Inertial Aided 3D LiDAR SLAM with Hybrid Geometric Primitives in Large-scale Environments

Wen Chen, Hongchao Zhao, Qi Shen, Chao Xiong, Shunbo Zhou, Yun-Hui Liu

Abstract— This paper presents a comprehensive inertial aided 3D LiDAR SLAM system with hybrid geometric primitives in large-scale environments, including a tightly-coupled LiDAR-Inertial-Odometry (LIO), a global mapping module supported by learning-based loop closure detection and a sub-maps matching algorithm. An efficient method is developed to simultaneously extract explicit plane features and point features from each raw point cloud. To make full use of the structural information of the surroundings, plane features and point features (ground and edge) are tracked across a fix-sized group of LiDAR keyframes in the local map. For effective loop closure detection in large-scale environments, we integrate the learning-based point cloud network and a keyframe sequence matching method to detect loops. Finally, a novel, deterministic and near real-time plane-driven sub-maps matching algorithm is proposed to close the loops. The proposed SLAM system is validated with experiments on different types of environments.

I. INTRODUCTION

The technology of estimating the robot states and simultaneously modeling the environments is the well-known Simultaneous Localization and Mapping (SLAM). In general, the sensors that are usually used in real-world applications can be divided into two types: proprioceptive and exteroceptive sensors [1]. Inertial Measurement Unit (IMU) is a typical proprioceptive sensor, as it can capture the motion of the robot without any information from the surroundings. But the rapid drifting is their biggest drawback. On the other hand, the exteroceptive sensors, such as 3D LiDAR, provide accurate range information of the surroundings. However, motion distortion and sparsity of the point clouds result in difficulty in feature extraction and association tasks. Especially, the LiDAR-only methods suffer from degenerated cases in structure-less scenarios. Thus, the fusion of these two types of sensors is vital to provide robust and accurate results of SLAM in real applications.

The combination of 3D LiDAR and inertial sensor is a research highlight. The most direct fusion way is utilizing the IMU integration results to interpolate the moving trajectory for the motion distortion compensation [2] [3]. In [4] [5], the IMU measurement deviations are minimized along with the surfel match errors and initial condition constraints in a nonlinear optimization framework. Inspired by the successful

applications of preintegration theorem in VIO [6] [7], Ye et al. [8] proposed a tightly-coupled LiDAR-IMU odometry algorithm by jointly optimizing the relative LiDAR factors and IMU preintegration factors in a sliding-window. To improve efficiency, Shan et al. [9] adopted performing scan-matching with a fixed-size sub-keyframes in local scale rather than matching LiDAR features with a global map.

Most of the above methods adopted the point feature based methods to improve the computational efficiency instead of using the whole point cloud. Recently, higher geometric primitives, like planes, have attracted a lot of researchers because they can leverage more structural information of the environments [10] [11]. However, utilizing 3D planes (only three degrees of freedom) in the SLAM framework is not a trivial task, as the standard plane model (Hesse model) is over-parametrized and is normally represented by a four-dimension vector. Several algorithms have been proposed to remove the singularity. Weingarten et al. [12] proposed to use three steps to fit the plane from points: initial plane fitting with PCA; points transformed into global xy-plane; uncertainty analysis using standard regression methods. Kaess et al. [13] presented the unit quaternion representation for infinite planes detected in RGB-D sensor. Hsiao et al. extended the work to keyframe-based SLAM [14] and incorporated the IMU measurements [15]. Geneva et al. [16] proposed the closest point representation based singularity free plane factor in the LiDAR-inertial 3D plane SLAM. However, the plane finding and fitting are performed off-line, and only simulation results are provided.

Finding planes from sparse point clouds is not an easy task. The techniques can be categorized into Hough-based [17], region-growing [18], or RANSAC-based [19] [20] approaches. The Hough-based and region-growing methods have higher probability of detecting these planes among dense point clouds. On the other hand, RANSAC based methods either attempt to directly fit planes or merge higher-level analysis to find better fits. The high computational cost makes it not usable in real-time applications. Serafin et al. [21] proposed a fast algorithm to extract line or plane features by three steps: ground points removal; surface normal computation for each point; segmentation and fitting.

Apart from the difficulties in the fusion of the measurements from LiDAR and IMU in the front-end odometry part, another challenge is the LiDAR-based loop closure for global optimization after long-term running, which can eliminate the accumulated drift error. Most of the recent 3D LiDAR SLAM systems [3] [22] [23] employ a simple loop-closure detection method based on the poses proximity, followed
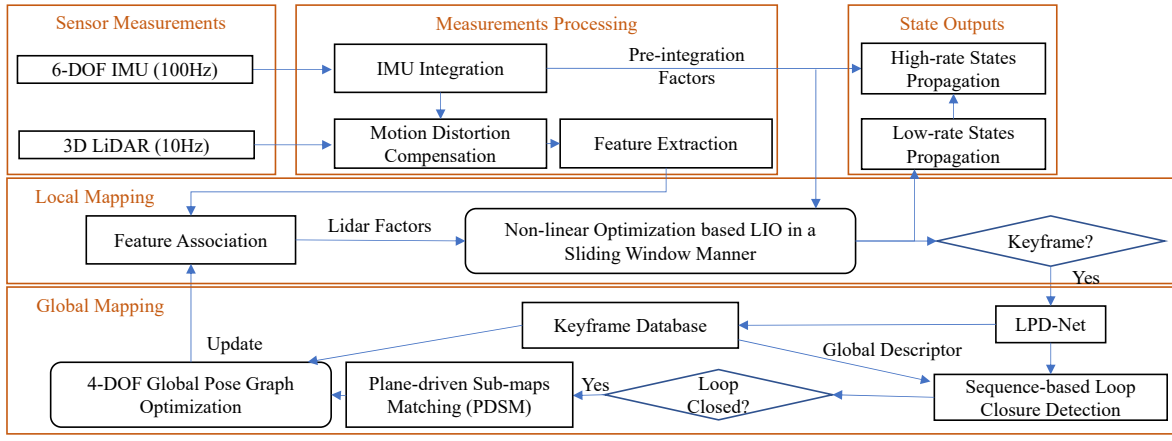
Fig. 1. The pipeline of the proposed inertial-aided 3D LiDAR SLAM system.

by the ICP-based methods to compute the relative transformation between the pair-wise point clouds of the newly added keyframe and the loop-closed keyframe. However, the poses proximity strategy is not reliable in large-scale environments, in which the accumulated drift error of the front-end odometry may be too large to detect loops efficiently. What's more, the locally optimal nature of the ICP-based point cloud registration methods makes it very adventurous to add the loop closure factors in the global pose graph optimization. On the other hand, although the branch-and-bound (BnB) based 3-DOF parameters searching approach for the scan-to-submap matching successfully used in the 2D LiDAR SLAM [24], the extremely high computational complexity of the 6-DOF parameters searching [25] [26] makes it not possible in the real-time application of the 3D LiDAR SLAM. SegMatch [27] proposed to utilize 3D segments that provide good compromise between local and global descriptions to complete the tasks of detecting and closing loops. But the availability of enough static objects in the environments limits its application.

In this paper, we propose a comprehensive inertial-aided 3D LiDAR SLAM with hybrid geometric primitives that can work in large-scale environments. To make full use of the structural information of the surroundings, plane features and point features are tracked across a fix-sized group of LiDAR keyframes in the local map. An efficient algorithm is developed to simultaneously extract explicit hybrid features from the raw point cloud. Along with the IMU preintegration factors [7] between the sequential keyframes, a sliding-window based tightly-coupled LiDAR Inertial Odometry (LIO) is implemented. For the loop closure detection, the learning-based point cloud place recognition network LPD-Net proposed in our previous work [28] is utilized to generate global descriptors. A sequence-based matching algorithm [29] is applied to accurately find loop closure sequences. A novel and efficient plane-driven sub-maps matching algorithm is proposed to estimate the relative transformation between the sub-maps assembled by the loop closure sequences. Finally, the pose graph constructed by the loop closure factors and the sequential keyframes factors is applied to optimize the whole trajectory. The main contributions of this work are summarized as follows:

- We develop an efficient algorithm to simultaneously extract explicit planes and points to be tracked in the tightly-coupled LiDAR Inertial Odometry.
- We integrate the learning-based point cloud network and sequence matching method to detect loops.
- A novel and efficient plane-driven sub-maps matching algorithm is proposed to close the loops.
- The proposed SLAM system is validated with experiments on different types of environments.

II. OVERVIEW

The pipeline of the proposed inertial-aided 3D LiDAR SLAM system is shown in Fig. 1. The measurements provided by the LiDAR and IMU are firstly processed in the *Measurements Processing* module (Sec. III), in which several types of structural features (ground features, edge features, and plane features) are extracted from the raw point cloud, and IMU measurements are integrated for high-rate state propagation or being assembled as pre-integration factors. In order to reduce computation cost, we don't aim to find all plane patches in the local region, only apparent planes with enough area are extracted for following processes. In the *Local Mapping* module, the structural features of the keyframes are associated, respectively, which can be formulated as different LiDAR factors. The non-linear optimization-based LIO tightly fuses the LiDAR factors and the IMU pre-integration factors in a sliding window manner.

In the *Global Mapping* module (Sec. IV), a certain size of point cloud is treated as the input of the LPD-Net for generating a global descriptor. A sequence-based loop closure detection method is applied to find loops in the keyframe database. Once the similarity of the two keyframe sequences has reached a certain condition, the proposed sub-maps matching algorithm can calculate the relative transformation without initial guess in a near real-time manner. Finally, with the help of the information from the loop closure detection, a 4-DOF global pose graph optimization is applied to eliminate the drift errors introduced by the LIO.

III. MEASUREMENTS PROCESSING AND LOCAL MAPPING

A. Tightly Coupled LIO

The sliding window based factor graph of the proposed LIO is shown in Fig. 2. The variable to be optimized is
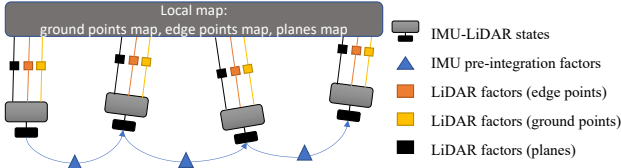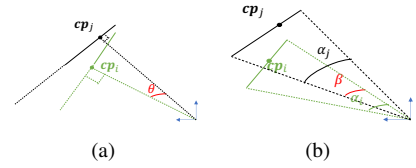
Fig. 2. The factor graph of the tightly-coupled LIO.

Fig. 3. The association methods for plane features. (a) the angle θ between planes $\mathbf{cp}_i$ and $\mathbf{cp}_j$. (b) the angular overlapping angle β.

the several IMU states $\mathbf{x}_i, i \in [0, K]$. K is the size of the window, which should be constant during running. The full state vector to be estimated is defined as

$$
\begin{aligned}
\mathbf{X} &= [\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_K] \\
\mathbf{x}_i &= [^W\mathbf{p}_i, {}^W_i\mathbf{v}, {}^W_i\mathbf{q}, \mathbf{b}_{a_i}, \mathbf{b}_{g_i}]
\end{aligned} \quad (1)
$$

where $(^W\mathbf{p}_i, {}^W_i\mathbf{v}, {}^W_i\mathbf{q})$ represent the position, velocity, rotation (quaternion) of the IMU frame $\{I\}$ at time i relative to the world coordinate frame $\{W\}$. $(\mathbf{b}_{a_i}, \mathbf{b}_{g_i})$ are the bias of the accelerator and the gyroscope represented in the local IMU frame.

Factor graph is used to estimate the full state $\mathbf{X}$, whose objective is minimizing the sum of the Mahalanobis norm of all measurement residuals to obtain the maximum posterior estimation results. In the proposed tightly-coupled LIO, five types of factors will be introduced (presented in Sec. III-B and Sec. III-D): marginalization factors, IMU pre-integration factors, LiDAR edge factors, LiDAR ground factors, and LiDAR plane factors.

B. IMU Pre-integration

IMU measurements includes 3-axis acceleration $\tilde{\mathbf{a}}$ and 3-axis angular velocity $\tilde{\omega}$. In the factor graph, IMU measurements are usually summarized as pre-integration terms [30]. As proposed in [31] and [7], the continuous-time quaternion-based derivation of the IMU pre-integration factor is applied in the system. The group of IMU measurements will be summarized as pre-integration term ${}^i_{i+1}\tilde{\mathbf{z}} = \left({}^i_{i+1}\tilde{\alpha}, {}^i_{i+1}\tilde{\beta}, {}^i_{i+1}\tilde{\gamma} \right)$, which will be used to construct preintegration factors $\mathbf{r}_I$ in the factor graph. A more detailed description of the preintegration factors can be found in [7] and [31].

C. Geometric Primitives Extraction

Raw point cloud is firstly corrected by IMU measurements [8]. Then, three types of structural features are extracted from the undistorted point cloud: ground points $\mathbf{P}_g$, edge points $\mathbf{P}_e$ and plane patches $\mathbf{P}_{cp}$.

1) Structural Feature Extraction: Following the feature extraction procedure proposed in the LeGO-LOAM [3], the undistorted points are firstly projected into a range image. The ground points are firstly extracted using the column-wise evaluation of the range image [32]. The remaining non-ground points are segmented into clusters using the image-based segmentation method [33]. The edge/planar features are extracted from the clusters with enough points by evaluating the roughness. Instead of using the planar features, we further explore their distribution to extract plane patches.

2) Plane Extraction, Modeling and Fusion: Firstly, each point is labelled by feature type: planar feature $+1$, edge feature -1, ground points 0, which are record in the feature type sign matrix $S_{16 \times 1800}$. The sign matrix is filtered by

mean filter with size (H, W). The indices of the explicit plane patches are determined by the filter sign value $s_{ij} > s_{th}$. All the planar features in the filter window are treated as plane points in the patch. The overlapping patches are fused. For each cluster of plane points, the range values are incrementally checked for removing outliers. A two-step plane fitting method is used to evaluate the plane model and the related covariance matrix. In order to avoid over-parameterization, the closest point model [16] $\mathbf{cp} = \mathbf{n} \cdot d$ is used as our final plane model. The Principal Component Analysis (PCA) is firstly applied to estimate the initial plane normal $\mathbf{n}^T$ and bias d. Then, the $\mathbf{cp} = \mathbf{n}d$ is used as initial guess of the closest point model based fitting problem. And the 3×3 covariance matrix can be derived by the Jacobian.

D. Structural Feature Association

Because of the sparse structure of the point cloud, we establish feature association in the local map constructed by the nearby historical keyframes. To be consistent with the extracted features, we divide the local map into three parts: ground points map $^W\mathbf{P}'_g$, edge points map $^W\mathbf{P}'_e$ and planes map $^W\mathbf{P}'_{cp}$. The current estimate ${}^W_i\hat{\mathbf{T}}$ is applied to associate the features with the local map. For each edge feature $^W\mathbf{p}_e \in {}^W\mathbf{P}_e$, we find several nearest points in the local map to fit a 3D line with center $\mathbf{u}$ and covariance matrix $\mathbf{\Sigma}$. The eigenvalues of $\mathbf{\Sigma}$ are used to evaluate the validation of the line [21]. The corresponding pair is defined as $\{\mathbf{p}_i - (\mathbf{p}_j, \mathbf{p}_l)\}$, $\mathbf{p}_j$ and $\mathbf{p}_l$ are sampled from the line near the center. The factor is defined by point-to-line residual:

$$
r_{pl} = \|(\mathbf{p}_i - \mathbf{p}_j) \times (\mathbf{p}_i - \mathbf{p}_l)\| / \|\mathbf{p}_j - \mathbf{p}_l\|. \quad (2)
$$

Similarly, the ground features are also associated with the ground points map. The corresponding pair for ground features is defined as $\{\mathbf{p}_i - (\mathbf{p}_j, \mathbf{p}_l, \mathbf{p}_m)\}$. The factor is defined by point-to-plane residual:

$$
r_{pp} = (\mathbf{p}_i - \mathbf{p}_j) \cdot ((\mathbf{p}_j - \mathbf{p}_l) \times (\mathbf{p}_j - \mathbf{p}_m)). \quad (3)
$$

As shown in Fig. 3, for the plane features, the association of planes is divided into two steps: the plane candidates are found using the angle θ between two closest points; the angular overlapping rate between candidates is evaluated by $r = max\left(\frac{\beta}{\alpha_i}, \frac{\beta}{\alpha_j} \right) > r_{th}$. The corresponding pair for planes are defined as $\{\mathbf{cp}_i - \mathbf{cp}_j\}$. The factor is defined by plane-to-plane residual:

$$
\mathbf{r}_{cp} = \mathbf{cp}_i - \mathbf{cp}_j. \quad (4)
$$

IV. GLOBAL MAPPING

A. Loop Closure Detection

In order to trigger global optimization, a distinct global descriptor is needed for each keyframe in the database. The state-of-the-art LPD-Net [28] is used to generate the
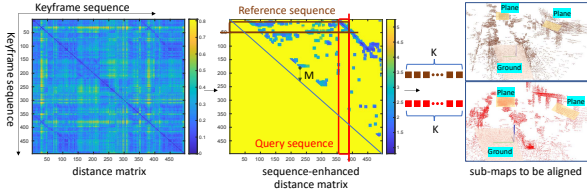
Fig. 4. Sequence-based loop closure detection.

global descriptor for each keyframe. As the size of input of the neural network is constant (4096×3), we prioritize more informative planar points and sharp points extracted in the measurement processing step. If points are not enough, we will get them randomly from the outlier points. Thus, the output 256-dimensional vectors of the network can be associated to the keyframes as the global descriptors. L_2 distance of two vectors is applied to evaluate the similarity of these two keyframes. The smaller value of the L_2 distance between two keyframes' descriptors, the more likely the two keyframes are from the same place. In Fig. 4, the leftmost figure shows the distance matrix in a typical scenario.

Instead of detecting loops by finding the best single match with the smallest distance between descriptors, a more robust and accurate sequence-based matching approach [29] is employed. For each newly added keyframe, we use it and the $(K-1)$ keyframes before it as the query index sequence. The best match reference sequence is found in the index region $[M, N - M]$. K is the sequence length, N is the size of the whole keyframe database, and M is the index threshold preventing useless calculations with the sequences that are too close. Several candidate trajectories are generated at each allowable index with different possible velocities ranging from V_{min} to V_{max}. The difference score S of each candidate trajectory is calculated as the mean of the distance values along the trajectory on the distance matrix passing through with different velocities. We store the trajectory with the smallest S for each index, as shown in the middle of Fig. 4. To find the best sequence match from an array of candidates, the windowed uniqueness score is applied [34]. Finally, as shown in the rightmost figure in Fig. 4, we get two loop-closed sequences of keyframes which can be used to construct two sub-maps with different structural features.

B. Efficient Plane-driven Sub-maps Matching

To register the point clouds of the two sub-maps, the ICP-based algorithms suffer from local minima. The RANSAC-based methods not only fail to guarantee the global optimality, but also the run-time increases exponentially with the outlier rate, which is usually very high in the 3D keypoint matching methods due to the nature of the 3D point cloud: irregular density and lack of useful texture. The BnB-based parameters searching algorithms can obtain globally optimal solution, but the high computational cost makes them impractical in 6-DOF problem.

In this section, we introduce the plane-driven sub-maps registration algorithm to solve the sub-maps matching problem. We iteratively sample two pairs of planes to estimate five degrees of freedom of the relative transformation, and then use point-point correspondences to estimate the remaining degree of freedom (1-DOF translation along the direction

of the intersection line of the two planes). The insight of the proposed algorithm is that the number of planes in 3D data is usually smaller than the number of points, and three pairs of non-parallel planes which can constraint all degrees of freedom in 3D registration are not easy to be captured in the local region of the environments [35]. However, it is much easier to find a pair of matching planes besides of the matched ground planes. The pipeline of the proposed plane-driven sub-maps registration algorithm is shown in Fig. 5.

Algorithm 1 Efficient Plane-driven Sub-maps Matching

Input: $\left(\pi_i^g, \pi_j^g\right)$, $(\mathbf{\Pi}_i, \mathbf{\Pi}_j)$, $\mathbf{M} \doteq \{\left(\mathbf{p}_i^k, \mathbf{p}_j^k\right)\}$ and ξ.

Output: Relative transformation $\mathbf{T}_i^j$ between two sub-maps which enables the largest consensus set in point cloud registration.

1: $\mathfrak{T}^* \leftarrow \emptyset$, $_j^i\mathbf{T}^* = \mathbf{I}$.
2: **for** $\left(\pi_i^l, \pi_j^l\right) \in (\mathbf{\Pi}_i, \mathbf{\Pi}_j)$ **do**
3:     Compute the transformation $\mathbf{R}_i^g$, $\mathbf{R}_j^g$, $\mathbf{t}_i^g$, $\mathbf{t}_j^g$, introduced by the ground plane-pair $\left(\pi_i^g, \pi_j^g\right)$ and the sampled plane-pair $\left(\pi_i^l, \pi_j^l\right)$ (Eqn. (5)(6)(7)).
4:     Transform point clouds respectively by $_i^{s_i}\mathbf{T}$ and $_j^{s_j}\mathbf{T}$.
5:     $\mathfrak{T} \leftarrow \emptyset$.
6:     **for** $\left(\mathbf{p}_i^k, \mathbf{p}_j^k\right) \in \mathbf{M}$ **do**
7:         Calculate the translation interval $\mathfrak{T}^k$.
8:         **if** $\mathfrak{T}^k$ valid **then**
9:             $\mathfrak{T} = \mathfrak{T} \cup \mathfrak{T}^k$.
10:         **end if**
11:     **end for**
12:     **if** $\mathfrak{T} \neq \emptyset$ **then**
13:         Search t_x that stabs the most intervals: $\mathfrak{T}^l$.
14:         **if** $|\mathfrak{T}^l| > |\mathfrak{T}^*|$ **then**
15:             Calculate $_j^i T$ using Eqn. (10).
16:             $\mathfrak{T}^* \leftarrow \mathfrak{T}^l$.
17:         **end if**
18:     **end if**
19: **end for**
20: **return** $_j^i\mathbf{T}^*$, $\mathfrak{T}^*$.

---

1) Sampling Planes: Considering the fact that the number of planes in real 3D data is much smaller than the number of points, in addition to the matched ground planes $\left(\pi_i^g, \pi_j^g\right)$, one more pair of planes $\left(\pi_i^l, \pi_j^l\right)$ will be sampled from all possible combinations in the plane sets $(\mathbf{\Pi}_i, \mathbf{\Pi}_j)$ as inlier plane match. Two pairs of plane-to-plane matches can fix five DOFs defined in the selected frames [36] such that

- π_i^g and π_j^g lie respectively in xy-planes of the frames.
- the x-axes of the two frames are along the intersection line $\mathbf{l}_i (\mathbf{l}_j)$ of the planes $\pi_i^g \cap \pi_i^l (\pi_j^g \cap \pi_j^l)$.

As shown in the middle of Fig. 5, given the ground plane $\pi^g \doteq (\mathbf{n}^g, d^g)$ and the sampled plane $\pi^l \doteq (\mathbf{n}^l, d^l)$, the intersecting 3D line is represented in *Plücker coordinates* as

$$\mathbf{l} = [\mathbf{l}_d, \mathbf{l}_m] = \left[\left(\mathbf{n}^g \times \mathbf{n}^l\right)^T \quad \left(d^g\mathbf{n}^l - d^l\mathbf{n}^g\right)^T \right]^T \quad (5)$$

where $\mathbf{l}_d$ and $\mathbf{l}_m$ are the direction and the moment of the 3D line. The point of line $\mathbf{l}$ closest to the origin is defined as

$$\mathbf{c} = (\mathbf{l}_d \times \mathbf{l}_m) / (\mathbf{l}_d \cdot \mathbf{l}_d). \quad (6)$$
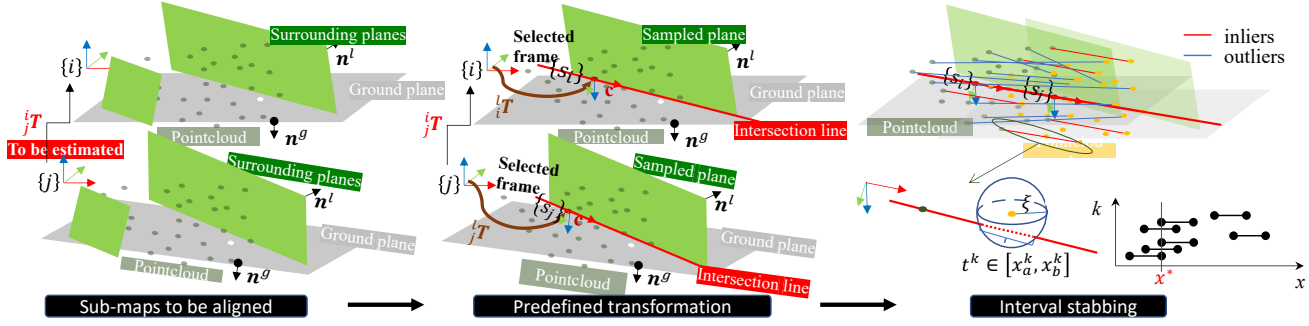
Fig. 5. The pipeline of the plane-driven sub-maps matching algorithm. The sub-maps to be aligned contain hybrid geometric primitives: ground plane, surrounding planes and point cloud. In addition to the ground plane, one more plane is sampled to determine the selected frame for each sub-map, with the predefined transformation being calculated using the intersecting planes. Applying the predefined transformations on the point clouds, the only uncertain degree of freedom is the translation along the x direction. For each pair of the corresponding points, only if the straight line parallel to the x-axis and passing through point p_i^k intersects the ξ-ball centered at p_j^k, can the pair $\left(p_i^k, p_j^k\right)$ have chance to be inlier. The condition for the intersection of line and ball defines an interval of x. For the group of intervals, one of the endpoints that stabs the most intervals is the optimal x^*.

The selected frame $\{s\}$ is based on the closest point $\mathbf{c}$ as the origin. A rigid transformation $_i^s\mathbf{T} \doteq (_i^s\mathbf{R}, _i^s\mathbf{t})$ is needed to transform the point cloud from the original frame to the selected frame. In order to agree with the above two conditions, the z-axis of frame $\{s\}$ is defined to be consistent with the ground plane normal $\mathbf{n}^g$; where the x-axis is defined to be consistent with the direction of the 3D line $\mathbf{l}$; the y-axis is defined according to the right-hand coordinate system:

$$_i^s\mathbf{R} = \left[\frac{\mathbf{l}_d}{\|\mathbf{l}_d\|}, \frac{\frac{\mathbf{l}_d}{\|\mathbf{l}_d\|} \times \frac{\mathbf{n}^g}{\|\mathbf{n}^g\|}}{\left\| \frac{\mathbf{l}_d}{\|\mathbf{l}_d\|} \times \frac{\mathbf{n}^g}{\|\mathbf{n}^g\|} \right\|}, \frac{\mathbf{n}^g}{\|\mathbf{n}^g\|} \right]^T \tag{7}$$

$$_i^s\mathbf{t} = -_i^s\mathbf{R}\mathbf{c}$$

where $\|\cdot\|$ is the Euclidean norm. Thus, the transformations $_i^{s_i}\mathbf{T}$, $_i^{s_j}\mathbf{T}$ predefined by the two pairs of plane-to-plane matches can be computed. After applying the predefined transformations $_i^{s_i}\mathbf{T}$ and $_i^{s_j}\mathbf{T}$ to the respective pointclouds, the only unknown parameter is translation along x-axis: t_x.

2) Searching t_x by Interval Stabbing: As shown in rightmost part of the Fig. 5, t_x is the only unknown parameter that need to be solved in the point cloud registration, which can be formulated as a consensus maximization problem

$$\arg\max_{t_x} \quad |C|$$
$$s.t. \quad \left\| \mathbf{p}_i^k + \begin{bmatrix} t_x \\ 0 \\ 0 \end{bmatrix} - \mathbf{p}_j^k \right\| \le \xi, \left(\mathbf{p}_i^k, \mathbf{p}_j^k\right) \in \mathbf{M}, k \in C, \tag{8}$$

where C is the consensus set of point-point correspondences, $|\cdot|$ is the size of the consensus set, ξ is the consensus threshold, $\mathbf{M}$ is the set of all point-point correspondences which are calculated using FPFH features [37].

The Eq. (8) aims to find the translation t_x that aligns as many point-point pairs $\left(\mathbf{p}_i^k, \mathbf{p}_j^k\right)$ as possible up to ξ. As shown in the rightmost Fig. 5, for each point $\mathbf{p}_i^k$, translating it with $[t_x, 0, 0]^T$ forms a straight line $\mathbf{l}_{p_i}$ along the x-axis in 3D space. As for the matching uncertainty ξ, we define a ξ-ball centered at $\mathbf{p}_j^k$ as $\mathbf{B}_{p_j} \doteq \{\tilde{\mathbf{p}}_j | \|\tilde{\mathbf{p}}_j - \mathbf{p}_j\| \le \xi\}$. We are aware that the pair $\left(\mathbf{p}_i^k, \mathbf{p}_j^k\right)$ can be aligned only if the straight line $\mathbf{l}_{p_i}$ intersects with the ball $\mathbf{B}_{p_j}$. Otherwise, they cannot be aligned by only adjusting t_x under the condition of sampled plane-pairs being aligned. Thus, for each inter-

secting pair k, we have a translation interval $\mathfrak{T}^k \doteq \{t_{x_i}^k | t_{x_i}^k \in [x_a^k, x_b^k]\}$ calculated by 3D line-sphere intersection.

For Eq. (8), we use the line-sphere intersection to convert the point matches into a set of translation intervals $\mathfrak{T} \doteq \{\mathfrak{T}^k\}, k \in \mathbf{M}_1$. $\mathbf{M}_1$ is the index set of intersecting point pairs. Our objective is to search a t_x that agrees with the largest number of the point matches, which can be formulated as the interval stabbing problem [38]

$$\arg\max_{t_x} \sum_{k \in \mathbf{M}_1} \mathbf{\Gamma}\left(t_x \in \mathfrak{T}^k\right) \tag{9}$$

where $\mathbf{\Gamma}(\cdot)$ is the indicator function that returns 1 if the condition is true. The Eq. (9) can be efficiently solved in $\mathcal{O}\left(|\mathbf{M}_1| \log |\mathbf{M}_1|\right)$ time [38] which is suitable in practice.

3) Summary of the Plane-driven Sub-maps Matching: Alg. 1 summarizes the methodology of the plane-driven sub-maps matching. Given the ground plane pair $\left(\pi_i^g, \pi_j^g\right)$, environmental plane set $\left(\mathbf{\Pi}_i, \mathbf{\Pi}_j\right)$, point correspondences $\mathbf{M} \doteq \{\left(\mathbf{p}_i^k, \mathbf{p}_j^k\right)\}$ and inlier threshold for point matches ξ, our method iteratively samples an environmental plane pair $\left(\pi_i^l, \pi_j^l\right)$, with the associated ground plane pair, to estimate 5 DOFs represented in the selected frame; Then, the remaining unknown translation t_x is efficiently searched by interval stabbing. The relative transformation is calculated as

$$_j^i\mathbf{T} = (_i^{s_i}\mathbf{T})^{-1} \, _{s_j}^{s_i}\mathbf{T}(t_x) _j^{s_j}\mathbf{T} \tag{10}$$

where $_{s_j}^{s_i}\mathbf{T}(t_x) \doteq \left(\mathbf{I}, (t_x, 0, 0)^T\right)$. The output includes $_j^i\mathbf{T}^*$ that maximizes the size of the consensus set $\mathfrak{T}^*$.

C. Global Optimization

Similar with the state-of-the-art VINS system [7], the roll and pitch angles are fully observable in our system owing to the inertial measurement of the gravity. We also apply the 4-DOF pose graph optimization to correct the drift errors in the yaw angle ϕ and 3D translation $\mathbf{t}$. Two types of edges are used in the pose graph, one is the sequential edges from the LIO; another is the loop-closure edges formed by the plane-driven sub-maps matching algorithm.

V. EXPERIMENTAL RESULTS

In this section, several tests are conducted to qualitatively and quantitatively evaluate the performance of the proposed SLAM system. Firstly, we show the results with the large-scale data collected by the sensors installed on a passenger

TABLE I

THE MATCHING RESULTS UNDER DIFFERENT TRANSFORMATIONS.

Ground truth	PDSM			PDSM + ICP			RANSAC		
(yaw, pitch, raw, **t**)	angle error (degree)	position error (m)	time (ms)	angle error (degree)	position error (m)	time (ms)	angle error (degree)	position error (m)	time (ms)
(60, -40, 80, 50, 20,-10)	0.027	0.2992	**73**	1.03E-05	6.15E-05	**302**	0.71	0.76	**225969**
(-60, 50, -20,-10, 30,-20)	0.045	0.2986	**80**	2.62E-05	1.26E-05	**288**	0.3	0.16	**199332**
(-6, 5, -2, -1, 3,-2)	1.78E-05	0.3	**92**	1.25E-05	3.05E-05	**303**	6.45E-06	3.31E-06	**193315**
(40, 10, 20, -10, 4, 2)	3.14E-05	0.3	**72**	2.34E-05	1.00E-05	**307**	0.13	0.053	**218273**
(-90,0,0,-20,10,-2)	9.29E-05	0.3	**98**	6.28E-06	2.96E-05	**289**	1.08E-05	7.26E-06	**185817**

car in the CUHK campus in which the roads are undulating and curved. Secondly, we demonstrate the competitive results in the relatively small scale place with only smooth motion. Lastly, we emphatically show the results of the proposed plane-driven sub-maps matching algorithm compared with other registration methods. Velodyne VLP-16 LiDAR (10Hz) and Xsens MTi-300 IMU (100Hz) are used in our tests.
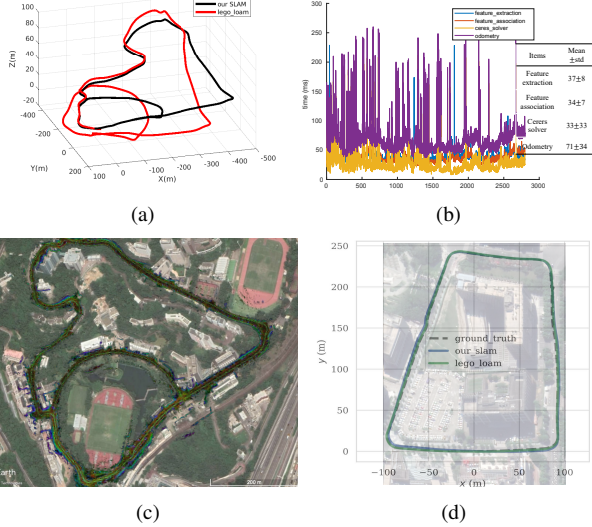

(a)

(b)

(c)

(d)

Fig. 6. Results of the proposed SLAM and LeGO-loam. (a) the comparison of the trajectories. (b) the running time analysis of the proposed LIO. (c) the cloud map generated by the proposed SLAM aligned with Google Earth.(d) the SLAM trajectories and the ground-truth trajectory.

A. Evaluation of SLAM System

1) Qualitative Analysis: In this test, we focus on illustrating the robustness and effectiveness of the proposed SLAM system in large-scale places, shown in Fig. 6(c). The testing data set is collected in challenging environments where the undulating, sharply curved roads and occasionally structural-less surroundings make the front-end odometry accumulate dramatic drift errors. The elevation change of the driven trajectory is up to 80 meters. The length of the first-passed big loop is about $2km$. In such a case, the pose-proximity strategy fails to detect loops effectively, see the trajectory generated by LeGO-loam [3] in Fig. 6(a). The LIO-SAM system proposed in [8] integrates a similar loop closure module. It failed to perform initialization due to the lack of enough IMU excitation in this environment. However, the proposed SLAM system succeeds in detecting loops and accurately estimating the relative transformation between the loop-closed sequences, shown in Fig. 6(a).

2) Quantitative Analysis: In order to investigate the global consistency of the SLAM trajectories, we compare the root mean square error with the representative LeGO-loam [3]. A small-scale environment is chosen to enable the pose-proximity strategy in LeGO-loam. As shown in Fig. 6(d),

both the SLAM trajectories are well aligned with ground-truth trajectory. The translation root mean square error results are: $0.94m$ (LeGO-loam), $1.02m$ (our SLAM). The rotation RMSE results are: $2.3°$ (LeGO-loam), $2.6°$ (our SLAM).

The statistics of the running time is shown in Fig. 6(b). The size of the sliding window is set as 7. The mean time of feature extraction is about $37ms$, including extracting the edge points, ground point and planes. Associating these features costs about $34ms$. The mean time of the LIO is about $71ms$. The peaks of the time arrays are mainly caused by the characters (noisy, structure level) of LiDAR measurements from the local region.

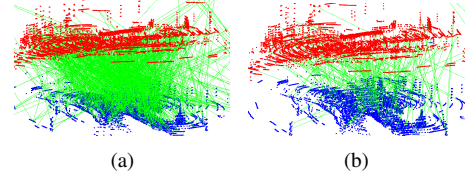### B. Evaluation of Plane-driven Sub-maps Matching

(a)

(b)

Fig. 7. Results of sub-maps matching $(-90, 0, 0, 0, 0, -20)$. (a) the initial correspondences. (b) the final inlier set computed by our algorithm.

In order to demonstrate the efficiency and precision of the proposed plane-driven sub-maps matching algorithm, in this test, we choose a typical sub-map (including 8 explicit plane patches and 10923 feature points) generated by our LIO. Firstly, we extract ISS keypoints [39] and use the FPFH descriptors [37] to establish initial point-to-point correspondences (costing 752ms), shown in Fig. 7(a). We apply different known transformations on the sub-map to generate 3D data with ground-truth transformation. As shown in Table I, under these salient transformations in all DOFs, the proposed registration algorithm converges within 100 milliseconds. The angle error is extremely small, and the position accuracy is up to the point matching uncertainty $\xi = 0.3m$. With the help from the regular ICP, we can get optimal results. From these results, the proposed matching algorithm is deterministic and very efficient, compared with the RANSAC-based algorithm implemented in the PCL [19].

VI. CONCLUSION

In this paper, we have presented a comprehensive inertial-aided 3D SLAM system, which leverages the hybrid geometric primitives in large-scale environments to perform real-time tightly-coupled LIO and efficient back-end sub-maps matching. The combination of LPD-Net and sequence-based matching algorithm made the point cloud based loop-closure detection more robust and accurate. What's more, the proposed plane-driven sub-maps matching algorithm is deterministic and achieves near real-time computation. In the future, we will evaluate our proposed SLAM system on benchmark datasets. And we will explore more on the optimality proof of the PDSM algorithm.

REFERENCES

[1] F. Zheng, "Proprioception-aided visual state estimation for mobile robots," Ph.D. dissertation, The Chinese University of Hong Kong (Hong Kong), 2019.

[2] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Proc. of Robotics:Science and Systems*, 2014, pp. 9–17.

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

[4] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1104–1119, 2012.

[5] C. Park, P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan, "Elastic lidar fusion: Dense map-centric continuous-time slam," in *Proc. of IEEE International Conference on Robotics and Automation*. IEEE, 2018, pp. 1206–1213.

[6] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2016.

[7] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

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

[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. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2020.

[10] Y. Yang, P. Geneva, X. Zuo, K. Eckenhoff, Y. Liu, and G. Huang, "Tightly-coupled aided inertial navigation with point and plane features," in *Proc. of International Conference on Robotics and Automation*, 2019, pp. 6094–6100.

[11] Y. Yang and G. Huang, "Observability analysis of aided ins with heterogeneous features of points, lines, and planes," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1399–1418, 2019.

[12] J. W. Weingarten, G. Gruener, and R. Siegwart, "Probabilistic plane fitting in 3d and an application to robotic mapping," in *Proc. of IEEE International Conference on Robotics and Automation*, vol. 1. IEEE, 2004, pp. 927–932.

[13] M. Kaess, "Simultaneous localization and mapping with infinite planes," in *Proc. of IEEE International Conference on Robotics and Automation*. IEEE, 2015, pp. 4605–4611.

[14] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, "Keyframe-based dense planar slam," in *Proc. of IEEE International Conference on Robotics and Automation*. IEEE, 2017, pp. 5110–5117.

[15] M. Hsiao, E. Westman, and M. Kaess, "Dense planar-inertial slam with structural constraints," in *Proc. of IEEE International Conference on Robotics and Automation*. IEEE, 2018, pp. 6521–6528.

[16] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, "Lips: Lidar-inertial 3d plane slam," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 123–130.

[17] W. S. Grant, R. C. Voorhies, and L. Itti, "Finding planes in lidar point clouds for real-time registration," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 4347–4354.

[18] J. Poppinga, N. Vaskevicius, A. Birk, and K. Pathak, "Fast plane detection and polygonalization in noisy 3d range images," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2008, pp. 3378–3383.

[19] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *Proc. of IEEE International Conference on Robotics and Automation*, Shanghai, China, May 9-13 2011.

[20] S.-Y. An, L.-K. Lee, and S.-Y. Oh, "Fast incremental 3d plane extraction from a collection of 2d line segments for 3d mapping," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 4530–4537.

[21] J. Serafin, E. Olson, and G. Grisetti, "Fast and robust 3d feature extraction from sparse point clouds," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2016, pp. 4105–4112.

[22] C. L. Gentil, T. A. Vidal-Calleja, and S. Huang, "IN2LAAMA: inertial lidar localisation autocalibration and mapping," *CoRR*, vol. abs/1905.09517, 2019. [Online]. Available: http://arxiv.org/abs/1905.09517

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

[24] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *Proc. of IEEE International Conference on Robotics and Automation*, 2016, pp. 1271–1278.

[25] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-icp: A globally optimal solution to 3d icp point-set registration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2241–2254, 2016.

[26] D. Campbell and L. Petersson, "Gogma: Globally-optimal gaussian mixture alignment," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5685–5694.

[27] R. Dub, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "Segmatch: Segment based place recognition in 3d point clouds," in *Proc. of IEEE International Conference on Robotics and Automation*, 2017, pp. 5266–5272.

[28] Z. Liu, S. Zhou, C. Suo, P. Yin, W. Chen, H. Wang, and Y.-H. Liu, "LPD-Net: 3D point cloud learning for large-scale place recognition and environment analysis," in *Proc. of the IEEE International Conference on Computer Vision*, 2019, pp. 2831–2840.

[29] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *Proc. of IEEE International Conference on Robotics and Automation*, 2012, pp. 1643–1649.

[30] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation." Georgia Institute of Technology, 2015.

[31] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 61–76, 2012.

[32] M. Himmelsbach, F. V. Hundelshausen, and H.-J. Wuensche, "Fast segmentation of 3d point clouds for ground vehicles," in *Proc. of IEEE Intelligent Vehicles Symposium*. IEEE, 2010, pp. 560–565.

[33] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3d laser scans for online operation," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2016, pp. 163–169.

[34] B. Talbot, S. Garg, and M. Milford, "Openseqslam2.0: An open source toolbox for visual place recognition under changing conditions," in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7758–7765.

[35] Y. Taguchi, Y. Jian, S. Ramalingam, and C. Feng, "Point-plane slam for hand-held 3d sensors," in *Proc. of IEEE International Conference on Robotics and Automation*, 2013, pp. 5182–5189.

[36] A. Mateus, S. Ramalingam, and P. Miraldo, "Minimal solvers for 3d scan alignment with pairs of intersecting lines," in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7234–7244.

[37] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Proc. of IEEE International Conference on Robotics and Automation*, 2009, pp. 3212–3217.

[38] M. de Berg, M. van Kreveld, M. Overmars, and O. C. Schwarzkopf, *More Geometric Data Structures*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 211–233.

[39] Y. Zhong, "Intrinsic shape signatures: A shape descriptor for 3d object recognition," in *Proc. of IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, 2009, pp. 689–696.