2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Real-Time Simultaneous Localization and Mapping with LiDAR Intensity

Wenqiang Du and Giovanni Beltrame

Abstract— We propose a novel real-time LiDAR intensity image-based simultaneous localization and mapping method, which addresses the geometry degeneracy problem in unstructured environments. Traditional LiDAR-based front-end odometry mostly relies on geometric features such as points, lines and planes. A lack of these features in the environment can lead to the failure of the entire odometry system. To avoid this problem, we extract feature points from the LiDAR-generated point cloud that match features identified in LiDAR intensity images. We then use the extracted feature points to perform scan registration and estimate the robot ego-movement. For the back-end, we jointly optimize the distance between the corresponding feature points, and the point to plane distance for planes identified in the map. In addition, we use the features extracted from intensity images to detect loop closure candidates from previous scans and perform pose graph optimization. Our experiments show that our method can run in real time with high accuracy and works well with illumination changes, low-texture, and unstructured environments.

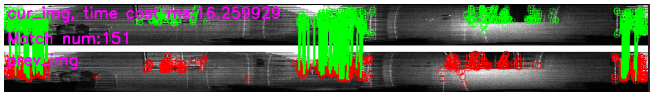
I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) is a fundamental problem in robotics. SLAM is the process of building a map of the environment and in the meanwhile, tracking the robot's pose on the generated map. There are many kinds of SLAM methods, such as visual SLAM [1], Light Detection and Ranging (LiDAR) SLAM [2], visual and LiDAR SLAM [3], visual-intertial and ranging SLAM (VIR-SLAM) [4] and so on. In this paper, we focus on the LiDAR SLAM.

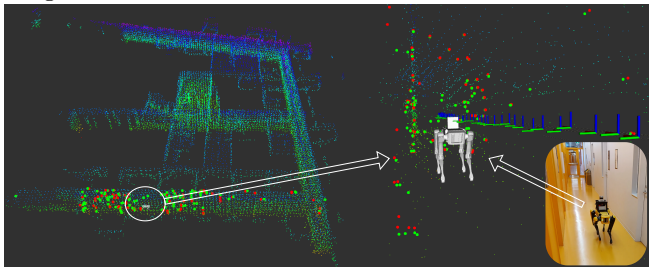
Numerous SLAM methods based on LiDAR [3], [5]–[7] have been proposed over the past few years. Most of them are based on the geometric features (e.g., edges and planes [2], [7]) of LiDAR point clouds. These methods are robust and accurate in most structured environment. However, when the environment has little structure, these methods can suffer from geometric degeneracy and fail [8], [9]. For example, in a long corridor or a cave environment, there might be enough plane features, but too few edge features to estimate the relative ego-motion movement between two consecutive scans. Unfortunately, aligned plane features (like in the corridor) alone are not sufficient to estimate the robot pose in six degrees of freedom (6DOF), meaning the robot could lose the sense of forward or backward movement.

To solve this problem, one needs an additional reference to constrain the sixth degree of freedom. One possible way is to extract feature points from textural information. Recently, many researchers tried to add intensity information to augment [10] or assist [11] LiDAR SLAM systems, improving their performance. However, these feature points from textural information are already presented in 3D space and theoretically are enough to constrain the 6DOF movement. Why not directly perform scan registration with these feature points? We propose a pure LiDAR intensity SLAM method which directly extracts feature points from intensity images and perform scan registrations to estimate the robot's ego-movement. Our contributions are:

- A novel real-time LiDAR intensity image-based SLAM system, aiming at solving the geometric degeneracy problem;
- Combining the benefits of visual SLAM systems with those of LiDAR SLAM systems, without suffering from blurring or illumination changes;
- A lightweight front-end due to fewer feature points and adding ground plane constraint and LiDAR Bundle Adjustment (BA) to the back-end;
- Intensity-based loop closure detection and pose graph optimization;



(a) Feature points in the current scan (top) and the previous scan (bottom) are matched with ORB features from intensity images.



(b) The matched feature points are used to perform the scan registration.

Fig. 1: The matched 3D points from two consecutive scans and their corresponding feature points. The points in (b) are the 3D points that are extracted from point clouds according to the indexes of matched features from (a). The red points represent matched points of the previous frame, and the green points stand for matched points in the current scan. Those points are then used for scan registration to estimate the relative poses between two consecutive frames.

The authors are with the Department of Computer Engineering and Software Engineering, Polytechnique Montreal, University of Montreal, Montreal, QC, H3T1J4, Canada. {wenqiang.du, giovanni.beltrame}@polymtl.ca

II. RELATED WORK

In the past decades, many researchers worked in SLAM and achieved many great results. In 1990, Smith et al. [12] firstly used extended Kalman filter (EKF) to estimate the relative position among objects in an unknown environment and build a stochastic map to store these estimations of spatial relationships and their uncertainties. Nowadays, there are many kinds of SLAM systems [13], many using an Inertial measurement unit (IMU), cameras, and/or LiDARs.

LiDAR-based SLAM systems have been well studied in both theory and industrial applications in recent years. One representative work is LOAM [2], a real-time method for estimating odometry and mapping. LOAM only consumes a LiDAR's point cloud and then extracts edge and planar points from this point cloud to register consecutive scans at 10Hz and providing map updates at 1Hz. Another popular work is Cartographer [14], which uses a scan-to-submap matching strategy with loop closure detection and graph optimization, achieving high accuracy even in real-time. LeGO-LOAM [7] is also a real-time odometry and mapping SLAM method, which uses only the LiDAR of ground vehicles as a front-end sensor. LeGO-LOAM extracts ground planar and edge features and assigns them different labels, which significantly decreases the number of features. Shang et al. [5] proposed LIO-SAM, which is a tightly-coupled LiDAR-inertial odometry system. LIO-SAM can estimate the odometry by using LiDAR and a 9-axis high-frequency IMU. Shang et al. results showed that LIO-SAM is much more accurate than LOAM and can be run in real-time on a computationally limited platform.

Differentiating from these feature-based odometry methods, Chen et al. [15] proposed a direct method to estimate the odometry using LiDAR and IMU. Their results showed that their method is more accurate than a feature-based approach. DLO [15] is another direct LiDAR odometry algorithm that can match consecutive frames of point cloud directly with high accuracy in real-time for computationally-limited robot platforms, but it is sensitive to dynamic objects. Lin et al. [16] [17] developed a robust and accurate SLAM system using solid-state LiDAR with small field of view (FoV).

In addition to traditional algorithms, some deep learning-based LiDAR SLAM [18]–[20] systems also achieved good performance. PointNet [21] and PointNet++ [22] leverage deep learning techniques to directly extract semantic features from 3D point clouds, enabling their use in segmentation and extraction of semantic information. In addition to feature extraction, deep learning was used in SegMatch [23] for place recognition matching 3D segments. PointNetVLAD [24] is a combination of PointNet and NetVLAD [25], which uses end-to-end learning to extract global features from a frame of point cloud, which are very useful for place recognition.

Recently, as LiDAR resolution is steadily increasing, we can generate much more clear and textured images (see Fig. 1a) from intensity information. In recent years, integrating intensity information into their SLAM system has emerged as a novel approach among researchers [26]–[28]. Wang et al. [11] introduce intensity features into their SLAM system, using both intensity and geometric features to improve the performance of the SLAM system. Li et al. [10] extract intensity edge points in a solid-state LiDAR-inertial SLAM system, while [27], [29] proposed a novel visual place recognition method using LiDAR intensity information.

In this paper, we propose a novel lightweight LiDAR odometry method which directly matches 3D feature points extracted from intensity images. This method can merge both the feature tracking ability of visual SLAM and LiDAR's high accuracy. We also propose an intensity image-based back-end, including additional constraints between intensity features and using intensity information to detect loop closures.

III. METHOD

The pipeline of our proposed method is shown in Fig. 2. In our system, a LiDAR generates a point cloud within 100ms that is called a frame or a scan. To estimate the movement of the robot, we need to calculate the relative pose between consecutive frames. We use intensity odometry to implement this procedure in the front-end. In addition, we use the term "odometry" to describe the relative pose between the current frame and the initial frame. The odometry from the front-end is generally inaccurate, so we need to use a back-end to optimize the odometry. In the back-end, we use the scan-to-map method [2] and LiDAR BA [30] to correct the drift. However, map optimization is generally not enough for removing the accumulated drift, and we add loop closure detection as an additional means to reduce drift. In our case, we perform loop closure detection on the LiDAR intensity image and use pose graph optimization to update the trajectory and generate the final trajectory of the robot.

A. Intensity Odometry

Assume that we have two consecutive frames of point clouds $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_J\}$ and $\mathcal{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_I\}$ from a LiDAR: one direct way of estimating the relative pose is directly applying an Iterative Closest Point (ICP) algorithm [31] to calculate the rotation matrix $\mathbf{R}$ and $\mathbf{T}$:

$$\underset{\mathbf{R},\mathbf{T}}{\arg\min} \sum_{\mathbf{X}_j \in \mathcal{X}, \mathbf{Y}_i \in \mathcal{Y}} \| \mathbf{Y}_i - \mathbf{R}\mathbf{X}_j - \mathbf{T} \|^2 \tag{1}$$

However, this method usually consumes significant time and computation resources [2]. To reduce the computation cost, we need to extract representative points for the scan registration, thereby reducing the number of points used for optimization (up to a limit: we

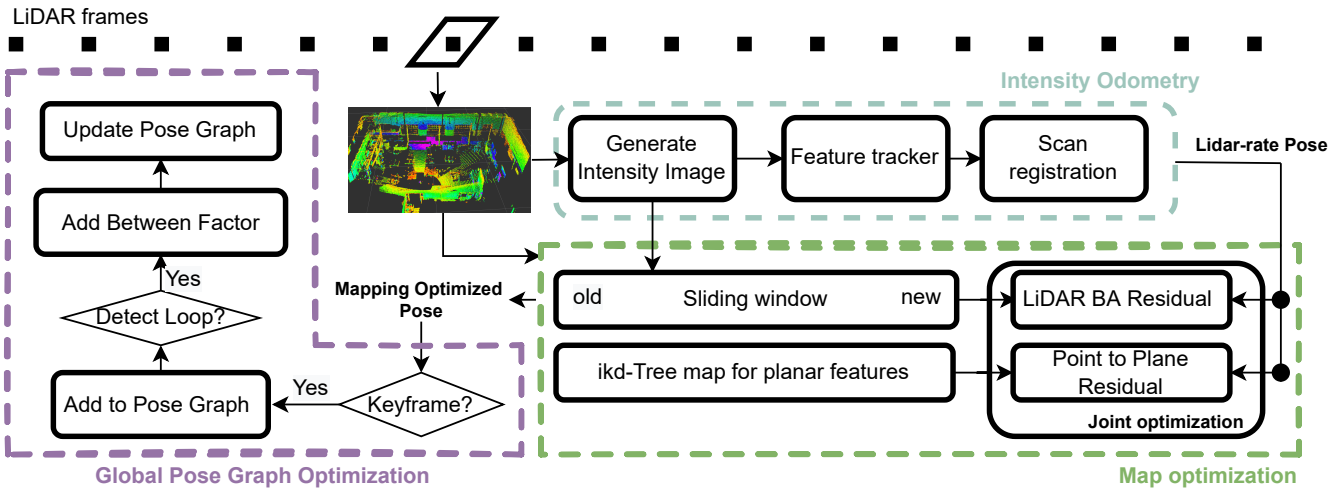


Fig. 2: System overview of the proposed method. The whole system consists of three parts, including intensity odometry, map optimization, and pose graph optimization. The intensity odometry part is the core of the proposed method. It consists of intensity image generation, feature tracking, and scan registration. The map optimization corrects the drift by jointly minimizing both LiDAR BA residual and point to map plane residual. Pose graph optimization corrects the whole trajectory by adding loop constraints.

still need to maintain the original relationships between two frames).

In order to decrease the number of points used for optimization, Zhang et al. [2] tried to extract edge and plane features. The points of the edge feature of the current frame can be matched with the edges in the map. The same goes for plane features.

With edge and plane features, we can optimize the point to line distance, and point to plane distance jointly, and estimate $\mathbf{R}$ and $\mathbf{T}$. However, sometimes we cannot extract edge features accurately enough, like in the long corridor or cave environment. In this case, we will lose the ability to estimate 6DOF movement.

To solve this issue, we extract and track features directly from intensity images. Fig. 1a shows the intensity images generated from an Ouster-64 LiDAR, where the image resolution is 1024×64. Even if the vertical resolution is low, we can still extract enough features (the red and green circles are the ORB features [32]–[34]) for estimating movement. We extract 3D points, $\mathcal{Y}_2 = \{\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_k\}$ directly from the point cloud according to the index of the matched ORB feature points from the intensity image. Each 3D feature point is assigned a score $\mathcal{S} \in \{\mathbf{S}_1, \mathbf{S}_2, \cdots, \mathbf{S}_n\}$ obtained during feature extraction. Similarly, we can extract the corresponding points $\mathcal{X}_2 = \{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_k\}$ in the following scan, and we scan match as a least squared estimation problem:

$$\bar{\mathbf{X}}_n = \mathbf{R}\mathbf{X}_n + \mathbf{T} \tag{2}$$

$$\arg\min_{\mathbf{R},\mathbf{T}} \sum_{n \in \mathcal{N}} \| \left(\mathbf{Y}_n - \bar{\mathbf{X}}_n\right) \cdot \mathbf{S}_n \|^2 \tag{3}$$

where $\mathcal{N} = [1, 2, \cdots, k]$, and $\mathbf{S}_n$ is the score of the nth matched feature point according to Hamming distance.

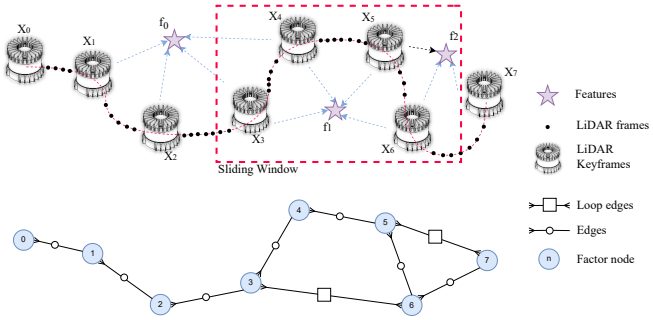


Fig. 3: Illustration of the sliding window strategy used for LiDAR BA (top) and Pose graph (bottom) with loop closure contraints.

This way, we can limit the number of features to around 200 points.

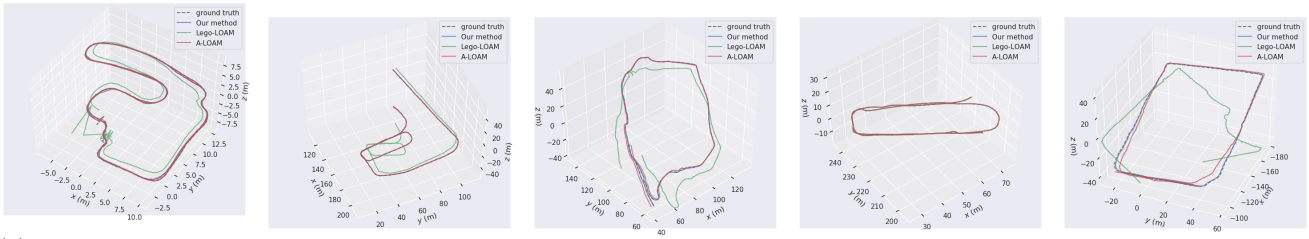
B. Map Optimization

The LiDAR intensity odometry generates a transformation matrix

$$\hat{\mathbf{T}} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix}$$

between current sensor frame and the map frame. In this module, we jointly optimize the scan-to-map residual and LiDAR BA residual to correct the pose drift.

LiDAR Bundle Adjustment (BA): Similar to visual SLAM BA, we can use the LiDAR BA (a non-linear optimization problem) to correct the drift. With this strategy, the last k frames are used in a residual function. We remove the oldest frame if the number of frames is larger than k, and add the newest frame to the sliding window as Fig. 3 showed. We can then match the current frame with the last k frames in the window and store the matched 3D feature points in $\mathcal{P}^c = \{\mathbf{P}_0^c, \mathbf{P}_1^c, \cdots, \mathbf{P}_k^c\}$,



(a) Trajectories in outdoor environment with up and down stairs (191m).

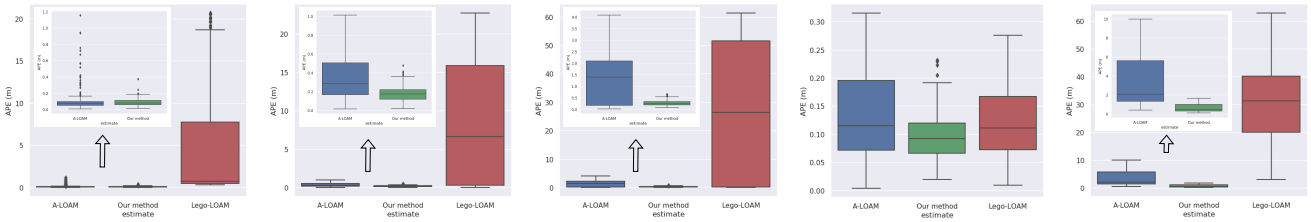
(b) Trajectories in outdoor environment with steep slope (414m).

(c) Trajectories on mountain with a long slope (507m).

(d) Trajectories in parking lot with flat ground (249m).

(e) Trajectories on the street with revisiting the start point (478m).

Fig. 4: Trajectories results in multiple environments. The experimental results prove that our method is able to estimate the position accurately in various scenarios. LeGO-LOAM's algorithm works well in flat environments, but not in environments with slopes.



(a) (b) (c) (d) (e)

Fig. 5: Absolute Pose Error in multiple environments. The APE of our method is the smallest in all these scenarios, followed by A-LOAM.

and $\mathcal{F}^l = \{\mathbf{F}_0^l, \mathbf{F}_1^l, \cdots, \mathbf{F}_k^l\}$ for current and last k frames, respectively. Where $\mathbf{P}_i^c = \{\mathbf{P}_{i0}^c, \mathbf{P}_{i1}^c, \cdots, \mathbf{P}_{im}^c\}$, $\mathbf{P}_{ij}^c = [p_x^{ij}, p_y^{ij}, p_z^{ij}, 1]^T$, The transformation matrix of the last k frames are calculated and stored in $\hat{\mathcal{T}}^l = \{\hat{\mathbf{T}}_0, \hat{\mathbf{T}}_1, \cdots, \hat{\mathbf{T}}_k\}$ by the previous map optimization step. The residual function is therefore:

$$\boldsymbol{f}_b = \sum_{i=0}^{k} \sum_{j=0}^{m} (\hat{\mathbf{T}}\mathbf{P}_{ij}^c - \hat{\mathbf{T}}_i \mathbf{F}_{ij}^l) \tag{4}$$

where $\boldsymbol{f}_b$ is the residual of LiDAR BA. In addition, we treat $\hat{\mathbf{T}}$ from the intensity odometry as an initial pose of current frame.

Scan-to-Map: To match with previous plane feature points, we create an ikd-Tree [35] map, which is a incremental 3D k-d Tree. This map incrementally updates a k-d tree with incoming plane feature points. Compared to static k-d trees, this method has lower computation cost. For plane features, we divide the plane into ground plane and general plane features. When meeting a flat area, adding ground plane constraints is better than plane only method since the ground plane can constrain the roll and pitch direction more accurately. We extract normal plane features like in [2]. As for the ground plane, we first give an initial height of robot to extract possible ground planes and then use RANSAC to segment the ground plane points. The plane features are extracted from the point cloud, and are stored in $\mathcal{P}^m = \{\mathbf{P}_0^m, \mathbf{P}_1^m, \cdots, \mathbf{P}_n^m\}$, where $\mathbf{P}_0^m = \{p_x, p_y, p_z\}^T$. For each plane feature point $\mathbf{P}_i^p$, we can find 3 nearby plane points from the ikd-Tree map, and store them in $\mathcal{F}^m = \{\mathbf{F}_0^m, \mathbf{F}_1^m, \cdots, \mathbf{F}_n^m\}$, where $\mathbf{F}_i^m = \{\mathbf{P}_i^{m0}, \mathbf{P}_i^{m1}, \mathbf{P}_i^{m2}\}$. The residual function can be formulated as:

$$\hat{\mathbf{P}}_i^m = \mathbf{R}\mathbf{P}_i^m + \mathbf{T} \tag{5}$$

$$\boldsymbol{f}_s = \sum_{i=0}^{n} (\hat{\mathbf{P}}_i^m - \mathbf{P}_i^{m0})^T \cdot \left(\frac{(\mathbf{P}_i^{m1} - \mathbf{P}_i^{m0}) \times (\mathbf{P}_i^{m2} - \mathbf{P}_i^{m0})}{|(\mathbf{P}_i^{m1} - \mathbf{P}_i^{m0}) \times (\mathbf{P}_i^{m2} - \mathbf{P}_i^{m0})|} \right) \tag{6}$$

where $\hat{\mathbf{P}}_i^m$ is the 3D mapped point which was converted by $\mathbf{P}_i^m$ from the sensor to the map coordinate system through (5). $\mathbf{P}_i^{m0}$, $\mathbf{P}_i^{m1}$, and $\mathbf{P}_i^{m2}$ are nearby points of $\hat{\mathbf{P}}_i^m$ on the map.

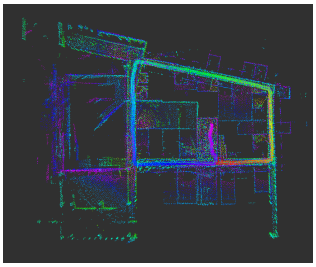
After getting the residual function of LiDAR BA and scan-to-map, we can formulate the whole optimization problem as:

$$\arg\min_{\mathbf{R},\mathbf{T}} (\boldsymbol{f}_b + \boldsymbol{f}_s) \tag{7}$$

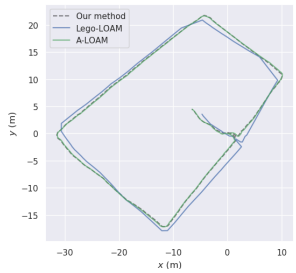
C. Pose Graph Optimization

During map optimization, we can get a better pose estimation of the current frame. Once done, we can use optimized results to correct the drift for the future frames and publish high-frequency optimized odometry in real time. In the backend, we build a pose graph on top of map optimization with LiDAR keyframes. First of all, we extract keyframes from whole LiDAR frames with three criteria:

- The distance between current frame and last keyframe is larger than a threshold.

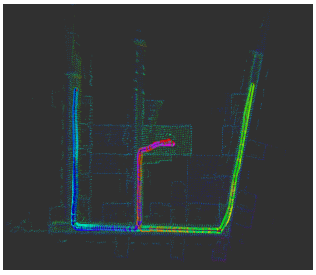


(a) Map generated by our method in the long corridor environment.

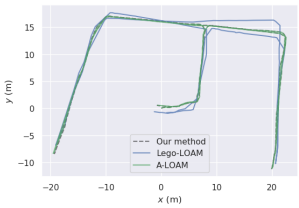


(b) Trajectories of A-LOAM, LeGO-LOAM and our method.

Fig. 6: Map and trajectories of the Spot robot in a building with long corridors. In this scene, we walked along the corridor back to the starting point. In this experiment, the drift of LeGO-LOAM is relatively large, and the performance of A-LOAM is close to our method, but it is not clear from these two graphs which method is better.



(a) Map result of our method with a different path.



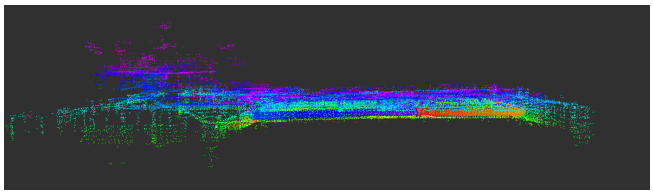
(b) Trajectories of different methods.

Fig. 7: Instead of going back around in a circle to the starting point, we went back and forth through all the corridors for this test and then returned to the starting point. This situation is extreme because there is no good loop to correct the drift in the pitch direction. The experimental results show that LeGO-LOAM drifts more than in the previous scenario. A-LOAM and our method are almost the same.

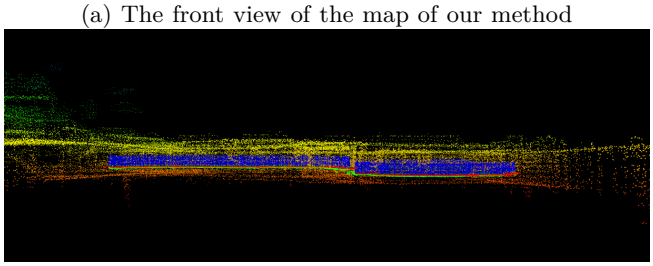
- The angle between two keyframes is larger than a threshold.
- The number of matched feature points is less than a threshold.

We can use the optimized poses of keyframes as the vertexes of the pose graph, and the relative poses between two keyframes as the edges of pose graph.

We also add loop constraints to the pose graph, as Fig. 3 shows, we use the latest keyframe as the anchor frame. With a trained vocabulary, we can compare descriptors of the current keyframe with a database where the history descriptors are stored. If we can not match it with history descriptors, then no loop closure is found for this keyframe [27]. If we successfully match a previous keyframe, we can then put it into the outlier rejection procedure to test whether it is an incorrect loop closure. If it is positive, we can add this loop constraint between the current and the loop candidate factor node in the pose graph. Finally, we use g2o [36] to solve the pose graph optimization problem.

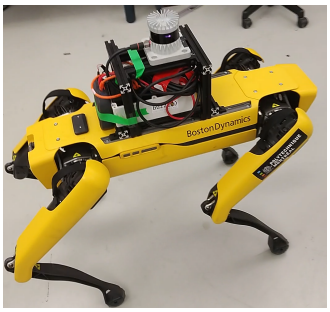


(a) The front view of the map of our method



(b) The front view of the map of A-LOAM

Fig. 8: Difference between our method and A-LOAM in long corridors. As we can see, the map generated by our method is smoothly connected to the starting position, but the map generated by A-LOAM has a clear break at the end.



(a) Spot robot for indoor experiment

(b) Indoor corridor

Fig. 9: Indoor environment and Spot robot used for testing our algorithm. Spot with Ouster LIDAR mounted on it walking and collecting data in the corridor shown on the right.

IV. EXPERIMENTAL RESULTS

To prove our algorithm's reliability, we present our experimental results in an indoor environment with long corridors (Fig. 9b), a multi-storey indoor environment, a mountain, and a street environment. The reason why we chose these environment is that they are all different from each other, and they are challenging for pure LiDAR SLAM. In the indoor environment, we have long corridors and narrow passages. In the mountain environment, we have steep slopes and narrow passages. In the street environment, we have many obstacles and many turns. Those scenarios are difficult for pure LiDAR SLAM to handle. In the following, we will introduce the details of our experiments.

In our experiments, we compared our approach with two other popular pure LiDAR SLAM systems: LeGO-LOAM [7] and A-LOAM [6]. The A-LOAM and LeGO-LOAM algorithms were obtained from their open-source code available on Github. The attempt to compare

TABLE I: Time consumption of different algorithms (ms)

	A-LOAM	LeGO-LOAM	Ours
Features extration	6.33 ± 1.99	10.78 ± 4.36	10.70 ± 1.50
Scan registration	20.04 ± 4.06	2.53 ± 3.88	3.15 ± 2.07
Map optimization	10.63 ± 2.61	21.61 ± 7.55	29.02 ± 6.23
PGO	N/A	23.48 ± 7.81	1.73 ± 10.51

the open-source Intensity-SLAM [11] was unsuccessful as we could not run it. Our intensity based SLAM system shows competitive performance in different environments, including indoor, outdoor, and some extreme scenarios, such as long corridors. Most other LiDAR SLAM systems fail in such extreme environment with fewer edge features. We first tested our method with a public dataset provided by Shan et al. [27] which was collected by Ouster OS1-128 LiDAR. LIO-SAM's trajectory [5] was treated as the ground truth since it is estimated based on LiDAR, 9-axis IMU and GPS, which is much more accurate than LiDAR only SLAM. In Fig. 4, we present the trajectories of our method, LeGO-LOAM, and A-LOAM in various terrains, while Fig. 5 shows the corresponding Absolute Position Error (APE) of Fig. 4. From Fig. 5, we can say our method has a significantly lower (T-test [37], p<1e-5) APE than others, except for the scenario in Fig. 5a where our result is not significant (T-test, p=0.857 compared with A-LOAM). The results in Fig. 4 and Fig. 5 prove our proposed approach achieves competitive results compared to both A-LOAM and LeGO-LOAM, especially in Fig. 4d where the trajectory is close to the ground truth trajectory in the end, while others drift a lot. The effectiveness of LeGO-LOAM is limited to level terrains as it relies on a ground plane constraint. It becomes challenging to extract the ground plane information in uneven terrains, thereby hindering its performance in such environments.

We also test our algorithm indoors with a Spot robot (from Boston Dynamics) equipped with Ouster Os0-64 LiDAR (Fig. 9a). The scene of this experiment mainly contains the same long corridor as Fig. 9b. In this scenario, we ran different algorithms for testing the ability of localization and map building in real world. Both Fig. 6a and Fig. 7a are the maps generated by our algorithm. Fig. 6b and Fig. 7b then show the trajectories of differnt algorithms in the corresponding environments. From the trajectories, we can see that LeGO-LOAM drifts a lot. A-LOAM's and our trajectories are almost the same. Due to the indoor environments, we can hardly collect the ground truth trajectories with RTK. So we try to analyze the map details (Fig. 8) to evaluate the algorithm strengths and weaknesses. Fig. 8a shows the front view of Fig. 6a, and Fig. 8b shows the front view of the same scenario generated by A-LOAM. We can see that our method can smoothly connect the start point at the end of the trip, but A-LOAM's map is disconnected. At this point, we can say that our method is more reliable in such an extreme environment. In addition, we analyzed the time consumption of different SLAM algorithms on the Intel processor with the same data collected by Os0-64 LiDAR. Table I shows that our intensity-based front-end is able to calculate the odometry within 15 ms and our method is efficient enough to meet the real-time requirements of 10 Hz LiDAR.

V. CONCLUSIONS

In this paper, we propose a novel intensity-based pure LiDAR SLAM method. We first proposed a novel lightweight intensity-based odometry method, which directly match 3D features point extracted from the intensity images. Then we propose a novel map optimization method, which jointly optimizes the LiDAR BA and point-to-plane residuals. Finally, we propose a novel intensity-based pose graph optimization method, which can optimize the pose graph based on the intensity image. We tested our method in both outdoor and indoor environments. The results proved that our method can achieve competitive results compared with other popular pure LiDAR SLAM methods. In the future, we will further improve our method by using more advanced feature extraction methods and loop closure detection methods. We will also test our method in more challenging environments.

References

[1] 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.

[2] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time." in Robotics: Science and Systems, vol. 2, no. 9, 2014.

[3] T. Shan, B. Englot, C. Ratti, and D. Rus, "LVI-SAM: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in 2021 IEEE international conference on robotics and automation (ICRA). IEEE, 2021, pp. 5692–5698.

[4] Y. Cao and G. Beltrame, "VIR-SLAM: Visual, inertial, and ranging slam for single and multi-robot systems," Autonomous Robots, vol. 45, no. 6, pp. 905–917, 2021.

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

[6] Q. Tong and C. Shaozu, "A-LOAM: Advanced implementation of loam," https://github.com/HKUST-Aerial-Robotics/A-LOAM, 2019.

[7] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018, pp. 4758–4765.

[8] K. Ebadi, M. Palieri, S. Wood, C. Padgett, and A.-a. Aghamohammadi, "DARE-SLAM: Degeneracy-aware and resilient loop closing in perceptually-degraded environments," Journal of Intelligent & Robotic Systems, vol. 102, pp. 1–25, 2021.

[9] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast direct lidar-inertial odometry," IEEE Transactions on Robotics, 2022.

[10] H. Li, B. Tian, H. Shen, and J. Lu, "An intensity-augmented lidar-inertial slam for solid-state lidars in degenerated environments," IEEE Transactions on Instrumentation and Measurement, vol. 71, pp. 1–10, 2022.

[11] H. Wang, C. Wang, and L. Xie, "Intensity-SLAM: Intensity assisted localization and mapping for large scale environment," IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 1715–1721, 2021.

[12] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," in Autonomous robot vehicles. Springer, 1990, pp. 167–193.

[13] B. Huang, J. Zhao, and J. Liu, "A survey of simultaneous localization and mapping," arXiv preprint arXiv:1909.05214, 2019.

[14] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in 2016 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2016, pp. 1271–1278.

[15] K. Chen, B. T. Lopez, A.-a. Agha-mohammadi, and A. Mehta, "Direct LiDAR Odometry: Fast Localization with Dense Point Clouds," IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 2000–2007, 2022.

[16] J. Lin and F. Zhang, "A fast, complete, point cloud based loop closure for lidar odometry and mapping," arXiv preprint arXiv:1909.11811, 2019.

[17] ——, "Loam livox: A fast, robust, high-precision LiDAR odometry and mapping package for LiDARs of small fov," in 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020, pp. 3126–3131.

[18] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss, "SuMa++: Efficient LiDAR-based Semantic SLAM," in Proceedings of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2019.

[19] X. Chen, T. Läbe, A. Milioto, T. Röhling, J. Behley, and C. Stachniss, "OverlapNet: A siamese network for computing LiDAR scan similarity with applications to loop closing and localization," Autonomous Robots, pp. 1–21, 2022.

[20] D. Cattaneo, M. Vaghi, and A. Valada, "LCDNet: Deep Loop Closure Detection and Point Cloud Registration for LiDAR SLAM," IEEE Transactions on Robotics, vol. 38, no. 4, pp. 2074–2093, 2022.

[21] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: A 3D Convolutional Neural Network for real-time object class recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652–660.

[22] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space," Advances in neural information processing systems, vol. 30, 2017.

[23] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "SegMatch: Segment based place recognition in 3D point clouds," in 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017, pp. 5266–5272.

[24] M. A. Uy and G. H. Lee, "PointNetVLAD: Deep Point Cloud Based Retrieval for Large-Scale Place Recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4470–4479.

[25] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN Architecture for Weakly Supervised Place Recognition," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 5297–5307.

[26] T. D. Barfoot, C. McManus, S. Anderson, H. Dong, E. Beerepoot, C. H. Tong, P. Furgale, J. D. Gammell, and J. Enright, "Into Darkness: Visual Navigation Based on a Lidar-Intensity-Image Pipeline," in Robotics Research: The 16th International Symposium ISRR. Springer, 2016, pp. 487–504.

[27] T. Shan, B. Englot, F. Duarte, C. Ratti, and D. Rus, "Robust Place Recognition using an Imaging Lidar," 2021 IEEE International Conference on Robotics and Automation (ICRA), pp. 5469–5475, 2021.

[28] T. Guadagnino, X. Chen, M. Sodano, J. Behley, G. Grisetti, and C. Stachniss, "Fast sparse lidar odometry using self-supervised feature selection on intensity images," IEEE Robotics and Automation Letters, vol. 7, no. 3, pp. 7597–7604, 2022.

[29] L. Di Giammarino, I. Aloise, C. Stachniss, and G. Grisetti, "Visual place recognition using lidar intensity information," in 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2021, pp. 4382–4389.

[30] Z. Liu and F. Zhang, "BALM: Bundle Adjustment for Lidar Mapping," IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 3184–3191, 2021.

[31] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in Proceedings third international conference on 3-D digital imaging and modeling. IEEE, 2001, pp. 145–152.

[32] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," IEEE transactions on robotics, vol. 31, no. 5, pp. 1147–1163, 2015.

[33] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," IEEE Transactions on Robotics, vol. 33, no. 5, pp. 1255–1262, 2017.

[34] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM," IEEE Transactions on Robotics, 2021.

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

[36] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in 2011 IEEE International Conference on Robotics and Automation. IEEE, 2011, pp. 3607–3613.

[37] T. K. Kim, "T test as a parametric statistic," Korean journal of anesthesiology, vol. 68, no. 6, pp. 540–546, 2015.