LIO-DOR: A Robust LiDAR Inertial Odometry with Real-time Dynamic Object Removal

Lingjian Mao[1], Wei Gao[1,*], Haoyao Chen[2] and Shiwu Zhang[1,*]

2023 IEEE International Conference on Robotics and Biomimetics (ROBIO) | 979-8-3503-2570-6/23/$31.00 ©2023 IEEE | DOI: 10.1109/ROBIO58561.2023.10355038

***Abstract*— To achieve better simultaneous localization and mapping (SLAM) under dynamic environments, this paper presents a novel robust pipeline for efficiently removing dynamic objects from the point cloud acquired by LiDAR scans. The proposed method realizes dynamic object identification through clustering cloud points and computing the collision volume between the same cluster from consecutive scans. The dynamic object removal is achieved at object level instead of point level, which ensures purer maps for subsequent path planning and navigation tasks. Extensive experiments are performed both indoors and outdoors using the open-source M2DGR dataset and a self-made dataset. The experimental results demonstrate that the proposed method outperforms mainstream SLAM methods in dynamic environments.**

I. INTRODUCTION

With the development of LiDARs, cameras, IMUs, etc., simultaneous localization and mapping (SLAM) has witnessed remarkable advancement in recent years. Particularly, the combination of LiDAR and IMU has garnered significant attention due to its favorable prospects for practical implementation. For instance, the most widely used method LOAM [1] introduces a feature extraction strategy based on curvatures to divide LiDAR scanned points into corner and planar points, reducing computational cost in state estimation. Built upon LOAM, LIO-SAM [2] further integrates IMU to improve localization accuracy in LiDAR-limited scenarios. On the other hand, FAST-LIO2 [3] employs a novel ike-tree data structure to achieve efficient and robust LiDAR-inertial odometry through error state Kalman filter.

These mainstream SLAM methods mostly operate under the assumption that the environment is static and the objects in it are rigid. However, the environment in reality where mobile robots operate is inevitably dynamic, filled with moving objects such as pedestrians, vehicles, etc. Moreover, these dynamic objects often exhibit non-rigid characteristics, like the swing of pedestrians' limbs, posing great challenges to SLAM systems. Especially at the pose estimation stage, dynamic objects tend to occupy LiDAR's field of view and cause erroneous information for point cloud registration, thus severely impacts localization accuracy and can even lead to system failures. Even if the influence of dynamic objects on localization accuracy can be neglected, the ghost artifacts caused by dynamic objects within the generated maps are still adverse to subsequent tasks, such as area segmentation and path planning.

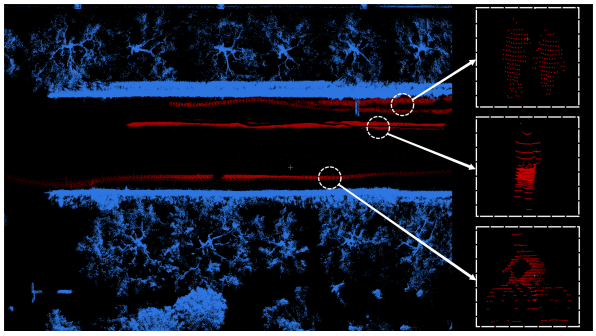


Fig. 1. Identification of dynamic objects with LIO-DOR. Static objects are in blue, while dynamic objects are in red. The ground has been removed for clarity. The zoomed-in insets show dynamic objects of a single frame.

To address this challenge, there are mainly two approaches: removing the dynamic objects in real-time during the SLAM process, or handling the dynamic objects in post-processing of the generated point cloud map. We believe that the adverse effects of dynamic objects on localization cannot be neglected, thus this paper proposes a novel real-time dynamic object removal method within the tightly-coupled LiDAR-inertial odometry framework, named LIO-DOR, to ensure robust SLAM in dynamic environments.

The main contributions of this paper are

- A new SLAM method that achieves real-time dynamic object detection and removal using IMU preintegration back-propagation and collision volume differentiation. No pre-trained networks are involved.
- The proposed method exhibits high-level LiDAR adaptability, applicable to both mechanical and solid-state LiDAR sensors.
- The proposed method is evaluated on both public and self-made datasets, with experimental results demonstrating superior performance in dynamic environments comparing to mainstream SLAM methods.

II. RELATED WORK

To remove the dynamic objects in real-time during the SLAM process, researchers have proposed two major approaches.

One approach involves generating a pre-trained dynamic object detection model through deep learning. The model

The authors gratefully acknowledge the support provided by the National Natural Science Foundation of China under U21A20119, U22B2040 and 62103395. Thanks to JT-Innovation (Guangdong) Intelligent Technology Co., Ltd. for providing the Livox Mid-360 LiDAR used in this work.

[1]Department of Precision Machinery and Precision Instrumentation, School of Engineering Science, University of Science and Technology of China, Hefei, Anhui 230026, China.

[2]College of Mechanical Engineering and Automation, Harbin Institute of Technology Shenzhen, Shenzhen, Guangdong 518055, China.

[*]Corresponding authors:
swzhang@ustc.edu.cn, weigao@ustc.edu.cn.

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on October 19,2024 at 03:58:38 UTC from IEEE Xplore. Restrictions apply.

takes raw sensor data as input and output high-quality information for point cloud registration. This method is commonly applied especially in visual SLAM and can accurately remove pedestrians, vehicles, etc. For instance, the method proposed by the Autonomous Systems Lab at ETH Zurich [4] utilizes the 3D-MiniNet network for real-time dynamic object detection. However, this approach not only requires a large amount of training data and computational resources but also relies heavily on accurate data annotation under substantial manual efforts. Additionally, this method primarily focuses on categories rather than properties of dynamic objects.

The other approach involves tracking the temporal position changes of points through traditional methods. For example, Schauer and Nuchter propose a dynamic filtering method that builds a regular voxel occupancy grid, traverses the grid along the lines of sight between the sensor and the measured points, and determines whether a point in the grid is dynamic based on whether the grid is hit or penetrated [5]. This method has general implications for grid-based approaches and has shown to be effective to some extent. Another method ERASOR [6] divides the scan frames and nearby submaps into sector-shaped bins at the same position, and removes dynamic points based on the differences of point cloud distribution within the bins. On the other hand, researchers have also directed their attention to range images. RF-LIO [7] performs coarse registration in the front-end odometry to provide initial values and conducts "scan-to-submap" fine registration in the back-end. During the fine registration iterations, dynamic points in the submap are continuously detected and removed based on the initial values and multi-resolution depth images. This approach achieves precise alignment based on "scan-to-static" registration. A similar concept is validated in [8], where the scan frame is projected into a depth image, as well as the submap near the same viewpoint. By comparing the depths of corresponding pixels on both depth images, a point is considered dynamic if there is a significant depth variation. To summarize, these methods are capable of removing dynamic points to some extent, but some of them are post-processing methods, and some often require a significant number of laser beams to generate high-resolution depth images. More importantly, the removal of dynamic objects in these methods is primarily in the manner of removing points rather than entire objects.

III. LIO-DOR

A. System overview

Figure 2 illustrates the overall framework of LIO-DOR. LIO-DOR mainly consists of three modules: the IMU preintegration module, the dynamic object removal module, and the mapping module. Firstly, the IMU preintegration module infers system motion and generates IMU odometry, which supports both the dynamic object removal and the mapping modules. Secondly, the dynamic object removal module detects and removes dynamic objects in real-time from the raw point cloud, which involves several key steps differing from mainstream SLAM methods: (i) After compensating the motion distortion of point cloud upon the arrival of a new laser scan, LIO-DOR does not immediately perform point cloud registration for pose estimation. Instead, it puts points with obvious static characteristics like ground points and backdrop points to the static point set, and takes the remaining as foreground points. (ii) The foreground points are then aligned with those from previous scan using the pose transformation obtained from IMU preintegration. (iii) Clustering on these aligned foreground points is performed and the Oriented Bounding Boxes are computed. (iv) For each bounding box, LIO-DOR searches for the most similar one from the previous frame based on the minimum distance, and the collision volume between the two bounding boxes determines whether the bounded object is dynamic. Upon the convergence of the entire point cloud segmentation, the remaining points are added to the static point set. Finally, the mapping module calculates the LiDAR odometry based on the static point set, which is tightly coupled with the IMU preintegration results in graph optimization to achieve high-precision pose estimation.

The following sub-sections provide detailed information about each module within LIO-DOR.

B. The IMU preintegration module

We label the world frame as W and the IMU frame which coincides with the robot body frame as B. The system states are described as

$$X = [\boldsymbol{p}, \boldsymbol{v}, \boldsymbol{R}, \boldsymbol{a}^b, \boldsymbol{\omega}^b, \boldsymbol{g}], \tag{1}$$

where $\boldsymbol{p}$ is the position vector, $\boldsymbol{v}$ is the speed vector, $\boldsymbol{R}$ is the rotation matrix from W to B, $\boldsymbol{a}^b$ is the IMU accelerometer bias vector, $\boldsymbol{\omega}^b$ is the gyroscope bias vector, and $\boldsymbol{g}$ is the gravitational acceleration vector. Therefore, the IMU measurement model can be constructed as

$$\begin{cases} \boldsymbol{a}_t^m = \boldsymbol{R}_t^T(\boldsymbol{a}_t - \boldsymbol{g}_t) + \boldsymbol{a}_t^b + \boldsymbol{a}_t^n, \\ \boldsymbol{\omega}_t^m = \boldsymbol{\omega}_t + \boldsymbol{\omega}_t^b + \boldsymbol{\omega}_t^n, \end{cases} \tag{2}$$

where $\boldsymbol{a}_t^m$ and $\boldsymbol{\omega}_t^m$ are the measurements of the IMU ground truth $\boldsymbol{a}_t$ and $\boldsymbol{\omega}_t$ in B at time t, affected by the bias $\boldsymbol{a}_t^b$ and $\boldsymbol{\omega}_t^b$, and the white noise $\boldsymbol{a}_t^n$ and $\boldsymbol{\omega}_t^n$, $\boldsymbol{R}_t^T$ is the rotation matrix from B to W at time t, respectively.

When the IMU measurement at time $t + \Delta t$ is received by the system, the IMU preintegration based on the pose at previous time t can be performed as

$$\begin{cases} \boldsymbol{v}_{t+\Delta t} = \boldsymbol{v}_t + \boldsymbol{g}\Delta t + \boldsymbol{R}_t(\boldsymbol{a}_t^m - \boldsymbol{a}_t^b - \boldsymbol{a}_t^n)\Delta t, \\ \boldsymbol{p}_{t+\Delta t} = \boldsymbol{p}_t + \frac{1}{2}(\boldsymbol{v}_t + \boldsymbol{v}_{t+\Delta t})\Delta t, \\ \boldsymbol{R}_{t+\Delta t} = \boldsymbol{R}_t \exp\left((\boldsymbol{\omega}_t^m - \boldsymbol{\omega}_t^b - \boldsymbol{\omega}_t^n)\Delta t\right), \\ \boldsymbol{R}_{\Delta t} = \boldsymbol{R}_t^T \boldsymbol{R}_{t+\Delta t}, \\ \boldsymbol{p}_{\Delta t} = \boldsymbol{R}_t^T(\boldsymbol{p}_{t+\Delta t} - \boldsymbol{p}_t - \boldsymbol{v}_t\Delta t - \frac{1}{2}\boldsymbol{g}\Delta t^2), \end{cases} \tag{3}$$

where $\boldsymbol{R}_{\Delta t}$ is the rotation matrix between the two frames [9], and $\boldsymbol{p}_{\Delta t}$ is the translation vector between the two frames. We utilize a factor graph to optimize the IMU bias alongside the LiDAR odometry factors. As a result, we can obtain the pose transformation relationship between the two frames as

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{R}_{\Delta t} & \boldsymbol{p}_{\Delta t} \\ \boldsymbol{0} & 1 \end{bmatrix}. \tag{4}$$

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on October 19,2024 at 03:58:38 UTC from IEEE Xplore. Restrictions apply.

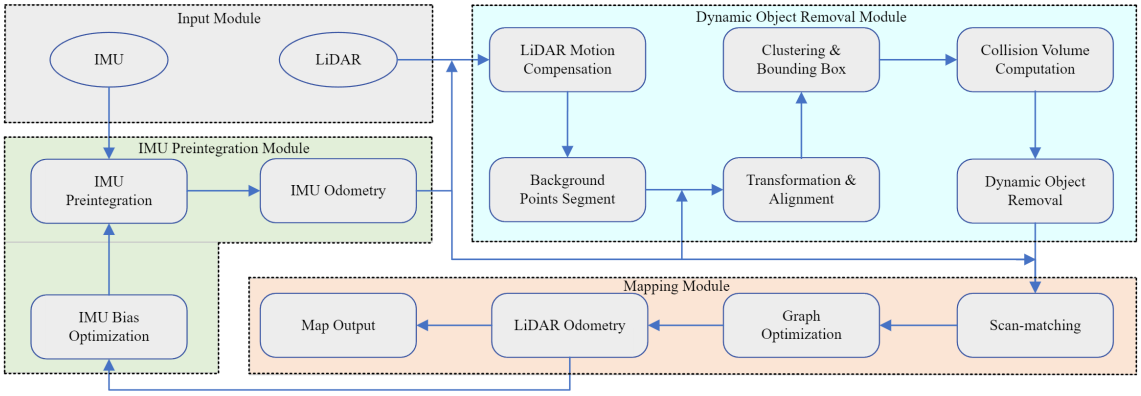


Fig. 2. The overarching framework of LIO-DOR involves real-time data input from a 3D LiDAR and an IMU. The data undergo three main modules: (a) the IMU preintegration module, (b) the dynamic object removal module, and (c) the mapping module.

C. The dynamic object removal module

When a new LiDAR scan frame arrives, motion distortion compensation is first performed by utilizing the timestamp information of the point cloud to find the nearest IMU data for each point. With the high-frequency IMU data, a transformation matrix can be calculated to project each point to the timestamp of the first point in that scan, obtaining a consistent scan frame. Subsequently, the process of dynamic object removal is described in the following steps:

1) Identification of background points: To extract obvious static points from the LiDAR scan data of the $(i+1)$-th frame P_{scan}^{i+1}, ground points are identified via the Principal Component Analysis (PCA) method [10] to obtain the normal vector $\vec{\boldsymbol{v}}_2$ for each point. To reduce the computational complexity of plane fitting, $\vec{\boldsymbol{v}}_2$ is calculated as the minimum eigenvector of the covariance matrix in the neighborhood of each point, demonstrated as

$$\begin{cases} \boldsymbol{c} = \frac{1}{k}\sum_{j=1}^{k}(p_j - \frac{1}{k}\sum_{i=1}^{k}p_i)(p_j - \frac{1}{k}\sum_{i=1}^{k}p_i)^T, \\ \boldsymbol{c}\cdot\vec{\boldsymbol{v}}_j = \lambda_j\cdot\vec{\boldsymbol{v}}_j,\ j\in\{0,1,2\}, \end{cases} \tag{5}$$

where p_i represents the points within the neighborhood of p_j, $\boldsymbol{c}$ is the covariance matrix describing these points, λ_j represents the eigenvalues, and $\vec{\boldsymbol{v}}_j$ corresponds to the eigenvectors. Points with a normal vector oriented within 15 degrees to the ideal upward vector are considered as potential ground points, where the value of 15 degrees is empirically decided to reduce the computational workload in plane fitting. For ground plane fitting, the RANSAC [11] is used to minimize the sum of distances from each point to the fitted plane. After obtaining the plane equation of the ground, points within a certain distance to the plane are considered as ground points P_{ground}^{i+1} and added to the static point set. Subsequently, the region-growing method is employed to segment the tall backdrop points, like trees and buildings, as $P_{backdrop}^{i+1}$ from the remaining points. Specifically, points with heights exceeding a predefined maximum height threshold are classified as seed points. Then for each seed point, its neighborhood points within an appropriate distance are searched and added to the seed point set accordingly until no points can be further added. At this point, the static point set can be denoted as

$$P_{background}^{i+1} = P_{ground}^{i+1} + P_{backdrop}^{i+1}, \tag{6}$$

and the remaining points are referred to as foreground points,

$$P_{foreground}^{i+1} = P_{scan}^{i+1} - P_{background}^{i+1}. \tag{7}$$

2) Calculation of transformation matrix and projection: Here Equation (4) is employed to obtain the pose transformation matrix $M_{cur\text{-}pre}$ between current and previous point clouds, which projects the foreground points $P_{foreground}^{i+1}$ from current frame to previous frame as

$${}^{align}P_{foreground}^{i+1} = \boldsymbol{M}_{cur\text{-}pre}P_{foreground}^{i+1}, \tag{8}$$

where ${}^{align}P_{foreground}^{i+1}$ is the point cloud after projection. An example of the foreground points projection is demonstrated in Figure 3.

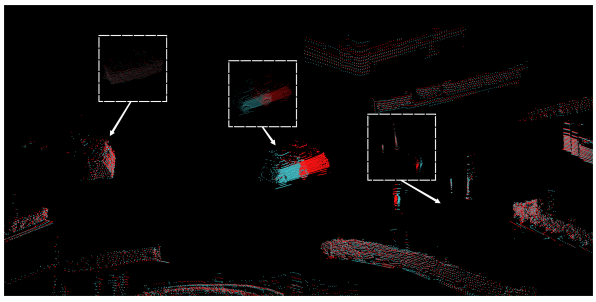


Fig. 3. Foreground points projection. Points from previous frame are in blue, while points from current frame are in red. Stationary objects exhibit overlapping, while dynamic objects exhibit misalignment.

3) Foreground points clustering and bounding box calculation: DBSCAN [12] is a widely used density-based clustering method that groups points based on their density and connectivity in the spatial domain. One of the pronounced advantages of DBSCAN lies in its capability to effectively handle noisy data to ensure robust clustering, as noise points are automatically excluded from the clustering process. This

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on October 19,2024 at 03:58:38 UTC from IEEE Xplore. Restrictions apply.

is exactly what LIO-DOR needs, and thus the DBSCAN algorithm is applied to categorize the foreground points as

$$^{align}P^{i+1}_{foreground} = \{^{align}C^{i+1}_1, ^{align}C^{i+1}_2, ..., ^{align}C^{i+1}_k\}. \quad (9)$$

For each cluster $^{align}C^{i+1}_k$, its Oriented Bounding Box is computed using the method proposed in [13] as

$$^{align}B^{i+1}_k = \{\boldsymbol{O}_c + ar_1\vec{\boldsymbol{v}}_1 + br_2\vec{\boldsymbol{v}}_2 + cr_3\vec{\boldsymbol{v}}_3 | a, b, c \in [0, 1]\}, \quad (10)$$

where $\boldsymbol{O}_c = (x_c, y_c, z_c)$ is the center point of the Oriented Bounding Box, and r_i and $\boldsymbol{v}_i$ represent the half-length and the unit direction vector for one of the three edges, respectively.

4) Dynamic objects removal: To determine whether an object within its Oriented Bounding Box is dynamic, the Oriented Bounding Box from previous frames B^i_j is also obtained to compute their collision volume, as shown in Figure 4. To reduce the computational burden, the two 3D bounding boxes $^{align}B^{i+1}_k$ and B^i_j are projected into 2D bounding boxes $^{align}_{2d}B^{i+1}_k$ and $_{2d}B^i_j$ on the XOY plane. Subsequently, the ratio $rate_{xoy}$ of the intersection area between the two 2D bounding boxes over the average area, and the ratio $rate_z$ of the overlapped Z-axis interval between the two 3D bounding boxes over the average interval, are calculated. Therefore, the overall collision volume ratio $rate_{overlap}$ between the two 3D bounding boxes is defined as

$$rate_{overlap} = rate_{xoy} \times rate_z. \quad (11)$$

A threshold λ for the collision volume ratio $rate_{overlap}$ is also defined. Thus when $rate_{overlap} > \lambda$, the object represented by the cluster is considered static; otherwise, the object is dynamic. All the clusters are iteratively processed and segmented into static and dynamic ones. Hence, the foreground points can expressed as

$$^{align}P^{i+1}_{foreground} = ^{align}P^{i+1}_{fore\text{-}static} + ^{align}P^{i+1}_{fore\text{-}dynamic}. \quad (12)$$

Finally, the point cloud within those static clusters are projected back to the original coordinate system using $\boldsymbol{M}^T_{cur\text{-}pre}$ as $P^{i+1}_{fore\text{-}static}$. As a result, the high-quality static point cloud is composed as

$$P^{i+1}_{static} = P^{i+1}_{ground} + P^{i+1}_{backdrop} + P^{i+1}_{fore\text{-}static}. \quad (13)$$

D. The mapping module

The mapping module is fed with the static point cloud P^{i+1}_{static} as input. To estimate the LiDAR odometry, various point cloud registration methods can be employed, in this regard, we draw inspiration from LEGO-LOAM [14], which has exhibited excellently robust performance in complex environments. For the sake of brevity, the subsequent steps are omitted here, as they are thoroughly described in [14]. However, unlike LEGO-LOAM, the use of line and plane features is abandoned in our method to enhance the algorithm's adaptability to LiDAR types and eliminate the need for computational cost associated with feature extraction.

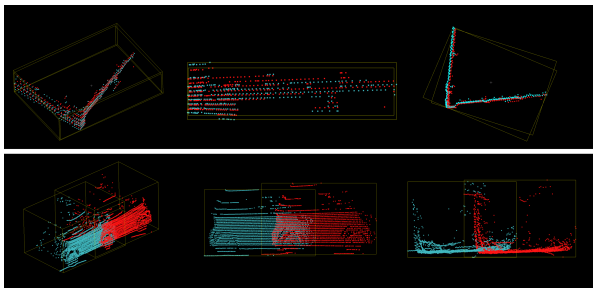


Fig. 4. The Oriented Bounding Boxes. The top row shows the static objects (e.g., railings), while the bottom row shows the dynamic objects (e.g., cars). Points from previous frame are in blue, while points from current frame are in red. From left to right, the images display: (1) bounding boxes, (2) bounding boxes of the height-wise projection, (3) bounding boxes of the XOY plane projection.

Therefore, the distance between corresponding points within two consecutive frames can be computed as

$$d_k = \frac{|(p_{i+1,k} - p_{i,u})[(p_{i,u} - p_{i,v}) \times (p_{i,u} - p_{i,w})]|}{|(p_{i,u} - p_{i,v}) \times (p_{i,u} - p_{i,w})|}, \quad (14)$$

where $p_{i+1,k}$ is the point in the current frame for which we need to calculate the distance, and $p_{i,u}$, $p_{i,v}$, and $p_{i,w}$ are the three closest points to $p_{i+1,k}$ in the previous frame, forming a plane. The Gauss-Newton method is then used to solve for the optimal transformation by minimizing

$$\min_{T_{i+1}}\{\sum_{p_{i+1,k} \in P_{static}} d_k\}. \quad (15)$$

At last, the relative pose transformation $\Delta\boldsymbol{T}$ between two consecutive frames can be obtained to represent the desired LiDAR odometry as

$$\Delta\boldsymbol{T}_{i,i+1} = \boldsymbol{T}^T_i \boldsymbol{T}_{i+1}. \quad (16)$$

We tightly couple the LiDAR odometry and the IMU odometry by adding constraints to the factor graph, obtaining the optimized accurate poses.

IV. EXPERIMENTS

A. Experimental setup

We evaluated the performance of LIO-DOR in various environments using the open-source M2DGR [15] dataset, which includes data from a nine-axis IMU and a mechanical LiDAR, and a self-made dataset, which includes data from a six-axis IMU and a solid-state LiDAR. M2DGR offers all the necessary data required, accompanied by high-frequency ground truth files. Additionally, for mainstream SLAM methods, M2DGR provides official optimal parameter configuration files, which contributes to the persuasiveness of our comparative experiments. The self-made dataset is collected using a sensing box, including a Livox Mid-360 LiDAR with an integrated low-cost six-axis IMU, an OrangePi4 mini single-board computer, a monocular camera, and a power supply system. And the data acquisition frequency from the LiDAR, the camera, and the IMU is 10 Hz, 30 Hz, and 200 Hz, respectively. Note that all datasets were collected at close proximity to the ground and scenarios involving aerial

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on October 19,2024 at 03:58:38 UTC from IEEE Xplore. Restrictions apply.

vehicles will be considered in future work. Details of the used sequences are summarized in Table I.

TABLE I
EXPERIMENTAL SEQUENCE DETAILS

Sequences	Trajectory Length (m)	Characteristics	Dynamic Level
M2DGR-street4	840	outdoor, loop back	Low
M2DGR-street5	469	outdoor, straight	Medium
M2DGR-street6	494	outdoor, one turn	Medium
Our-Indoor	351	indoor, repeated	Medium
Our-Outdoor	344	outdoor, random	Medium

In the comparative experiments, we compared LIO-DOR with FAST-LIO2 [3], LIO-Livox (https://github.com/Livox-SDK/LIO-Livox) and LIO-SAM [2]. We note that 1) LIO-DOR and FAST-LIO2 are applicable to both mechanical and solid-state LiDARs. 2) LIO-SAM utilizes GTSAM for the back-end optimization of factor graph, similar to LIO-DOR. 3) LIO-Livox is exclusively designed for solid-state LiDAR. On the other hand, in the ablation experiments, we conducted a comparative analysis on LIO-DOR by evaluating its performance with and without using our proposed dynamic object removal module. Note that due to the unavailability of open-source code and datasets from similar algorithms like RF-LIO [7], direct experimental comparison was not able to be conducted.

In all experiments, we made efforts to keep the parameters of all methods in their optimal states. All the methods are implemented in C++ and executed on a computer with an AMD Ryzen7 5800H CPU using the Robot Operating System (ROS) in Ubuntu20.04, without using GPU. All the methods only use LiDAR and IMU data, while the GPS data are only used as the ground truth. We utilized RVIZ, PCL and CloudCompare for point cloud visualization and employed the EVO tool for trajectory evaluation.

B. M2DGR dataset

From M2DGR, we selected a straight sequence, a loop closure sequence and an around lawn sequence for comparative experiments. We used two effective metrics to evaluate the performance of the methods: one is the absolute trajectory error compared to the ground truth, and the other is the removal rate of dynamic points obtained through manual annotation, the results of which are shown in Tables II and III, respectively. These metrics effectively reflect the performance of selected methods and the extent of dynamic object removal. Due to the incompatibility of LIO-Livox with mechanical LiDAR data, it was not included in.

From Table II, it can be seen that LIO-DOR exhibits a significantly improved performance under dynamic environments. For example, in the medium dynamic sequence street5, LIO-DOR achieved a reduction of up to 25.63% and 12.51% in absolute trajectory error compared to LIO-SAM and FAST-LIO2, respectively. The results are also presented in Figure 5. Meanwhile, in the corresponding ablation experiments, LIO-DOR with the dynamic object removal module

TABLE II
ABSOLUTE TRAJECTORY RMSE [M] USING M2DGR DATASET.

Sequences	LIO-SAM	FAST-LIO2	LIO-DOR (intact)	LIO-DOR (ablation)
M2DGR-street4	0.916571	0.452403	0.864260	0.897135
M2DGR-street5	0.343641	0.292125	0.255566	0.309996
M2DGR-street6	1.664701	1.625455	1.356711	1.530352

TABLE III
REMOVAL RATE USING M2DGR DATASET.

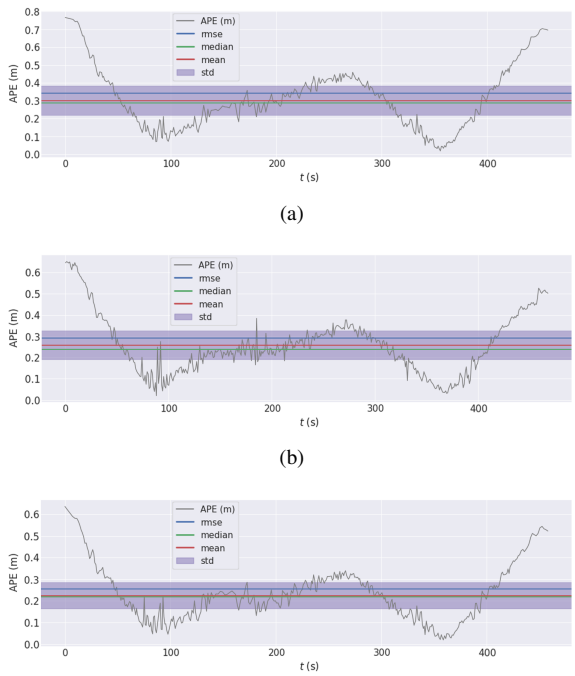
Sequences	LIO-DOR
M2DGR-street4	93.67%
M2DGR-street5	94.34%
M2DGR-street6	93.27%

exhibited a reduction of 17.56% in absolute trajectory error compared to its counterpart without the module, indicating the favorable effectiveness of our dynamic object removal module. However, in the street4 sequence, the performance of LIO-DOR appears to be inferior to FAST-LIO2 and is close to the ablation results, yet it still outperforms LIO-SAM. We attribute this observation to the fact that in low-dynamic sequence, the influence of few dynamic objects on localization accuracy is negligible, and removing them may not lead to a significantly improvement. This further corroborates that LIO-DOR improves SLAM performance by removing dynamic objects. The filter-based FAST-LIO2 has a smaller computational time compared to most optimization-based methods, allowing for more iterations, thus exhibiting better performance. When slightly relaxing the time constraint, LIO-DOR achieves results comparable to FAST-LIO2, with the absolute trajectory error be as low as 0.388900. We will continue to refine this aspect in our future work. On the other hand, LIO-DOR achieved a removal rate of over 93% for dynamic points in the three sequences, effectively removing almost all dynamic points. It is worth noting that with more aggressive parameter settings, the removal rate can even approach 100%. However, when the removal rate reaches 93% or higher, the remaining few dynamic points do not significantly affect localization accuracy and do not leave ghost artifacts in the map. On the contrary, the finer parameters introduce unnecessary performance overhead. Therefore, the dynamic object removal method we proposed is an effective approach to enhance SLAM performance in dynamic environments. Figure 6 illustrates the map generated by the selected methods on the street5 sequence, and the results on other sequences are similar. It can be seen that LIO-SAM and FAST-LIO2 render a lot of ghost tracks in the point cloud map, while LIO-DOR can get a purer map.

C. The self-made dataset

To comprehensively evaluate the performance of LIO-DOR, we collected dataset with dynamic objects in both indoor and outdoor scenarios using our sensing box. In general, we agree that the absolute trajectory accuracy indirectly reflects the mapping precision. However, this may not be

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on October 19,2024 at 03:58:38 UTC from IEEE Xplore. Restrictions apply.



(a)

(b)

(c)

Fig. 5. The distribution of absolute trajectory accuracy for the three methods: (a) LIO-SAM, (b) FAST-LIO2, and (c) LIO-DOR.

entirely accurate in practice. Therefore, in this experiment, in addition to the removal rate of dynamic points, we have also designed a new metric to reflect the mapping accuracy of our method: the mapping consistency index. In the environment, we set up a plane model that can be distinguished based on the reflectivity of the LiDAR, as shown in Figure 7. During data collection, we scanned these plane models multiple times. After map building, we employed RANSAC to extract plane equations and calculated the root mean square error (RMSE) of each point relative to the plane equations. The ground truth of the plane models was obtained through static LiDAR scans. Detailed results can be found in Tables IV and V. Due to the incompatibility of LIO-SAM with solid-state LiDARs, it was not included in.

TABLE IV

REMOVAL RATE USING THE SELF-MADE DATASET.

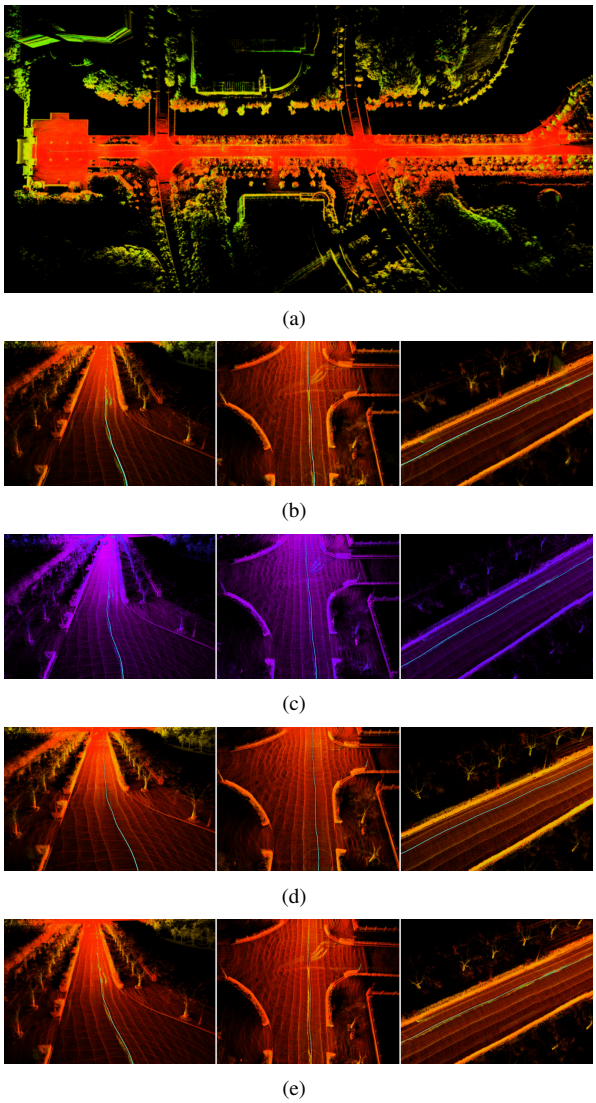
Sequences	LIO-DOR	Sequences	LIO-DOR
Our-Indoor	98.55%	Our-Outdoor	94.68%

TABLE V

MAPPING CONSISTENCY INDEX RMSE [M] USING THE SELF-MADE DATASET.

Sequences	LIO-Livox	FAST-LIO2	LIO-DOR	Ground Truth
Our-Indoor	0.016307	0.014082	0.013582	0.011812
Our-Outdoor	0.043012	0.018501	0.017179	0.012089

In both collected sequences, LIO-DOR consistently



(a)

(b)

(c)

(d)

(e)

Fig. 6. Comparative experiments using the street5 sequence in M2DGR. (a) Overhead view of the map by LIO-DOR, (b) Partial zoom-in of the map by LIO-SAM, (c) Partial zoom-in of the map by FAST-LIO2, (d) Partial zoom-in of the map by LIO-DOR, (e) Control group without the dynamic object removal module enabled in LIO-DOR.

achieves a remarkably high removal rate of dynamic points and exhibits outstanding performance in terms of mapping consistency index. This further demonstrates that the maps constructed using LIO-DOR possess better usability for subsequent planning and navigation tasks. Additionally, the analysis indicates that LIO-DOR performs better indoors compared to outdoors. We speculate that the flat indoor ground and structured environment are more favorable for ground fitting and backdrop point segmentation. In contrast, the complexity of outdoor environment introduces unfavorable factors.

D. Stress experiment

The processing timing experiments is conducted to verify the real-time performance of LIO-DOR. To our knowledge, under the same conditions, the point cloud scale is the most

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on October 19,2024 at 03:58:38 UTC from IEEE Xplore. Restrictions apply.

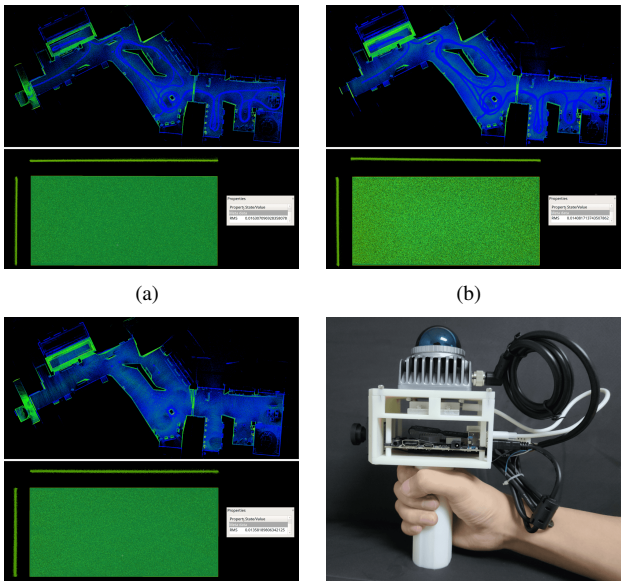


Fig. 7. Comparative experiments using the self-made dataset, results for the indoor sequence are shown for (a) LIO-Livox, (b) FAST-LIO2, (c) LIO-DOR. (d) demonstrates the sensing box. Below the maps are the preset plane models. To better demonstrate the ghosting effects caused by indoor dynamic objects, the maps have been processed to remove the roofs.

significant factor affecting the processing time, the results of which are shown in Table VI.

TABLE VI

PROCESSING TIME [MS] UNDER DIFFERENT POINT CLOUD SCALES.

Point Num.	15000±2%	30000±2%	50000±2%	62000±2%
LIO-DOR	19.244	24.716	58.593	86.656

We also recorded stress sequence, which returns to the original position after undergoing random flips, occlusions, rapid jumps, and high-frequency oscillations, as shown in Figure 7. Due to the lacking of ground truth, the performance is qualitatively summarized in Table VII.

Fig. 8. Performance of LIO-DOR on the stress sequence.

TABLE VII

THE PERFORMANCE ON THE STRESS SEQUENCE.

	LIO-Livox	FAST-LIO2	LIO-DOR
Stress	Fail	Success/some drift	Success/minor drift

V. CONCLUSIONS

This paper proposes LIO-DOR which employs a dynamic object removal module to address the challenges of localization and mapping in dynamic environments. The proposed method does not rely on any pre-trained data and thus is not limited by the category or the number of dynamic objects. Furthermore, unlike other methods that remove dynamic points on a point-by-point basis, LIO-DOR utilizes a clustering strategy to ensure the complete removal of dynamic objects. Therefore, LIO-DOR can be robustly applied in various scenarios. For future work, we will further enhance LIO-DOR to address situations where IMU noise is significant enough to impact the projection. Additionally, we will employ real-time point cloud completion based on deep learning to mitigate the errors caused by volumetric changes of the static objects obscured by dynamic objects.

REFERENCES

[1] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." in *Robotics: Science and systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.

[2] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2020, pp. 5135–5142.

[3] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[4] P. Pfreundschuh, H. F. Hendrikx, V. Reijgwart, R. Dubé, R. Siegwart, and A. Cramariuc, "Dynamic object aware lidar slam based on automatic generation of training data," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 11 641–11 647.

[5] J. Schauer and A. Nüchter, "The peopleremover—removing dynamic objects from 3-d point cloud data by traversing a voxel occupancy grid," *IEEE robotics and automation letters*, vol. 3, no. 3, pp. 1679–1686, 2018.

[6] H. Lim, S. Hwang, and H. Myung, "Erasor: Egocentric ratio of pseudo occupancy-based dynamic object removal for static 3d point cloud map building," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2272–2279, 2021.

[7] C. Qian, Z. Xiang, Z. Wu, and H. Sun, "Rf-lio: Removal-first tightly-coupled lidar inertial odometry in high dynamic environments," *arXiv preprint arXiv:2206.09463*, 2022.

[8] G. Kim and A. Kim, "Remove, then revert: Static point cloud map construction using multiresolution range images," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 758–10 765.

[9] 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.

[10] T. Kurita, "Principal component analysis (pca)," *Computer Vision: A Reference Guide*, pp. 1–4, 2019.

[11] O. Chum and J. Matas, "Optimal randomized ransac," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 8, pp. 1472–1482, 2008.

[12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[13] S. A. Gottschalk, *Collision queries using oriented bounding boxes*. The University of North Carolina at Chapel Hill, 2000.

[14] 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.

[15] J. Yin, A. Li, T. Li, W. Yu, and D. Zou, "M2dgr: A multi-sensor and multi-scenario slam dataset for ground robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2266–2273, 2021.

Authorized licensed use limited to: University of Chinese Academy of SciencesCAS. Downloaded on October 19,2024 at 03:58:38 UTC from IEEE Xplore. Restrictions apply.