading to a probabilistic point-to-surfel registration similar to our probabilistic plane representation. However, [20], [21] only considered the LiDAR measurement noises but ignored the uncertainties caused by errors in LiDAR pose estimation. In contrast, our plane uncertainty considers both uncertainties. Notice that the pose estimate is used to transform LiDAR points measured in the local body frame to the world frame where the surfel/plane is estimated, hence the pose estimation error will contribute to the surfel/plane uncertainty.

On the other hand, our voxel map is similar to the normal distribution transformation (NDT) [22], where the map consists of many voxels (of different sizes, e.g., [23]). In each voxel, a 3D Gaussian distribution is fitted from map points lying in the voxel and is used to register a new point by maximizing the likelihood of the point being taken from the distribution. LiTAMIN2 [24] further improves the accuracy of NDT by incorporating ICP with NDT. When compared with these NDT methods, our method explicitly parameterizes and estimates a plane in a voxel. Such explicit plane parameterization allows performing direct point-to-plane registration, which is more sound and often accurate than NDT where the 3D Gaussian distribution over constrains the new point by penalizing all three directions (instead of the plane normal only). The explicit plane parameterization also allows incorporating noises in both LiDAR point measurements and pose estimation, which provides further accuracy improvements.

In terms of data structure, our proposed octo-hash data structure and coarse-to-fine mapping method are similar to OGN [25] and HSP [26]. OGN [25] introduces an octo-hash representation into a deep convolution decoder, and achieves computation- and memory-efficient generation of volumetric 3D output. HSP [26] uses a coarse-to-fine prediction on a hierarchical surface for a more accurate 3D object reconstruction. While good results of 3D object output and object reconstruction from OGN [25] and HSP [26] are presented, we integrate the efficient octo-hash data structure and coarse-to-fine plane representation into a new type of map, and apply the map to the LiDAR odometry problem, enhancing the adaptability, efficiency and accuracy of LiDAR odometry.

III. METHODOLOGY

A. Probabilistic Plane Representation

Our voxel map contains one probabilistic feature in each voxel. Without loss of generality, we use plane features due to their vast availability in environments and present the uncertainty model of a plane feature in this section.

Since a plane feature is estimated from its associated points, any noises in the points will contribute to the plane estimation uncertainty. We first investigate the point noises in Section III-A1 and then derive how the noises contribute to the plane uncertainty in Section III-A2. Notice that since the voxel map (hence the contained plane features) is represented in the world frame, the point noises should also be investigated in the world frame, which leads to two possible noise sources: one is the raw point measurement noises, which is relative to the local LiDAR body frame, and the other is the error in LiDAR pose estimation that projects the local LiDAR points to the world frame.

1) Uncertainty of Point $^W\mathbf{p}_i$: According to the analysis of the measurement noises for LiDAR sensors in [27], the uncertainty of a LiDAR point in the local LiDAR frame consists of two parts, the ranging uncertainty and the bearing direction uncertainty (see Fig. 1(a)). Let $\boldsymbol{\omega}_i \in \mathbb{S}^2$ be the measured bearing direction, $\boldsymbol{\delta}_{\boldsymbol{\omega}_i} \sim \mathcal{N}(\mathbf{0}_{2\times 1}, \boldsymbol{\Sigma}_{\boldsymbol{\omega}_i})$ be the bearing direction noise in the tangent plane of $\boldsymbol{\omega}_i$, d_i be the depth measurement and $\delta_{d_i} \sim \mathcal{N}(0, \Sigma_{d_i})$ be the ranging noise. Then the noise $\boldsymbol{\delta}_{L\mathbf{p}_i}$ of
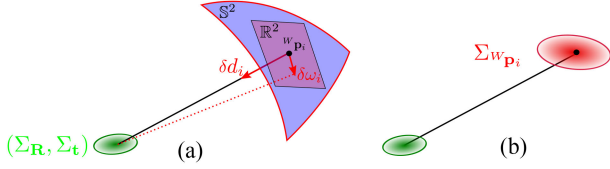
Fig. 1. The uncertainty model of the registered point: (a) the LiDAR measurement uncertainty model, including bearing direction and depth measurement uncertainty; (b) point covariance $\mathbf{\Sigma}_{W_{\mathbf{p}_i}}$ computed from (3).
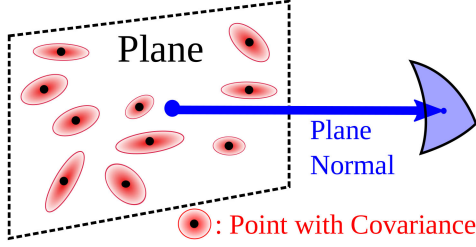

Fig. 2. The Uncertainty model of the plane normal.

the measured point $^L\mathbf{p}_i$ and its covariance $\mathbf{\Sigma}_{L_{\mathbf{p}_i}}$ is

$$\boldsymbol{\delta}_{L_{\mathbf{p}_i}} = \underbrace{\left[\boldsymbol{\omega}_i \quad -d_i\lfloor\boldsymbol{\omega}_i\rfloor_\wedge \mathbf{N}(\boldsymbol{\omega}_i)\right]}_{\mathbf{A}_i} \begin{bmatrix} \delta_{d_i} \\ \boldsymbol{\delta}_{\omega_i} \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{L_{\mathbf{p}_i}}),$$

$$\mathbf{\Sigma}_{L_{\mathbf{p}_i}} = \mathbf{A}_i \begin{bmatrix} \Sigma_{d_i} & \mathbf{0}_{1\times2} \\ \mathbf{0}_{2\times1} & \mathbf{\Sigma}_{\omega_i} \end{bmatrix} \mathbf{A}_i^T. \tag{1}$$

where $\mathbf{N}(\boldsymbol{\omega}_i) = \begin{bmatrix} \mathbf{N}_1 & \mathbf{N}_2 \end{bmatrix} \in \mathbb{R}^{3\times2}$ is an orthonormal basis of the tangent plane at $\boldsymbol{\omega}_i$, $\lfloor \ \rfloor_\wedge$ denotes the skew-symmetric matrix mapping the cross product. Detailed derivation of equation (1) can be found in [27].

Considering that the LiDAR point $^L\mathbf{p}_i$ will be further projected to the world frame through the estimated pose $^W_L\mathbf{T} = (^W_L\mathbf{R}, {^W_L\mathbf{t}}) \in SE(3)$, with estimation uncertainty $(\mathbf{\Sigma}_\mathbf{R}, \mathbf{\Sigma}_\mathbf{t})$ as shown in Fig. 1(a), by the following rigid transformation

$$^W\mathbf{p}_i = {^W_L\mathbf{R}}{^L\mathbf{p}_i} + {^W_L\mathbf{t}} \tag{2}$$

Therefore, the uncertainty of the LiDAR point $^W\mathbf{p}_i$ is hence

$$\mathbf{\Sigma}_{W_{\mathbf{p}_i}} = {^W_L\mathbf{R}}\mathbf{\Sigma}_{L_{\mathbf{p}_i}}{^W_L\mathbf{R}}^T + {^W_L\mathbf{R}}\lfloor^L\mathbf{p}_i\rfloor_\wedge\mathbf{\Sigma}_\mathbf{R}\lfloor^L\mathbf{p}_i\rfloor_\wedge^T{^W_L\mathbf{R}}^T + \mathbf{\Sigma}_\mathbf{t} \tag{3}$$

where $\mathbf{\Sigma}_\mathbf{R}$ is the uncertainty of the $^W_L\mathbf{R}$ in the tangent space and $\mathbf{\Sigma}_\mathbf{t}$ is the uncertainty of the $^W_L\mathbf{t}$. Considering the uncertainty due to LiDAR measurement and pose estimation, the covariance of point cloud at different positions has a great difference: points at closer distances have their noises dominated by the ranging noise while at further distances dominated by the bearing noise. The uncertainty analysis of a single LiDAR point is also the basis of the uncertainty model of a plane feature.

2) Plane Uncertainty Modelling: Let a plane feature consist of a group of LiDAR points $^W\mathbf{p}_i$ $(i = 1, \ldots, N)$, each has an uncertainty $^W\mathbf{\Sigma}_{\mathbf{p}_i}$ due to the measurement noise and pose estimation error as shown in Fig. 2. Denote the points covariance matrix be A:

$$\bar{\mathbf{p}} = \frac{1}{N}\sum_{i=1}^{N} {^W\mathbf{p}_i}; \quad \mathbf{A} = \frac{1}{N}\sum_{i=1}^{N} \left(^W\mathbf{p}_i - \bar{\mathbf{p}}\right)\left(^W\mathbf{p}_i - \bar{\mathbf{p}}\right)^T; \tag{4}$$
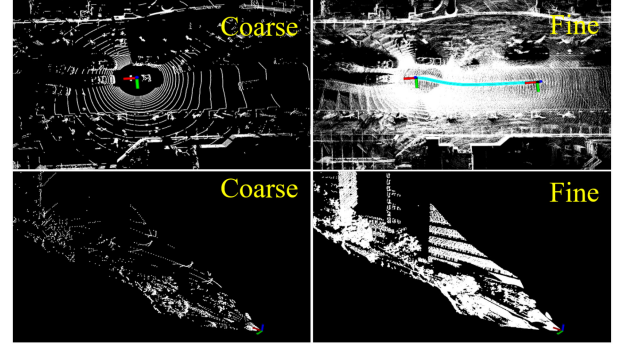

Fig. 3. The coarse-to-fine phenomenons in a spinning LiDAR (top two figures) and a non-repetitive solid-state LiDAR (bottom two figures).

Then, the plane can be represented by its normal vector $\mathbf{n}$, which is the eigenvector associated with the minimum eigenvalue of $\mathbf{A}$, and the point $\mathbf{q} = \bar{\mathbf{p}}$, which lies in this plane. As both $\mathbf{A}$ and $\bar{\mathbf{p}}$ are dependent on $^W\mathbf{p}_i$, we can denote the plane parameters $(\mathbf{n}, \mathbf{q})$ as functions of $^W\mathbf{p}_i$ shown below:

$$[\mathbf{n}, \mathbf{q}]^T = \mathbf{f}(^W\mathbf{p}_1, {^W\mathbf{p}_2}, \ldots, {^W\mathbf{p}_N}). \tag{5}$$

Based on the uncertainty analysis in III-A1, each LiDAR point $^W\mathbf{p}_i$ has a noise $\boldsymbol{\delta}_{w_{\mathbf{p}_i}} \sim \mathcal{N}(\mathbf{0}_{3\times1}, \mathbf{\Sigma}_{w_{\mathbf{p}_i}})$. Therefore, the ground-truth normal vector $\mathbf{n}^{gt}$ and ground-truth position $\mathbf{q}^{gt}$ (the real value in the real world that is to be estimated) are

$$\left[\mathbf{n}^{gt}, \mathbf{q}^{gt}\right]^T = \mathbf{f}(^W\mathbf{p}_1 + \boldsymbol{\delta}_{w_{\mathbf{p}_1}}, {^W\mathbf{p}_2} + \boldsymbol{\delta}_{w_{\mathbf{p}_1}}, \ldots, {^W\mathbf{p}_N} + \boldsymbol{\delta}_{w_{\mathbf{p}_N}})$$

$$\approx [\mathbf{n}, \mathbf{q}]^T + \sum_{i=1}^{N} \frac{\partial\mathbf{f}}{\partial^W\mathbf{p}_i}\boldsymbol{\delta}_{w_{\mathbf{p}_i}} \tag{6}$$

Here, $\frac{\partial\mathbf{f}}{\partial^W\mathbf{p}_i} = [\frac{\partial\mathbf{n}}{\partial^W\mathbf{p}_i}, \frac{\partial\mathbf{q}}{\partial^W\mathbf{p}_i}]^T$. Assume $\mathbf{A}$ has the eigenvector matrix $\mathbf{U}$, the minimum eigenvalue λ_3 and the corresponding eigenvector $\mathbf{u}_3$, referring to [28], we can take the derivative of the $\mathbf{n}$ and $\mathbf{q}$ with respect to each point $^W\mathbf{p}_i$ as below:

$$\frac{\partial\mathbf{n}}{\partial^W\mathbf{p}_i} = \mathbf{U}\begin{bmatrix} \mathbf{F}_{1,3}^{\mathbf{p}_i} \\ \mathbf{F}_{2,3}^{\mathbf{p}_i} \\ \mathbf{F}_{3,3}^{\mathbf{p}_i} \end{bmatrix}, \mathbf{F}_{m,3}^{\mathbf{p}_i} = \begin{cases} \frac{(^W\mathbf{p}_i - \mathbf{q})^T}{N(\lambda_3 - \lambda_m)}(\mathbf{u}_m\mathbf{n}^T + \mathbf{n}\mathbf{u}_m^T), m\neq3, \\ \mathbf{0}_{1\times3} \qquad\qquad , m=3. \end{cases}$$

$$\frac{\partial\mathbf{q}}{\partial^W\mathbf{p}_i} = \mathbf{diag}\left(\frac{1}{N}, \frac{1}{N}, \frac{1}{N}\right) \tag{7}$$

Then covariance matrix $\mathbf{\Sigma}_{\mathbf{n},\mathbf{q}}$ of $\mathbf{n}$ and $\mathbf{q}$ is therefore:

$$\mathbf{\Sigma}_{\mathbf{n},\mathbf{q}} = \sum_{i=1}^{N} \frac{\partial\mathbf{f}}{\partial^W\mathbf{p}_i}\mathbf{\Sigma}_{W_{\mathbf{p}_i}}\frac{\partial\mathbf{f}}{\partial^W\mathbf{p}_i}^T \tag{8}$$

It is seen that $\mathbf{n}$ and $\mathbf{q}$ are not independent as they are calculated from the same set of noisy points.

B. Coarse-to-Fine and Efficient Voxel Map Construction

In this section, we first explain the motivation for a coarse-to-fine voxel-based map and then the methodology to build and update the voxel map efficiently.

1) Motivation: As shown in Fig. 3, LiDAR points are typically sampled sequentially, hence a scan of the point cloud is always accumulated from sparse to dense, especially in outdoor
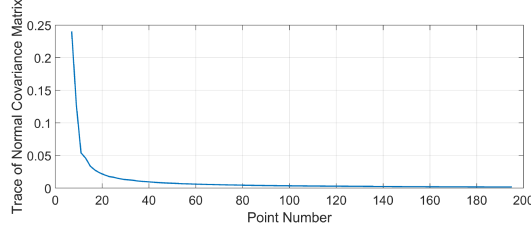
Fig. 4. The convergence of the trace of the normal vector covariance matrix with the increase of the number of points.

environments where the points are distributed in a larger space. When the point cloud is relatively sparse, common surfel-based fine-to-coarse mapping methods can usually obtain a very small number of planes only, limiting their applications to high-resolution LiDARs and relatively low scan rate (e.g., 10 Hz) such that a sufficient number of points could be accumulated. This would, however, require compensating the motion within the accumulated period. To address this issue, we propose a coarse-to-fine voxel mapping method that can build a rough voxel map when the point cloud is sparse and refine the map when more points are received.

2) Voxel Map Construction: To achieve voxel map construction in a coarse-to-fine manner, we build an adaptive voxel map organized by a Hash table and an octree for each Hash entry. More specifically, we first cut the space (in the global world frame) into voxels, each with the size of the coarse map resolution. Then, for the first LiDAR scan, which defines the world frame, the contained points are distributed into voxels. Voxels populated with points are indexed into a Hash table. Then, for each populated voxel, if all the contained points lie on a plane (the minimum eigenvalue of the point covariance matrix is less than a specified threshold), we store the plane points and calculate the plane parameters $(\mathbf{n}, \mathbf{q})$ as in (5) and their uncertainty $\boldsymbol{\Sigma}_{\mathbf{n},\mathbf{q}}$ as in (8); otherwise, the current voxel will break into eight octants and repeat plane checking and voxel cutting in each one until reaching the maximum number of layers. Notice that the voxels have different sizes, each voxel contains one plane feature fitted from the contained LiDAR raw points.

3) Voxel Map Update: For online LiDAR odometry, new scans of LiDAR point cloud are continuously coming and their poses are estimated as in Section III-D. Then the estimated pose is used to register the new points into the global map. When the new points lie in an unpopulated voxel, it will construct the voxel. Otherwise, when the new points are added to an existing voxel, the parameters and the uncertainty of the plane in the voxel should be updated. This will cause a growing processing time with the constant reception of new points. To address this issue, we find that the uncertainty of the plane parameters will quickly converge as illustrated in Fig. 4, where the position of each point carries a Gaussian noise with a zero mean and a variance of $0.1 \ m^2$. It is seen that the uncertainty of the normal vector converges when the number of points reaches 50. After the uncertainty converges, we discard all historical points and retain the estimated plane parameters $(\mathbf{n}, \mathbf{q})$ and covariance $\boldsymbol{\Sigma}_{\mathbf{n},\mathbf{q}}$. Once more new points are coming, we keep only the latest 10 points and calculate the new plane normal vector composed of those 10 points. If the new normal vector and the previously converged normal vector continue to appear a relatively large difference, we assume that this area of the map has changed and needs to be reconstructed as Section III-B2.

C. Point-to-Plane Match

This section details how to match points in a new LiDAR scan with the voxel map to construct the constrain for the pose estimation and subsequent point cloud registration. Based on the accurate uncertainty modeling of the points and planes, we could easily implement the point-to-plane scan match. Given a LiDAR point ${}^{W}\mathbf{P}_i$ predicted in the world frame with the pose prior, we first find which root voxel (with the coarse map resolution) it lies in by its Hash key. Then, all the contained sub-voxels are polled for a possible match with the point. Specifically, let a sub-voxel contains a plane with normal $\mathbf{n}_i$ and center $\mathbf{q}_i$, we calculate the point-to-plane distance:

$$d_i = \mathbf{n}_i^T({}^{W}\mathbf{P}_i - \mathbf{q}_i) \tag{9}$$

As analyzed above, the normal vector $\mathbf{n}_i$, the LiDAR point ${}^{W}\mathbf{P}_i$, and the center $\mathbf{q}_i$ have uncertainties. Considering all these uncertainties, we obtain:

$$
\begin{aligned}
d_i &= (\mathbf{n}_i^{gt} \boxplus \boldsymbol{\delta}_{\mathbf{n}_i})^T [({}^{W}\mathbf{P}_i^{gt} + \boldsymbol{\delta}_{{}^{W}\mathbf{P}_i}) - \mathbf{q}_i^{gt} - \boldsymbol{\delta}_{\mathbf{q}_i}] \\
&\approx \underbrace{\mathbf{n}_i^{gt\,T}({}^{W}\mathbf{P}_i^{gt} - \mathbf{q}_i^{gt})}_{\mathbf{0}} + \underbrace{\mathbf{J}_{\mathbf{n}_i}\boldsymbol{\delta}_{\mathbf{n}_i} + \mathbf{J}_{\mathbf{q}_i}\boldsymbol{\delta}_{\mathbf{q}_i} + \mathbf{J}_{{}^{W}\mathbf{P}_i}\boldsymbol{\delta}_{{}^{W}\mathbf{P}_i}}_{\mathbf{w}_i}
\end{aligned}
$$

which implies

$$d_i \sim \mathcal{N}(0, \Sigma_{\mathbf{w}_i}), \quad \Sigma_{\mathbf{w}_i} = \mathbf{J}_{\mathbf{w}_i} \Sigma_{\mathbf{n}_i, \mathbf{q}_i, {}^{W}\mathbf{P}_i} \mathbf{J}_{\mathbf{w}_i}^T, \tag{10}$$

where,

$$\mathbf{J}_{\mathbf{w}_i} = \left[\mathbf{J}_{\mathbf{n}_i}, \mathbf{J}_{\mathbf{q}_i}, \mathbf{J}_{{}^{W}\mathbf{P}_i} \right] = \left[({}^{W}\mathbf{P}_i - \mathbf{q}_i)^T, -\mathbf{n}_i^T, \mathbf{n}_i^T \right]$$

$$\Sigma_{\mathbf{n}_i, \mathbf{q}_i, {}^{W}\mathbf{P}_i} = \begin{bmatrix} \Sigma_{\mathbf{n}_i, \mathbf{q}_i} & 0 \\ 0 & \Sigma_{{}^{W}\mathbf{P}_i} \end{bmatrix}. \tag{11}$$

That is being said, if the point lies on the candidate plane, its distance d_i should be subject to the distribution in (10). Therefore, let the standard deviation of the distribution be $\sigma = \sqrt{\Sigma_{\mathbf{w}_i}}$, we can examine whether the measured point-to-plane distance falls within 3σ. If so, it is selected to be an effective match. Besides, if a point matches more than one plane based on the 3σ criterion, the plane with the highest probability will be matched. If no plane passes the 3σ test, the point is discarded to remove the possible false matches caused by voxel quantization.

D. State Estimation

We build a LiDAR(-inertial) odometry system based on an iterated extended Kalman filter similar to FAST-LIO2 [13]. Assume that we are given a state estimation prior $\hat{\mathbf{x}}_k$ with covariance $\hat{\mathbf{P}}_k$. The prior can be provided from a constant velocity assumption for LiDAR-only Odometry (experiment A and B) or IMU propagation for LiDAR-inertial Odometry (experiment C). This prior will be fused with the point-to-plane distance matched in Section III-C to form a maximum a posteriori (MAP) estimation. Specifically, the i-th valid point-to-plane match leads to the observation equation

$$\mathbf{z}_i = \mathbf{h}_i(\mathbf{x}_k) + \mathbf{v}_i \tag{12}$$

Sensor	Type	Frequency	Scan pattern	FoV
HDL-64E S2	Mechanical	10Hz	repetitive	$360° \times 32°$
Realsense L515	Solid-state	30Hz	repetitive	$70° \times 55°$
Livox Avia	Solid-state	10Hz	non-repetitive	$70° \times 77°$

where $\mathbf{z}_i$ is the point-to-plane distance residual d_i in equation (9), $\mathbf{h}_i(\mathbf{x}_k)$ is the observation function and $\mathbf{v}_i \sim (0, \mathbf{R}_i)$ is the observation noise. By substituting the state $\mathbf{x}_k$ (i.e., the sensor pose $\mathbf{T}_k$) into (9) and linearize it around the current state update $\bar{\mathbf{x}}_k$, we obtain:

$$\mathbf{z}_i = \mathbf{h}_i(\mathbf{x}_k) + \mathbf{v}_i = \mathbf{n}_i^T({}^W\mathbf{p}_i - \mathbf{q}_i)$$

$$= (\mathbf{n}_i^{gt} \boxplus \boldsymbol{\delta}_{\mathbf{n}_i})^T[(\mathbf{T}_k^{gt} \boxplus \boldsymbol{\delta}_{\mathbf{T}_k})({}^L\mathbf{p}_i^{gt} + \boldsymbol{\delta}_{{}^L\mathbf{p}_i}) - \mathbf{q}_i^{gt} - \boldsymbol{\delta}_{\mathbf{q}_i}]$$

$$\approx \underbrace{\mathbf{n}_i^{gt\,T}(\mathbf{T}_k^{gt\,L}\mathbf{p}_i^{gt} - \mathbf{q}_i^{gt})}_{\mathbf{0}} + \underbrace{\mathbf{J}_{\mathbf{T}_i}\boldsymbol{\delta}_{\mathbf{T}_k}}_{\mathbf{H}_i\delta\mathbf{x}_k}$$

$$+ \underbrace{\mathbf{J}_{\mathbf{n}_i}\boldsymbol{\delta}_{\mathbf{n}_i} + \mathbf{J}_{\mathbf{q}_i}\boldsymbol{\delta}_{\mathbf{q}_i} + \mathbf{J}_{{}^L\mathbf{p}_i}\boldsymbol{\delta}_{{}^L\mathbf{p}_i}}_{\mathbf{v}_i}$$

which implies

$$\mathbf{R}_i = \mathbf{J}_{\mathbf{v}_i} \boldsymbol{\Sigma}_{\mathbf{n}_i, \mathbf{q}_i, {}^L\mathbf{p}_i} \mathbf{J}_{\mathbf{v}_i}^T, \qquad (13)$$

where

$$\mathbf{J}_{\mathbf{v}_i} = [\mathbf{J}_{\mathbf{n}_i}, \mathbf{J}_{\mathbf{q}_i}, \mathbf{J}_{{}^L\mathbf{p}_i}] = [(\bar{\mathbf{T}}_k{}^L\mathbf{p}_i - \mathbf{q}_i)^T, -\mathbf{n}_i^T, \mathbf{n}_i^T\bar{\mathbf{R}}_k]$$

$$\boldsymbol{\Sigma}_{\mathbf{n}_i, \mathbf{q}_i, {}^L\mathbf{p}_i} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{n}_i, \mathbf{q}_i} & 0 \\ 0 & \boldsymbol{\Sigma}_{{}^L\mathbf{p}_i} \end{bmatrix}.$$

Finally, combining the state prior with all effective measurements, we can obtain the MAP estimation:

$$\min_{\mathbf{x}_k} \Big(\|\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k\|_{\hat{\mathbf{P}}_k}^2 + \sum_{i=1}^{m} \|d_i - \mathbf{H}_i \cdot (\mathbf{x}_k \boxminus \bar{\mathbf{x}}_k)\|_{\mathbf{R}_i}^2 \Big) \quad (14)$$

where the first part is the state prior and the second part is the measurement observation.

IV. EXPERIMENTS

In this section, in order to show the high accuracy, efficiency, and adaptability of our mapping method to different environments and LiDAR types, we conduct experiments on three typical LiDARs in three different environments (urban environment, indoor environment, and unstructured outdoor environment), respectively. The details of the LiDAR we used are shown in Table I. In each experiment, we compare our method with state-of-the-art counterparts. All the experiments are conducted on a desktop computer with Intel i7-10700 @ 2.9 GHz with 16 GB RAM and Nvidia GeForce GTX 730 with 2 GB RAM (in case a GPU is used).

A. Urban Environment Test

In this experiment, we evaluate our approach on the odometry datasets of the KITTI Vision Benchmark [2], where the LiDAR data is collected by Velodyne HDL-64E S2 in urban environments and the in-frame motion has been compensated in advance. As in [29], a vertical angle of $0.22°$ is employed to correct the calibration errors in raw point clouds from the KITTI dataset.

Comparison with state-of-the-art: We compare our method with FAST-LIO2 [13], a point-cloud map-based LiDAR-inertial odometry approach, MULLS [30], a multi-feature-based LO system, LiTAMIN2 [24], an improved ICP-NDT hybrid method, Suma [14], a surfel-based LiDAR SLAM approach, and Lego-Loam [7], a ground-optimized LiDAR odometry and mapping system.

For the implementation, we run our algorithm on all sequences with the same parameters, where the maximum voxel size is 3 m and the maximum octo-tree layer is 3, leading to a minimum voxel size of 0.375 m. The full version of the algorithm is denoted as *Ours (full)*. To verify the effectiveness of the adaptive voxel module and features probabilistic module, we use a fixed size voxel (i.e., 2 m) (termed as *Ours (w/o adaptive)*) and a zero plane uncertainty in (8) (termed as *Ours (w/o prob.)*), respectively, while retaining all other modules with the same parameter values. For FAST-LIO2, it is designed for LiDAR-inertial fusion, where IMU propagation is served as a motion prior. Since the KITTI dataset does not have IMU data, we use a constant velocity model similar to ours (see (14)) to serve as the motion prior in FAST-LIO2 for a fair comparison. The other parameters of FAST-LIO2 are their default values recommended for multi-line spinning LiDARs in outdoor environments. For MULLS, we conduct the experiments using the open-sourced implementation with the default parameters. For LiTAMIN2, we directly use the results presented in the original paper [24]. The work in LiTAMIN2 [24] further benchmarked their results with Suma [14] and Lego-Loam [7], both with loop closure and without loop closure. We directly draw these results and compare them with ours. Since our implementation is a LiDAR odometry without any loop closure, for the sake of fair comparison, we only compare with the results where loop closure for MULLS, LiTAMIN2, Suma, and Lego-Loam are off.

Results and Discussion: Table II shows the absolute trajectory error (ATE) [31] of all methods, where the results for LiTAMIN2, Suma, and Lego-Loam are directly drawn from [24]. As can be seen, our method (*Ours (full)*) has the best overall accuracy than all other algorithms, especially in long-term sequences like 00, 02 and 08. To further show the high accuracy of our method, we plot the estimated LiDAR trajectories of all sequences and their ground-truth in Fig. 5. It is seen that all of our global trajectories are very close to the ground truth, verifying the high accuracy of our method. Notice that all variants of our methods use the same parameters across all sequences.

The improvement of our is mainly attributed to the adaptive voxelization and probabilistic plane representation of the map. The effectiveness of the adaptive voxelization can be seen from the performance degradation from *Ours (full)* to *Ours (w/o adaptive)*. With the adaptive voxelization, our method is able to capture plane features of different sizes (both large-size and finer-level small planes) in the environment. The effectiveness of the probabilistic plane representation can be seen from the performance degradation from *Ours (full)* to *Ours (w/o prob.)*. Without the probabilistic plane representation, the performance of our method is at the same level of FAST-LIO2, which is reasonable because FAST-LIO2 also uses a deterministic plane representation and the same IKFoM framework [32]. Compared to the adaptive voxelization, the probabilistic plane representation plays a much more significant role in the accuracy since the performance degradation by turning off the module is much

TABLE II
ACCURACY (ATE IN METERS) COMPARISON ON KITTI ODOMETRY TRAINING SEQUENCES

Approach (Total length [m])	00 (3724)	01 (2453)	02 (5067)	03 (560)	04 (393)	05 (2205)	06 (1232)	07 (694)	08 (3222)	09 (1705)	10 (919)	Avg. of all scans [deg]/[m]
Ours (full)	**0.9**/2.8	1.9/7.8	**1.7**/6.1	1.2/**0.7**	0.6/**0.3**	0.8/1.2	**0.4**/**0.4**	0.7/0.7	**1.1**/2.3	**1.0**/1.9	1.0/1.1	**1.2**/2.9
Ours (w/o adaptive)	1.1/3.1	2.0/8.2	1.7/7.1	1.2/0.7	0.7/0/5	0.8/1.3	0.5/0.5	0.9/0.7	1.8/3.1	1.1/2.0	1.3/1.2	1.3/3.4
Ours (w/o prob.)	2.2/3.9	2.5/9.5	1.9/8.1	1.4/0.9	0.7/0.4	1.0/1.2	0.5/0.6	1.0/0.6	3.0 /6.8	1.2/1.9	1.2/1.5	1.8/4.5
LiTAMIN2 [24]	1.6/5.8	3.5/15.9	2.7/10.7	2.6/0.8	2.3/0.7	1.1/2.4	1.1/0.9	1.0/0.6	1.3/2.5	1.7/2.1	1.2/**1.0**	1.8/5.1
MULLS [30]	1.7/6.1	**1.0**/**2.4**	2.4/10.6	**0.7**/0.7	**0.2**/0.9	1.0/2.4	0.4/0.6	**0.5**/0.6	1.9/4.3	1.4/2.5	**0.5**/1.1	1.5/4.8
Suma [14]	1.0/2.9	3.2/13.8	2.2/8.4	1.5/0.9	1.8/0.4	**0.7**/1.2	0.4/0.4	0.7/**0.5**	1.5/2.8	1.1/2.9	0.8/1.3	1.4/3.9
FAST-LIO2 [13]	2.0/3.8	2.5/11.9	2.8/12.8	1.2/0.8	0.7/0.5	1.2/2.1	1.1/1.2	1.2/0.8	1.8/3.0	1.4/2.0	1.1/1.5	1.8/4.9
Lego-Loam [7]	2.8/6.3	3.8/119.4	4.1/14.7	4.1/0.9	3.3/0.8	1.9/2.8	1.4/0.8	1.5/0.7	2.5/3.5	2.2/2.1	1.9/1.8	2.8/11.1

■ Ours
■ Ground Truth
● Start Point
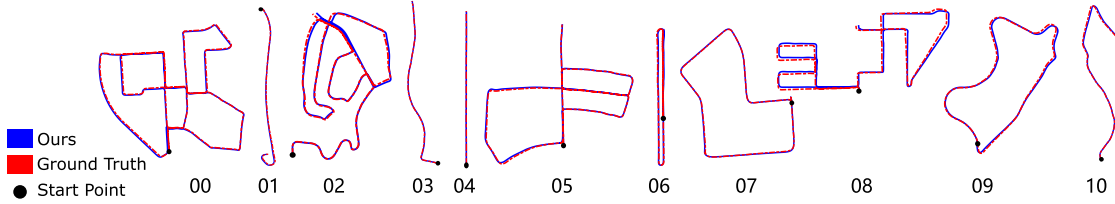00 01 02 03 04 05 06 07 08 09 10

Fig. 5. Trajectories of sequence 00 to 10. The blue trajectories are the camera paths obtained from our method and the red trajectories are the gournd truth. The black dot represents the starting point.
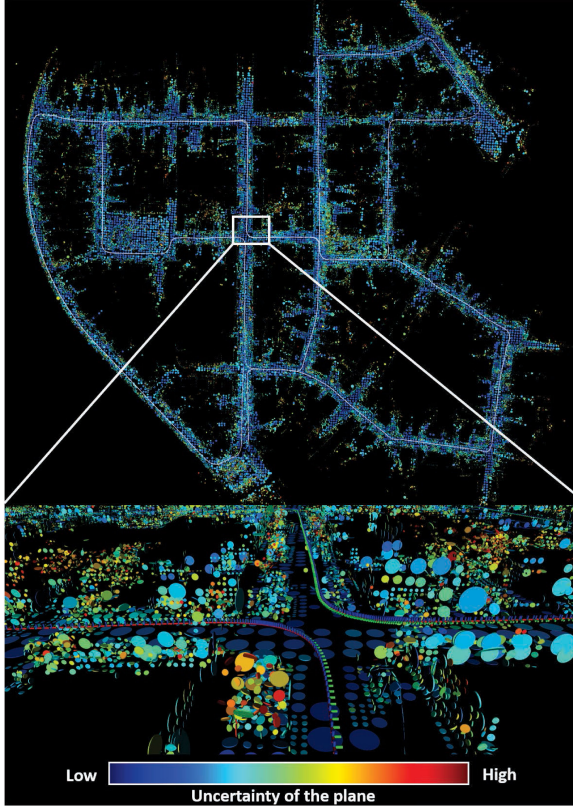

Low ——— High
Uncertainty of the plane

Fig. 6. Illustration of plane features in our adaptive voxel maps on KITTI training sequence 00. The plane feature is colored by its uncertainty(the trace of the plane covariance matrix in equatioin (8)). The plane is displayed as a disk with radius determined by the maximum eigen value of the points covariance matrix **A** defined in equation (4).

TABLE III
COMPARISON OF TIME CONSUMPTION PER SCAN

Approach	Ours (full)	MULLS [30]	Suma [14]	FAST-LIO2 [13]	Lego-Loam [7]
mean (ms)	**26.45**	67.82	129.36	29.11	57.13
std (ms)	**9.87**	27.52	62.22	10.94	21.52

of different sizes: planes on the ground and walls are usually large while those on cluttered objects (e.g., moving cars, road-side objects) and their nearby regions are small. Plane features contained in the voxel map are also colored by their respective uncertainty (the trace of the plane covariance matrix (8) in Fig. 6. As can be seen, plane features far away from the road have much higher uncertainty than that nearby. This is caused by the uneven LiDAR point measurements, where nearby areas have many more measurements than far areas due to the repeat observation. We also notice that plane features in cluttered areas have higher uncertainties than that in structured areas (e.g., ground, wall) and that plane features on moving objects (e.g., cars) have higher uncertainties than static ones. All these phenomena are as expected.

Runtime: Besides accuracy, we also evaluate the computation efficiency of our method against others. Since the LiTA-MIN2 [24] is not open-sourced, we compare our efficiency with MULLS, Suma, FAST-LIO2 and Lego-Loam. For Suma and Lego-loam, we use their own implementation on Github with the default parameters. Similar to the accuracy evaluation, all loop-closure modules of MULLS, Suma and Lego-Loam are completely turned off to avoid unnecessary computations. Suma uses GPUs to accelerate, while other methods only use CPU. The statistics of time consumption per LiDAR scan averaged over all KITTI training sequences are summarized in Table III Suma takes more time in our experiment than in the original paper [14] due to the difference in GPU performance. As can be seen, our method has the lowest processing time, and the next is FAST-LIO2 [13].When compared to MULLS [30], we do not need feature extraction and save the time therein. When compared to FAST-LIO2 [13] and Lego-Loam [7], we use the voxel map described in III-B2 to avoid building a kd-tree in real-time [13] or periodically [7]. Moreover, searching neighbor points on kd-trees has a time complexity $O(m \log(n))$, where m

larger. Finally, it is also interesting to see that our method with only probabilistic plane representation *Ours (w/o adaptive)* can even exceed all comparison methods in most sequences.

Fig. 6 illustrates the plane features estimated in our voxel maps. As can be seen, our voxel map contains plane features

TABLE IV
COMPARISON RESULTS ON L515 DATASETS

Method	End-to-End error (m) / Avg. Comp. Time (ms)		
	Lab1 (25.9m)	Lab2 (32.6m)	Warehouse (36.2m)
Ours (full)	**0.02/7.18**	**0.02/7.26**	**0.01/6.38**
SSL_SLAM [33]	0.71/27.7	5.80/32.9	0.93/28.3

TABLE V
COMPARISON RESULTS ON LIVOX AVIA DATASETS

Method	End-to-End error (m) / Avg. Comp. Time (ms)			
	Park1 (485m)	Park2 (485m)	Park3 (815m)	Mountain (3490m)
Ours (full)	**0.03/8.1**	**0.02/8.3**	**0.01/7.7**	**6.9**/34.2
Faster-LIO [35]	0.88/9.8	0.99/9.8	0.07/9.9	9.8/**32.0**
FAST-LIO2 [13]	1.09/12.9	0.76/13.4	0.08/12.9	7.1/44.3

is the dimension and n is the number of points in the dense point cloud map. In contrast, the speed of searching adjacent planes in the voxel map is near $O(1)$, which ensures efficiency.

B. Indoor Environment Test

In this experiment, we test our method in indoor environments with a solid-state LiDAR Intel L515. As in experiment A, a constant velocity model is used to provide the motion prior. We use a handheld device to collect three groups of LiDAR data in laboratory and warehouse environments. All data collection routes start and end intentionally at the same place. We compare our method with SSL_SLAM [33], a LiDAR SLAM approach designed for this type of LiDAR. We run our algorithm on all datasets with the same parameters, where the maximum voxel size is 0.5 m and the maximum octo-tree layer is 2. For SSL_SLAM, we conduct the experiments using the open-sourced implementation with the default parameters. Table IV records the end to end error and average calculation time per scan of our method and SSL_SLAM. Compared with SSL_SLAM, our method avoids feature extraction and considers the uncertainty of both the LiDAR point and plane map, resulting in better accuracy and efficiency than SSL_SLAM. We also test the two methods on the dataset used by the original SSL_SLAM paper [33], and the results are comparable (0.12 m / 5.1 ms for our method and 0.15 m / 25.1 ms for SSL_SLAM [33]). The small improvement is mainly attributed to the less challenging slow motion in the data, which was collected on a UGV moving quite stably and slowly. In contrast, the three sequences in this letter are collected on hands with much faster motions (especially rotations). Through the comparison, we see that our method is more robust to motion speed.

We further show the aggregated colored point cloud map of our method and SSL_SLAM in the warehouse environment in Fig. 1 of the supplementary material [34]. It can be seen that our colored point cloud map is more precision, while SSL_SLAM's is more blurred, especially for the area near the return route.

C. Unstructured Environment Test

Both experiments in Section IV-A and Section IV-B are carried out in structural environments. To further verify the adaptability and applicability of our method in unstructured environments, we conduct experiments in a park and a mountain environment using a non-repetitive scanning solid-state LiDAR Livox Avia. The sensor has a built-in IMU and together they provide 10 Hz LiDAR data and 200 Hz IMU data. We use the built-in IMU to compensate the motion distortion and provide motion prior similar to FAST-LIO2 [13]. In these experiments, the same computation platform and parameters (the maximum voxel size is 3 m and the maximum octo-tree layer is 3) as in Section IV-A are used despite the drastically different LiDAR scanning patterns and environment structures. In order to validate the advantages of our map structure, we compare our results with Faster-LIO [35] and FAST-LIO2 [13]. The three methods

use the same IKFoM framework [32] and plane features, except for the map structure. Specifically, our method uses a probabilistic adaptive voxel map, Faster-LIO uses a Hash-indexed iVoxel map, and FAST-LIO2 uses an incremental k-d tree. For a fair comparison, the default 1:3 downsample used in Faster-LIO and FAST-LIO2 are turned off, so all raw points are used in all three methods. The other settings are kept default as in their open release. All results are averaged over 5 runs.

1) Park Environment: In this experiment, we use a handheld device to collect the LiDAR and IMU data in a large park filled with trees. In order to show the drift, the data collection route starts and ends at the same place. Besides, to avoid randomness, we collect the data of three different trajectories, of which two groups are 485 m long around the periphery of the park, and one group is 815 m long around the periphery and the interior of the park. The mapping result of the park with the longest trajectory is shown in Fig. 2 of the supplementary material [34].

2) Mountain Environment: In this experiment, we use a UAV to carry our LiDAR and face the LiDAR to the ground to collect point cloud data on a mountain. Compared to the park, the mountain environment is even less structured and more challenging. The final mapping results and some partially enlarged views can be seen in Fig. 3 of the supplementary material [34]. We can clearly see the large planes with low uncertainty in the highway between mountains and small planes with high uncertainties on vegetation areas.

3) Result Analysis: The end to end error and average computation time per scan of the three methods under comparison are shown in Table V. As can be seen, our method achieves much higher accuracy than both Faster-LIO and FAST-LIO2 in all three sequences, mainly due to the accurate consideration of plane uncertainty in the map where Faster-LIO and FAST-LIO2 both assume deterministic plane maps. In terms of computation efficiency, both our method and Faster-LIO use a Hash table to organize the map, so they both outperform FAST-LIO2 which uses a variant kd-tree as explained in Section IV-A. When compare among our method and Faster-LIO, our method requires an additional calculation of the probability distribution of each point and the plane. While this additional calculation requires more computation time, the computed probabilistic information enables to reject further outliers as detailed in Section III-C. So the computation time of our method and Faster-LIO is more or less comparable (see Table V).

V. CONCLUSION

this letter proposes an efficient, probabilistic adaptive voxel mapping method for online LiDAR odometry. An accurate uncertainty model of the LiDAR point and plane is proposed by considering both LiDAR measurements noises and sensor pose estimation errors. The voxel map is constructed using coarse-to-fine adaptive resolution method, which is robust to different Li-DAR scanning patterns and environments. this letter also shows how to implement the proposed mapping method in an iterated

extended Kalman filter-based LiDAR(-inertial) odometry. The tests on the KITTI dataset show that our method can achieve better performance than all point cloud-based, NDT-based, and surfel-based methods. The proposed voxel mapping method not only performs well in the spinning LiDAR in structured urban environments, but also shows good performance with solid-state LiDAR L515 in indoor environments and non-repetitive small FoV LiDAR in unstructured environments such as parks and mountains. A direction to improve this work is to add additional features such as edge to the voxel to expand the universality of the proposed map representation method.

REFERENCES

[1] Z. Liu, F. Zhang, and X. Hong, "Low-cost retina-like robotic LiDARs based on incommensurable scanning," *IEEE/ASME Trans. Mechatron.*, vol. 27, no. 1, pp. 58–68, Feb. 2022.

[2] 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.

[3] W. Xu and F. Zhang, "FAST-LIO: A fast, robust LiDAR-inertial odometry package by tightly-coupled iterated Kalman filter," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3317–3324, Apr. 2021.

[4] Y. Okubo, C. Ye, and J. Borenstein, "Characterization of the hokuyo urg-04 lx laser rangefinder for mobile robot obstacle negotiation," *Proc. SPIE*, vol. 7332, 2009, Art. no. 733212.

[5] R. Ravi, A. Habib, and D. Bullock, "Pothole mapping and patching quantity estimates using LiDAR-based mobile mapping systems," *Transp. Res. Rec.*, vol. 2674, no. 9, pp. 124–134, 2020.

[6] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time," in *Proc. Robot.: Sci. Syst.*, 2014, vol. 2, no. 9, pp. 1–9.

[7] T. Shan and B. Englot, "Lego-loam: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2018, pp. 4758–4765.

[8] 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 *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 8899–8906.

[9] 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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 5135–5142.

[10] K. Li, M. Li, and U. D. Hanebeck, "Towards high-performance solid-state-LiDAR-inertial odometry and mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 5167–5174, Jul. 2021.

[11] G. C. Sharp, S. W. Lee, and D. K. Wehe, "ICP registration using invariant features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 1, pp. 90–102, Jan. 2002.

[12] J. Servos and S. L. Waslander, "Multi channel generalized-ICP," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 3644–3649.

[13] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct LiDAR-inertial odometry," *IEEE Trans. Robot.*, to be published, doi: 10.1109/TRO.2022.3141876.

[14] J. Behley and C. Stachniss, "Efficient surfel-based SLAM using 3D laser range data in urban environments," in *Proc. Robot.: Sci. Syst.*, vol. 2018, 2018, p. 59.

[15] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma : Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4530–4537.

[16] D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled LiDAR-visual-inertial odometry," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1004–1011, Apr. 2021.

[17] D. Droeschel, M. Schwarz, and S. Behnke, "Continuous mapping and localization for autonomous navigation in rough terrain using a 3D laser scanner," *Robot. Auton. Syst.*, vol. 88, pp. 104–115, 2017.

[18] 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. (IROS)*, 2021, pp. 5499–5506.

[19] J. Stückler and S. Behnke, "Multi-resolution surfel maps for efficient dense 3D modeling and tracking," *J. Vis. Commun. Image Representation*, vol. 25, no. 1, pp. 137–147, 2014.

[20] C. Park, S. Kim, P. Moghadam, C. Fookes, and S. Sridharan, "Probabilistic surfel fusion for dense LiDAR mapping," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2017, pp. 2418–2426.

[21] 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.

[22] M. Magnusson, A. Lilienthal, and T. Duckett, "Scan registration for autonomous mining vehicles using 3D-NDT," *J. Field Robot.*, vol. 24, no. 10, pp. 803–827, 2007.

[23] J. Saarinen, H. Andreasson, T. Stoyanov, J. Ala-Luhtala, and A. J. Lilienthal, "Normal distributions transform occupancy maps: Application to large-scale online 3D mapping," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2013, pp. 2233–2238.

[24] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "Litamin2: Ultra light LiDAR-based SLAM using geometric approximation applied with kl-divergence," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 11619–11625.

[25] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2088–2096.

[26] C. Häne, S. Tulsiani, and J. Malik, "Hierarchical surface prediction for 3D object reconstruction," in *Proc. Int. Conf. 3D Vis.*, 2017, pp. 412–420.

[27] C. Yuan, X. Liu, X. Hong, and F. Zhang, "Pixel-level extrinsic self calibration of high resolution LiDAR and camera in targetless environments," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 7517–7524, Oct. 2021.

[28] Z. Liu and F. Zhang, "BALM: Bundle adjustment for LiDAR mapping," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 3184–3191, Apr. 2021.

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

[30] 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.

[31] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 7244–7251.

[32] D. He, W. Xu, and F. Zhang, "Kalman filters on differentiable manifolds," 2021. [Online]. Available: https://arxiv.org/abs/2102.03804

[33] H. Wang, C. Wang, and L. Xie, "Lightweight 3-D localization and mapping for solid-state LiDAR," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1801–1807, Apr. 2021.

[34] C. Yuan, W. Xu, X. Liu, X. Hong, and F. Zhang, "Supplementary material: Efficient and probabilistic adaptive voxel mapping for accurate online LiDAR odometry," to be published, doi: 10.1109/LRA.2022.3187250.

[35] 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 Robot. Automat. Lett.*, vol. 7, no. 2, pp. 4861–4868, Apr. 2022.