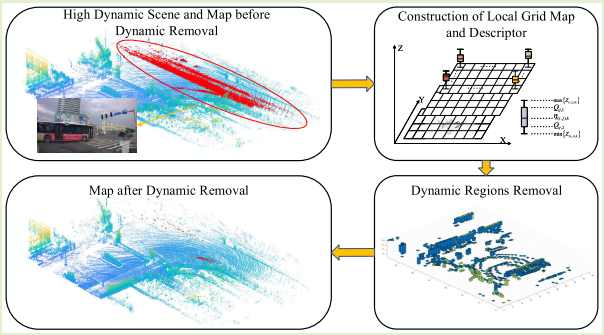
DRR-LIO: A Dynamic-Region-Removal-Based LiDAR Inertial Odometry in Dynamic Environments

Yankun Wang, Weiran Yao, *Member, IEEE*, Bing Zhang, Jinyu Fu, *Graduate Student Member, IEEE*, Jian Yang, and Guanghui Sun, *Senior Member, IEEE*

***Abstract*—This article aims to solve the problem of ghost trail effect left by dynamic objects and improve the accuracy of localization and mapping purity. Based on the tightly coupled LiDAR inertial odometry via smoothing and mapping (LIO-SAM), a real-time dynamic region removal method is proposed to challenge the real high dynamic environment. A vertical voxel height descriptor is presented to accurately discriminate dynamic and static points. Inertial measurement unit (IMU) preintegration is used for initial pose estimation to preferentially remove dynamic objects. A weighted optimization strategy is designed to improve the accuracy of pose estimation. The proposed algorithms are tested on the self-collected dataset and the public UrbanLoco dataset, and they achieve good real-time performance, mitigating the effect of dynamic objects in various scenes. The results verify that the LiDAR-inertial-based dynamic region removal odometry (DRR-LIO) can well remove dynamic objects and improve localization accuracy.**



***Index Terms*— Dynamic region removal, real-time, vertical voxel height descriptor, weighted optimization.**

I. Introduction

LIGHT detection and ranging (LiDAR)-based simultaneous localization and mapping (SLAM) technology has become widely used in a variety of industries recently, including logistics for warehouses, field rescue operations, park distribution, and services for retail centers. Current projects generally estimate the localization of robot based on matching the current frame with the 3-D features extracted from the previous frame, and the global map is composed of the extracted feature points after screening. These feature-based SLAM algorithms are generally based on static environments, and the feature points are assumed to be fixed in space. In the real environment, dynamic geometric information will be extracted from the numerous dynamic objects, as shown in Fig. 1. High uncertainty of the dynamic geometric features will affect the accuracy of pose estimation. The dynamic elements added to the map will bring unnecessary trajectories and reduce the performance of localization and navigation.

At present, most of the LiDAR SLAM systems are LiDAR odometry and mapping (LOAM) [1] scheme or its variants [2], [3]. This scheme builds a system by extracting features from the scans and creating odometry via frame-to-frame or frame-to-map matching. The establishment of the system is based on the extraction of corner and surface features. Most of them do not consider the filtering of dynamic features [4], which affect the mapping purity and the localization accuracy. These algorithms can be divided into online and offline methods.

The offline methods require accurate pose input and can only remove the dynamic points from the global map to the greatest extent [5], but cannot affect the accuracy of the pose estimation output. The online methods can give real-time localization and dynamic removal to obtain more accurate pose estimation and static map output [6], [7], [8]. But how to perform the dynamic object removal followed by pose matching is a problem.

Manuscript received 27 March 2023; accepted 17 April 2023. Date of publication 28 April 2023; date of current version 14 June 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62106062, Grant 62033005, and Grant 62103386; in part by the Heilongjiang Provincial Natural Science Foundation of China under Grant YQ2021F010; and in part by the Fundamental Research Funds for the Central Universities under Grant FRFCU5710050922. The associate editor coordinating the review of this article and approving it for publication was Prof. Yu-Dong Zhang. *(Corresponding author: Weiran Yao.)*

Yankun Wang, Weiran Yao, Bing Zhang, Jinyu Fu, and Guanghui Sun are with the School of Astronautics, Harbin Institute of Technology, Harbin 150001, China (e-mail: 20B904057@stu.hit.edu.cn; yaoweiran@hit.edu.cn; bz154964@gmail.com; 18642861102@sina.cn; guanghuisun@hit.edu.cn).

Jian Yang is with Beijing Spacecrafts, China Academy of Space Technology, Beijing 100094, China (e-mail: yj529698@163.com).

Digital Object Identifier 10.1109/JSEN.2023.3269861

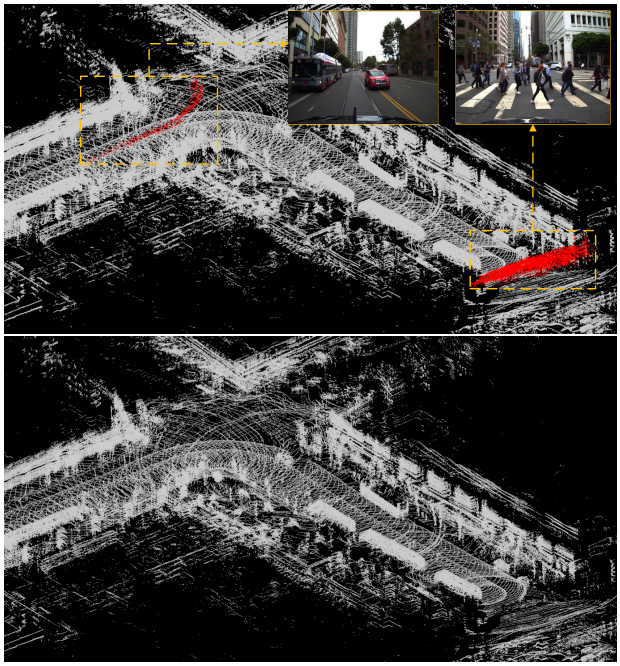


Fig. 1. Point cloud map before and after dynamic removal on the UrbanLoco dataset. The red points in the yellow boxes denote the pictures of pedestrians and cars.

It has been confirmed that the removal-first tightly coupled LiDAR inertial odometry (RF-LIO) [9] improves the localization accuracy by preferential removing the dynamic points. Similar to the RF-LIO, this article also adopts a prior dynamic objects' removal. To realize the dynamic region removal, this article proposed a real-time tightly coupled LiDAR-inertial odometry framework derived from LIO-SAM [2]. Different from LIO-SAM, we design a voxel occupancy descriptor to remove the dynamic points before pose matching, use the inertial measurement unit (IMU) information instead of the LiDAR odometry to provide a priori pose, and then adopt a weighted optimization strategy to get optimal pose estimation. This can achieve purer map and higher localization accuracy. To summarize, the primary contributions are as follows.

1) A real-time dynamic objects' removal framework is proposed to get more accurate localization service and purer map.
2) To discriminate dynamic and static regions, a vertical voxel occupancy descriptor is designed to describe the occupancy of dynamic objects at multidimensional hierarchy.
3) A weighted point-to-line and point-to-plane optimization strategy is proposed to improve the optimization accuracy.
4) The self-collected dataset[1] has been released including multisensors. The dataset can well show that the high dynamic urban environments can be used for relevant experimental verification.

The rest of the article is organized as follows: In Section II, the main related works are summarized, and the proposed methodology is described at length in Section III. The experimental results are shown in Section IV. Finally, the article ends with a conclusion in Section V.

[1]The dataset is provided in https://github.com/victoryogo/dataset.git

II. Related Works

According to the different processing methods of dynamic objects, the algorithms are usually divided into two categories. The first is to remove the point cloud of dynamic objects of the map by postprocessing. In the process, the information of all LiDAR frames are considered, and the dynamic points can be filtered more comprehensively, but this requires accurate pose input. The second is to filter the dynamic points online during the SLAM process. To ensure real-time performance, it is necessary to use the point cloud of several adjacent frames (temporally or spatially adjacent) or use the current frame to compare with the submap to detect and remove the feature points of dynamic objects. Finally, more accurate pose estimation and static map can be obtained. This article mainly focuses on the online approaches, which can be further subdivided into the following three methods.

A. Segmentation-Based Methods

This type of method is usually based on clustering. Yoon et al. [6] proposed a dynamic clustering scheme based on region growth filtering. One frame is taken as the reference frame before and after the query scan. The former reference frame is used to compare the point-to-point distance, and the point with too large distance is identified as a potential dynamic point. The potential dynamic point is placed in the latter reference frame for verification; if the point is passed by the laser beam of the frame, it will be confirmed as a dynamic point. The confirmed dynamic points are used as seed points for cluster growth in the query scan to obtain dynamic clusters. Besides, Pfreundschuh et al. [7] constructed a real-time unsupervised neural network to segment dynamic objects using clustering to achieve automatic labeling of arbitrary dynamic objects.

For point cloud segmentation of dynamic objects, there are also many methods using deep learning [8], [10], [11], [12]. Suma++ [8] used fully convolutional neural networks to effectively extract semantic information, filter out dynamic objects, and build static maps. RangeNet++ [10] infers the complete semantic segmentation of LiDAR point clouds accurately and quickly by constructing a 2-D range image from a spherical projection of the input point cloud, and then reconstructing the original points using semantics. However, these methods currently rely heavily on supervised labels and are susceptible to human error or unknown categories [13], and these methods still cannot achieve dynamic objects' removal before pose matching.

B. Visibility-Based Methods

The visibility-based methods [9], [14], [15], [16], [17] identify dynamic laser points by checking whether the laser points are occluded on the optical path. In combination with occupancy grids, Xiao et al. [15] proposed a local cylindrical reference frame for interpolating occupancy between rays to

solve irregular point densities and occlusions. Qian et al. [9] essentially used this method to distinguish dynamic points through visibility by comparing the range images of query scan and submap. However, this method is more sensitive to pose accuracy and image resolution. In addition, when the incident angle is close to 90 degrees, the old point will be severely occluded by the new point, which will cause much false deletion. Considering the angle error, ranging error, light spot effect, etc., of LiDAR frame, the false deletion will be more serious. Kim and Kim [14] proposed some tricks to solve the occlusion problem. However, some large moving objects will block the LiDAR beams, and the static points behind the dynamic points will not be observed, which make it difficult to remove the dynamic points [19].

C. Voxel-Based Methods

The voxel-based methods mainly maintain the map in a probabilistic way [18], [19], [20], [21], [22], [23], [24], [25], [26]. This method requires preserving a huge voxel map, which will consume a lot of memory and computing resources. Hornung et al. [22] proposed some postprocessing tricks to solve these problems. Dewan et al. [23] used object detection and a new voxel traversal method to speed up the process of building an occupancy map, enabling dynamic objects removal to run in real-time. However, the premise of using these methods is to have very accurate localization information; paradoxically, before removing dynamic objects, we cannot get more accurate pose information of the robot.

III. Methodology

A. System Overview

The proposed framework deviates from the above methods and is a real-time model-free change-detection-based method. The framework of our system is shown in Fig. 2, which introduces all the modules included in the system and the relationship between the modules. The input data are the IMU and LiDAR frames. First, the IMU is preintegrated to obtain the initial pose output, which provides the initial pose estimation for the LiDAR motion compensation and coordinate transformation of the current frame of the process module. The preprocessing module includes feature extraction and segmentation of the LiDAR frame and other preprocessing before the dynamic removal module. Then dynamic removal and matching is carried out in the dynamic removal module. The scan and submap that have removed the dynamic points are optimized for point-to-line and point-to-plane matching to obtain accurate pose estimation, which is fed back to IMU odometry for joint optimization to update the current frame pose.

In the preprocessing module, the LiDAR frames and submaps are segmented in each local grid map and their covariance of each grid is calculated. Unlike the global grid map in the previous voxel-based methods, in our method, the local grid map is built locally. When the dynamic removal is completed, the local grid map will be cleared to release abundant storage resources and reduce the amount of computation. In the dynamic removal module, a vertical voxel occupancy descriptor is used to judge the dynamic occupancy. The dynamic objects' removal process is as follows: 1) calculate the voxel occupancy descriptors of the current LiDAR frame and the submap, which is composed of the nearby keyframes; 2) calculate the difference between their voxel occupancy descriptors and select the occupancy grids which are larger than a given threshold as the dynamic region; and 3) remove the dynamic region to obtain the static region of the current LiDAR frame. Considering the different geometric distributions of each dynamic or static feature point in the local grids, a weighted point-to-line and point-to-plane optimization strategy is used to get a more accurate pose estimation.

B. Problem Description

The problem is how to separate the dynamic region of LiDAR scan and get an accurate pose estimation along with the final global static map.

The LiDAR scans of the original point cloud are denoted as $S_1, S_2, \ldots, S_n$, where n is the sequence number of the LiDAR frame. In each LiDAR scan, there are a set of point clouds. Let ${}^{W}_{S}T_k$ be SE(3) pose transformation from the LiDAR (base_link) coordinate to the world (map) coordinate associated with S_k; among this, $k = 1, 2, \ldots, n$. The LiDAR scan can be transformed to the map coordinate by (1); before that, the point cloud distortion of every frame should be corrected. In this article, the initial pose estimation is provided by IMU preintegration, and LiDAR scan matching provides accurate pose

$$S_k^W = {}^{W}_{S}T_k \cdot S_k, \quad (k = 1, 2, \ldots, n). \tag{1}$$

Then we project S_k^W and build a local grid map (denoted by S_k^G) with a length of h and a width of w. The grid map S_k^G is on the world coordinate system.

We build the LiDAR keyframes near the frame S_k^G into a submap B_k by

$$B_k = \left\{ S_k^G \,|k \in |T| \right\} \tag{2}$$

where $|T|$ is a frame interval near frame k. We determine the dynamic points of LiDAR scan S_k^W by the discrepancies between S_k^G and B_k^G. In S_k^G, we define the dynamic regions as ${}^{\text{dyn}}S_k^G$ and define the static regions as ${}^{\text{sta}}S_k^G$.

Every grid consists of the feature points that are not known to be dynamic or static in frame S_k^W; by separating dynamic and static grids, we can get dynamic and static points through the corresponding relationship of the grid

$${}^{\text{dyn}}S_k^W = \left\{ \bigcup P_{(i,j),k} \;\middle|\; (i,j) \in {}^{\text{dyn}}S_k^G \right\} \tag{3}$$

$${}^{\text{sta}}S_k^W = \left\{ \bigcup P_{(i,j),k} \;\middle|\; (i,j) \in {}^{\text{sta}}S_k^G \right\} \tag{4}$$

where $P_{(i,j),k}$ are the points of the Kth LiDAR key frame in the (i, j)th grid. Therefore, we convert the removal of individual dynamic points into dynamic region removal. The static global map GM_{sta} is constituted by the LiDAR scans

$$GM_{\text{sta}} = \bigcup_{k=1,2,\ldots,n} {}^{\text{sta}}S_k^W \tag{5}$$

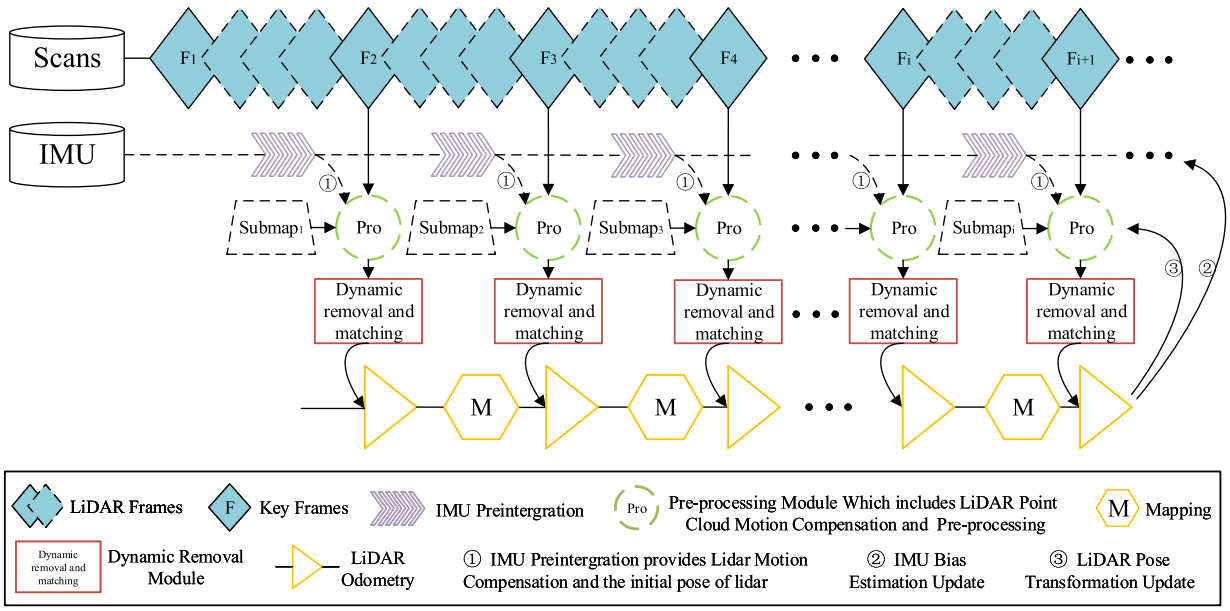


Fig. 2. Framework structure of DRR-LIO. The input data are LiDAR scans and IMU. The framework structure consists of four modules: IMU preintegration module (purple), preprocessing module (green), dynamic removal module (red), and mapping module including odometry estimation and mapping (yellow).

where n is the total frame of LiDAR scans. We transform the construction of the static global map into the dynamic removal of every grid. In Section III-D, we will present the dynamic region removal strategies to obtain GM_{sta}.

To obtain ${}^{W}_{S}T_k$, we perform a point-to-edge and point-to-plane matching on the submap and the current LiDAR frame whose dynamic points have been removed.

C. Construction of Local Grid Map and Descriptor

First, IMU preintegration is used to correct the distortion of the new LiDAR frame to eliminate the impact of motion. The features of frame are extracted and divided into corner features and surface features. The features are converted into the world coordinate system using the initial pose provided by the IMU preintegration. The LiDAR frame in the world coordinate S_k^W can be constituted with

$$S_k^W = {}^{\text{cor}}S_k^W \bigcup {}^{\text{surf}}S_k^W. \tag{6}$$

Different from the octomap, the local grid map is built locally, and the purpose is to divide the feature points into small grids to construct the descriptor. So we build the grid map of S_k^W preliminarily. We use the robot body coordinate system as the center of grid map and transform the center to the map coordinate because the map coordinate system remains constant throughout the mapping process. The pose transformation between the map and the body coordinate system is ${}^{W}_{S}T_k$. The center of the grid map is

$$C_k^W = {}^{W}_{S}T_k \cdot C_k \tag{7}$$

where C_k is the center of the body coordinate system, and C_k^W is the center of the map coordinate system.

Take C_k^W as the central coordinate, then we build the grid map with a length of h and a width of w, and the size of each

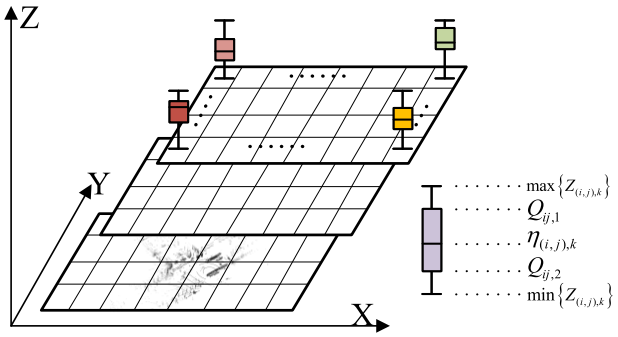


Fig. 3. Occupancy descriptor of every grid in a local grid map.

grid is calculated according to the selected scene size

$$\varepsilon = \frac{\sqrt{h^2 + w^2} \cdot \alpha}{\sqrt{2}} \tag{8}$$

where α is the sampling factor. α can be chosen based on the size of real scenes. Then we will get the grid map S_k^G of LiDAR scan. We can traverse each grid by

$$S_k^G = \bigcup_{\substack{i=1,2,\ldots,h \\ j=1,2,\ldots,w}} S_{(i,j),k}^G. \tag{9}$$

Motivated by Wang et al. [27] and Zhou et al. [28], we will extract critical areas that are more relevant for registration to improve the robustness of localization, because not all the points are equally significant in the registration process of point clouds. The ordered distribution of geometric features represents the structural degree of this region, and it is beneficial for pose estimation to allow more structural regions to participate in the registration. To determine the extraction quantity and distribution of feature points, we compute the

covariance of feature points of each grid to indicate the degree of structure by

$$\sigma^G_{(i,j),k} = \frac{1}{\left|N^G_{(i,j),k}\right|} \sum_{n=1}^{N^G_{(i,j),k}} \left(\left\| P^n_{(i,j),k} - \bar{P}_{(i,j),k} \right\| \right) \tag{10}$$

where (i, j) is the grid of the kth frame, and $N^G_{(i,j),k}$ is the number of points in (i, j) grid.

Inspired by Lim et al. [19] and Jung et al. [29], vertical information is the key information of grid occupancy. To a certain extent, the vertical height difference describes whether a grid is occupied by obstacles. Lim et al. [19] use the vertical height difference as the occupancy descriptor, but this method is easily disturbed by noise or errors. We propose a hierarchical vertical height descriptor based on distribution description, which can better reflect the central position and dispersion range of point cloud data distribution in a grid. As Fig. 3 shows, the descriptor is similar to the boxplot; there is a line in the middle of the box, which represents the median of the data. The top and bottom of the box are the upper quartile and lower quartile of the data, respectively, which means that the box contains 50% of the data. The height of the box reflects the degree of fluctuation of the data to some extent. The upper and lower edges represent the maximum and minimum values of the set of data, respectively. Therefore, we need to calculate the upper and lower edges, the upper and lower quartiles, and the median of the box in each grid.

Sort all the feature points in a grid $S^G_{(i,j),k}$ by coordinate Z, and first calculate the height difference between the upper and lower edges

$$\Delta h_{(i,j),k} = \max\left\{Z_{(i,j),k}\right\} - \min\left\{Z_{(i,j),k}\right\} \tag{11}$$

where max and min denote the maximum and minimum of all Z coordinate in $S^G_{(i,j),k}$, respectively. Then calculate the interquartile range (IQR)

$$\Delta \mathrm{R}_{(i,j),k} = Q_{ij,1} - Q_{ij,2} \tag{12}$$

where $Q_{ij,1}$ and $Q_{ij,2}$ represent the upper quartile and the lower quartile, respectively. Finally, calculate the median $\eta_{(i,j),k}$.

To describe the point cloud data comprehensively, the point cloud of each grid is divided into L layers according to the vertical height. Then we calculate the number of point clouds in each layer of the vertical grid. We call the layer with the largest number the mode $L_{(i,j),k}$. Then the occupancy descriptor is

$$\Omega_{(i,j),k} = \left\{\Delta h_{(i,j),k}, \Delta \mathrm{R}_{(i,j),k}, \eta_{(i,j),k}, L_{(i,j),k}\right\}. \tag{13}$$

The hierarchical occupancy descriptor can take full advantage of the vertical spatial information of the LiDAR point cloud.

D. Dynamic Removal

Our method is similar in concept to the existing method [18]. The difference is that we use local grids instead of global grids to reduce storage pressure and improve real-time performance, and we use the occupancy descriptors instead of ray casting to perform dynamic point recognition. Based on the process in Section III-C, we can get the grid occupancy descriptor of the LiDAR scan S^G_k and the submap B^G_k. Subtract the two descriptors, and we will obtain

$$\Omega^{\mathrm{diff}}_{(i,j),k} = \Omega^S_{(i,j),k} - \Omega^B_{(i,j),k}. \tag{14}$$

If $\Omega^{\mathrm{diff}}_{(i,j),k}$ exceeds the given threshold τ, we consider the grid to be the dynamic grids $S^G_{(i,j),k} \in {}^{\mathrm{dyn}}S^G_k$, and the dynamic map point is defined as

$${}^{\mathrm{dyn}}S^G_k = \left\{ \bigcup S^G_{(i,j),k} \,\middle|\, \Omega^{\mathrm{diff}}_{(i,j),k} > \tau \right\} \tag{15}$$

where τ is the prescribed vector threshold that we set up according to different dynamic scenes.

Algorithm 1 illustrates the process of dynamic removal.

Algorithm 1 Dynamic Removal

Initialization: LiDAR keyframe S_k, ${}^W_S T_k$
Procedure:
1: Initialize parameters τ and b.
2: $S^W_k \leftarrow {}^W_S T_k \cdot S_k$.
3: Grid projection S^G_k and B^G_k.
4: **for** each $i, j \in S^G_k, B^G_k$ **do**
5: Compute $\Omega^S_{(i,j),k}$.
6: Compute $\Omega^B_{(i,j),k}$.
7: $\Omega^{diff}_{(i,j),k} \leftarrow \Omega^S_{(i,j),k} - \Omega^B_{(i,j),k}$.
8: **end for**
9: **if** $\Omega^{diff}_{(i,j),k} > \tau$ **then**
10: ${}^{dyn}S^G_k \leftarrow \bigcup S^G_{(i,j),k}$.
11: **end if**
12: ${}^{sta}S^G_k \leftarrow S^G_k - {}^{dyn}S^G_k$.
13: Output ${}^{dyn}S^G_k$ and ${}^{sta}S^G_k$.

Fig. 4 shows the occupancy descriptors we proposed for each grid in the real environment. The region "a" shows the occupancy of the pedestrian point cloud in the current frame; it can be seen that there is a big difference between the occupancy of current frame and submap; we can easily calculate the position of the dynamic grids ${}^{\mathrm{dyn}}S^G_k$. Region "b" is the ground points. Due to the limitation of the layer height $L_{(i,j),k}$, the ground points will not be regarded as the dynamic points by mistake. For the occupancy of static objects, region "c" shows that there is almost no difference between the descriptor of the current frame and the descriptor of the submap.

The dynamic region is composed of the dynamic grids ${}^{\mathrm{dyn}}S^G_k$. The static region is the complementary set of the dynamic region in one LiDAR frame

$${}^{\mathrm{sta}}S^G_k = S^G_k - {}^{\mathrm{dyn}}S^G_k. \tag{16}$$

Gathering the dynamic points and static points in the region, we obtain ${}^{\mathrm{dyn}}S^W_k$ and ${}^{\mathrm{sta}}S^W_k$. Then we can get the final static map GM_{sta} based on (5).

E. Weighted Pose Estimation

Based on the above process, we get the static point cloud ${}^{\mathrm{sta}}S^W_k$ and the dynamic point cloud ${}^{\mathrm{dyn}}S^W_k$. The dynamic points

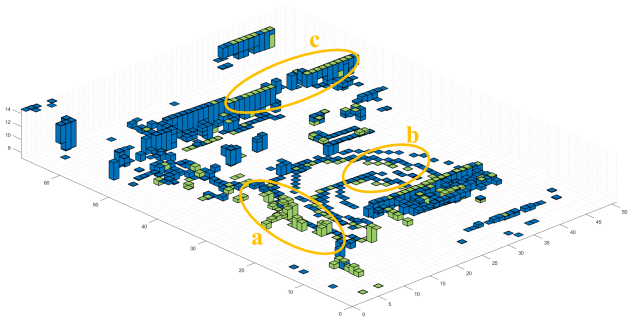


Fig. 4. Actual occupancy of the descriptor in each grid (the blue and green boxes are the occupancy of the descriptors of the submap and the current scan, respectively).

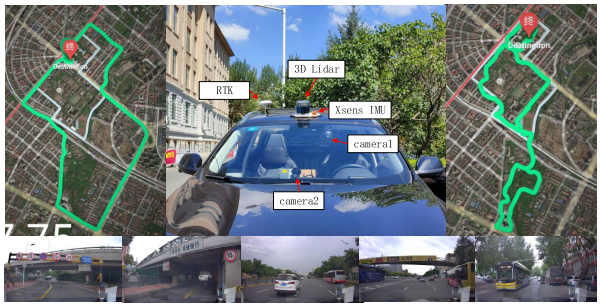


Fig. 5. Construction of the platform and description of the collected dataset.

TABLE I
SOME DETAILED DATA OF DATASET

Dataset	Scans	Total distance	Max speed	Average speed
Campus-HT	15575	8.35 km	23.2 km/h	19.3 km/h
Urban-HR	11187	7.86 km	42.5 km/h	25.2 km/h
CA-RussianHill	15860	3.20 km	36.9 km/h	17.9 km/h
CA-MarketStreet	14471	5.90 km	48.4 km/h	25.5 km/h

${}^{\mathrm{dyn}}S_k^W$ have been removed, so the matching of ${}^{\mathrm{sta}}S_k^W$ and ${}^{\mathrm{sta}}S_{k+1}^W$ between the two frames is performed. The matching method has been proved in [1] and [3] and has good robustness in outdoor scenes. The IMU odometry first provides the initial estimation for the pose estimation between two frames. Due to the uncertainty of IMU accelerometer integration, we only use rotation in the initial estimation.

We divide the static LiDAR scan ${}^{\mathrm{sta}}S_k^W$ into two parts: surface points ${}^{\mathrm{sta}}_{\mathrm{surf}}S_k^W$ and corner points ${}^{\mathrm{sta}}_{\mathrm{cor}}S_k^W$ in static points ${}^{\mathrm{sta}}S_k^W$.

Considering the local distribution of feature points, the weight is defined by

$$W_p = W_{(i,j),k}^{\mathrm{sta,surf}} = \frac{\exp\left(-\sigma_{(i,j),k}^{\mathrm{sta,surf}}\right)}{\sum_{n=1}^{M}\exp\left(-\sigma_k^n\right)} \tag{17}$$

$$W_e = W_{(i,j),k}^{\mathrm{sta,cor}} = \frac{\exp\left(-\sigma_{(i,j),k}^{\mathrm{sta,cor}}\right)}{\sum_{n=1}^{N}\exp\left(-\sigma_k^n\right)} \tag{18}$$

where M and N, respectively, represent the number of the grids of surface points and corner points. σ represents the covariance of feature points in the kth LiDAR frame, and (i, j) denotes the (i, j)th grid according to (10).

Using nearest neighbor search, we can get the associated feature points m of $\{{}^{\mathrm{sta}}_{\mathrm{surf}}S_k^W, {}^{\mathrm{sta}}_{\mathrm{cor}}S_k^W\}$ in $\{{}^{\mathrm{sta}}_{\mathrm{surf}}S_{k+1}^W, {}^{\mathrm{sta}}_{\mathrm{cor}}S_{k+1}^W\}$. The distance calculation formula is described in [1]. The transformation is estimated by minimizing the weighted sum of the point-to-planar distance d^p and the point-to-edge distance d^e

$$\min_{\Delta T_{k,k+1}}\left(\sum W_p d^p + \sum W_e d^e\right) \tag{19}$$

where $k+1$ is the $k+1$th frame. d^p and d^e are

$$\begin{aligned} d^p &= d^p_{\mathrm{sta},k+1} \\ d^e &= d^e_{\mathrm{sta},k+1}. \end{aligned} \tag{20}$$

The Jacobian matrices of plane and edge residual are

$$\begin{cases} J_p^{\mathrm{sta}} = W_{(i,j)}^{\mathrm{sta,surf}}\dfrac{\partial d_p^{\mathrm{sta}}}{\partial Tm}\dfrac{\partial Tm}{\partial \delta\xi} = W_{(i,j)}^{\mathrm{sta,surf}} n_e^B \cdot J_m \\ J_e^{\mathrm{sta}} = W_{(i,j)}^{\mathrm{sta,cor}}\dfrac{\partial d_e^{\mathrm{sta}}}{\partial Tm}\dfrac{\partial Tm}{\partial \delta\xi} = W_{(i,j)}^{\mathrm{sta,cor}} P_n \cdot \left(n_e^B \times J_m\right) \end{cases} \tag{21}$$

where P_n is the unit vector of the point-to-edge distance. n_e^B is the main direction of the covariance matrix. $W_{(i,j)}^{\mathrm{sta,surf}}$ and $W_{(i,j)}^{\mathrm{sta,cor}}$, respectively, represents the weight of the grid (i, j) in surface feature points and corner feature points. J_m is the Jacobian using the left perturbation model with $\delta\xi \in \mathfrak{se}(3)$ [30]

$$J_m = \frac{\partial Tm}{\partial \delta\xi} = \begin{bmatrix} I_{3\times3} & -[Tm]_\times \\ 0_{0\times3} & 0_{0\times3} \end{bmatrix} \tag{22}$$

where $\mathfrak{se}(3)$ is the Lie algebra of SE(3). $[Tm]_\times$ is the skew symmetric matrix, which converts the homogeneous coordinates into 3-D coordinates.

Then we use the Gauss–Newton method to solve the nonlinear equation and find the optimal pose estimation. The LiDAR odometry can be written as

$$T_{k+1} = T_k \Delta T_{k,k+1}. \tag{23}$$

By applying iterative pose optimization, the current pose estimation can be resolved.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

To validate the performance of the proposed method in complex dynamic urban road scenes, we put our method to the test using the dataset collected from busy streets and campus. (The dataset is provided in https://github.com/victoryogo/dataset.git). We also benchmark our method with the open UrbanLoco dataset [31]. In a real environment with ground-truth annotations, we showed that the method was applicable to a multitude of different speeds of dynamic scenes, and we evaluated the dynamic objects' removal effect of the map and effect on improving the localization accuracy. Besides, we compared the performance of a standard LIO-SAM pipeline with our method. Experiments were carried out with an Intel I7-1165G7 CPU.

TABLE II
COMPARISON OF MAE AND ABSOLUTE TRAJECTORY RMSE OF RESULTING TRAJECTORIES FROM DIFFERENT APPROACHES

Dataset	Length (km)	method	MAE in Translation (m) X	Y	Z	MAE in Rotation (deg) X	Y	Z	RMSE in Trans. (m)	Trajectory Accuracy(%)
Campus-HT	8.4	LeGO-LOAM [3]	63.586	57.153	48.631	88.978	42.354	89.075	125.988	1.500
		LIO-SAM [2]	26.828	16.844	7.529	7.472	9.249	5.851	38.047	0.453
		DRR-LIO (Ours)	**9.469**	**10.718**	**3.898**	**4.492**	**5.259**	**4.874**	**16.032**	**0.191**
Urban-HR	7.9	LeGO-LOAM [3]	144.049	94.27	34.19	87.639	42.43	87.465	208.042	2.633
		LIO-SAM [2]	25.752	58.132	9.482	7.77	8.789	9.123	46.529	0.589
		DRR-LIO (Ours)	**13.818**	**12.991**	**4.479**	**5.768**	**7.771**	**8.028**	**17.951**	**0.227**
CA-RussianHill	3.2	LeGO-LOAM [3]	18.697	20.917	25.939	94.374	41.568	40.627	53.562	1.674
		LIO-SAM [2]	14.973	10.599	9.268	6.276	7.023	3.161	22.663	0.708
		RF-LIO [9]	-	-	-	-	-	-	**12.17**	**0.380**
		DRR-LIO (Ours)	**2.26**	**3.882**	**11.449**	**5.492**	**6.043**	**1.789**	14.295	0.447
CA-MarketStreet	5.9	LeGO-LOAM [3]	55.929	35.735	17.782	88.128	10.306	38.128	93.179	1.579
		LIO-SAM [2]	21.293	34.175	**2.505**	4.85	7.419	2.573	52.189	0.884
		RF-LIO [9]	-	-	-	-	-	-	15.89	0.269
		DRR-LIO (Ours)	**5.125**	**4.075**	2.849	**3.794**	**4.231**	**1.608**	**8.358**	**0.142**

A. Dataset

We collected two datasets in the campus of the university and the crowded streets in the urban area, recording more than 26 000 scans in total, with point clouds recorded at 10 Hz with 1800 points per revolution. The middle of Fig. 5 shows the data collection platform based on ROBOSENSE-32 LiDAR, XSENS MTI-30 IMU sensors, and HI-TARGET RTK system. The left and right of Fig. 5 are the driving trajectory of the platform, with a total mileage of more than 16 km. The lower part of Fig. 5 shows some images of the dataset. It can be seen that the road conditions are relatively complex, and most of the dynamic objects are pedestrians and vehicles of different sizes. In our dataset, some of the sections are far away from buildings, resulting in a lack of stable structural features; the registration of scan-to-submap will be more dependent on the structural features reflected by moving vehicles. The movement of the vehicles will seriously affect the global localization accuracy. In addition, we collected the public UrbanLoco dataset for validation, which provides ground-truth and highly dynamic streets data. Some detailed data of these datasets are shown in Table I.

B. Localization Error Comparison

We tested the dataset collected by ourselves and the public on the localization error. We used the trajectory recorded by RTK as the ground truth and contrasted our method with LeGO-LOAM and LIO-SAM which are the SOTA SLAM algorithms that do not remove dynamic objects. Table II shows the mean absolute error (MAE) of trajectory in all the directions of translation and rotation, and the absolute trajectory root mean square error (RMSE) in translation. The RMSE in translation verifies the advantages of our method in reducing the localization error. Compared with LIO-SAM, the localization accuracy of our method improved by 57.9%, 61.4%, 36.9%, and 83.9% on the four datasets, respectively. Fig. 6 shows that the translation and rotation errors of our method are relatively more gentle and smooth with the increase in path length. The improvement of the translation and rotation

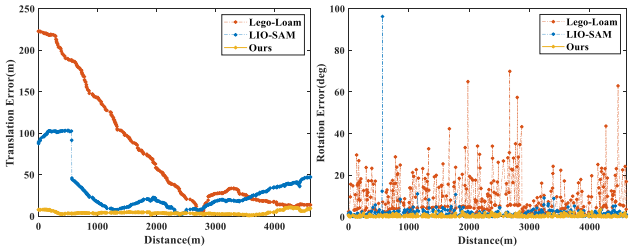


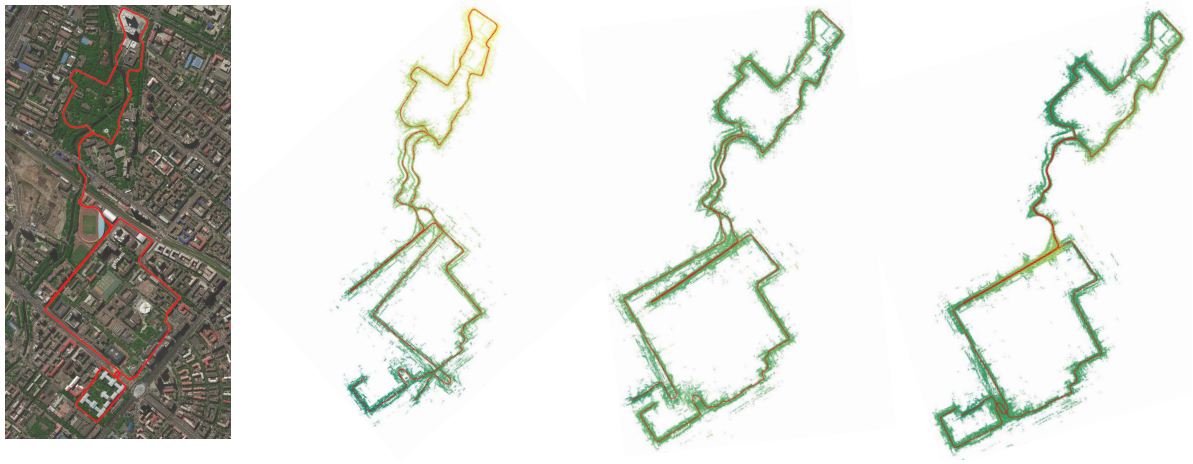
Fig. 6. Translation and rotation errors of the CA-MarketStreet dataset.

accuracy is limited in the z-axis direction because the low vertical resolution makes it difficult to handle the drift of LiDAR data in the z-axis direction. In addition, we compared our method with RF-LIO in terms of absolute trajectory localization error. This comparison is based on two public datasets used in the validation experiments of RF-LIO; since RF-LIO does not disclose its MAE in translation and rotation, we only compare the error with its RMSE in Table II. In the CA-MarketStreet dataset, the localization accuracy is 47.4% higher than RF-LIO. In the CA-RussianHill dataset, because there is a lot of sloping ground, the localization accuracy of our method is slightly lower than RF-LIO, but it does not affect its basic effectiveness.

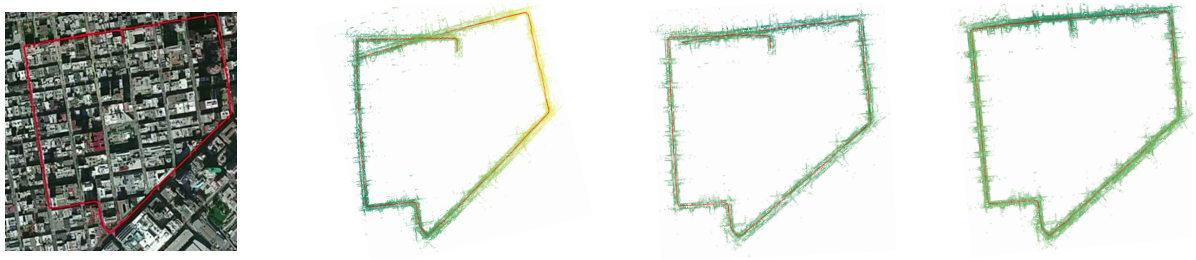
Fig. 7 shows the mapping results of our method and some classical methods. Obviously, the trajectory of LeGO-LOAM drifts the worst, and LIO-SAM is better than LeGO-LOAM. By comparison, our method is closer to ground truth and has smaller drift due to the removal of moving objects.

C. Removal Effect

To verify the effectiveness of dynamic removal, we first computed the error between the predicted center and the optimized center of the grid map based on (7). As shown in Fig. 8, the grid map of current frame was constructed with low error, guaranteeing a good accuracy of dynamic removal. Besides, we compared the dynamic objects' removal effect



(a)



(b)

Fig. 7. Comparison of the mapping results, respectively; it represents the mapping results of ground truth, LeGO-LOAM, LIO-SAM, and our method. (a) Mapping result of the Campus-HT dataset. (b) Mapping result of the CA-RussianHill dataset.

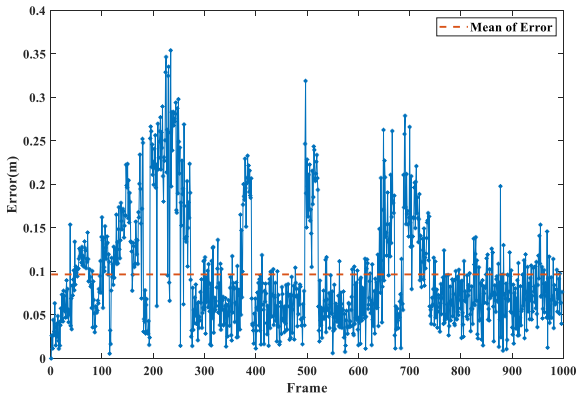


Fig. 8. Error between the predicted center and the optimized center.

of different scenes in the tested dataset and carried out a qualitative analysis.

As illustrated in Fig. 9(A) and (B), we selected a typical dynamic scene from the self-collected dataset and the UrbanLoco dataset, respectively. Fig. 9(A) is a dynamic scene that the moving objects wait first, and then chase. Fig. 9(B) is a scene of multiple vehicles driving in the same direction. The original map generated by the LIO-SAM is compared with the dynamic removal map generated by the proposed method. The map that has removed the dynamic points is purer, but some dynamic points are still not completely removed. This is because of the limited perspective angle of LiDAR, resulting in no observation of the dynamic objects, and the vertical resolution is too low to find the dynamic objects. To evaluate

TABLE III
PRESERVATION RATE AND REJECTION RATE

Dataset	PR	RR
Campus-HT	85.91%	92.18%
Urban-HR	87.23%	94.42%
CA-RussianHill	79.49%	87.59%
CA-MarketStreet	82.35%	95.25%
average	83.75%	92.36%

the quality of the dynamic objects' removal, we need to consider two aspects: one is the removal rate of the dynamic objects, denoted as RR; the other is the preservation rate of static objects, denoted as PR. This evaluation method is well described in [14]

$$\begin{cases} \mathrm{PR} = \dfrac{P_{\mathrm{ps}}}{P_{\mathrm{ts}}} \\ \mathrm{RR} = 1 - \dfrac{P_{\mathrm{pd}}}{P_{\mathrm{td}}}. \end{cases} \tag{24}$$

According to (24), the total static and dynamic points are represented by P_{ts} and P_{td} on the raw map, respectively. P_{ps} and P_{pd} indicate the preserved static points and dynamic points, respectively.

We took five dynamic scenes from every dataset, then counted the number of residual dynamic points and the number of all the dynamic points created by LIO-SAM in each scene, and finally calculated the average dynamic removal rate in each dataset. Similarly, we counted the number of all the static

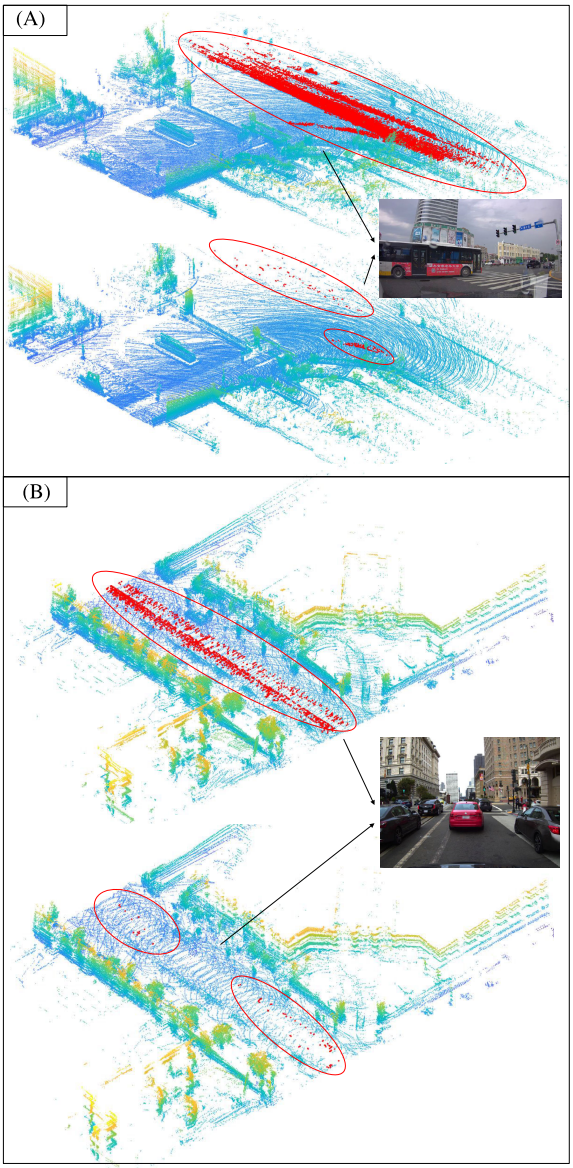


Fig. 9. Final dynamic removal effect (the map fragment in (A) is from the Urban-HR dataset, and the map fragment in (B) is from the CA-MarketStreet dataset).

map points and the number of remaining static points after dynamic removal and obtained the average static preserved rate. As given in Table III, the removal rates of all four datasets are above 80% or close to 80%, and the preserved rates are over 85%. The average removal rate was 83.75% and the average preserved rate reached 92.36%. The results show that the method has good dynamic removal performance.

D. Runtime

To verify the real-time performance, we counted the average time and the average number of points for our method to process one frame. As given in Table IV, the cost time is basically proportional to the number of points to be processed. In the experiments, the processing of one frame and the corresponding submap costs 40–90 ms. To further test the real-time performance, we used a 64-beam LiDAR

TABLE IV
RUNTIME OF THE DATASET

Dataset	Time (ms)	Average number of points in one frame
Campus-HT	79.37	18275
Urban-HR	89.10	18407
CA-RussianHill	48.08	12760
CA-MarketStreet	46.83	12365

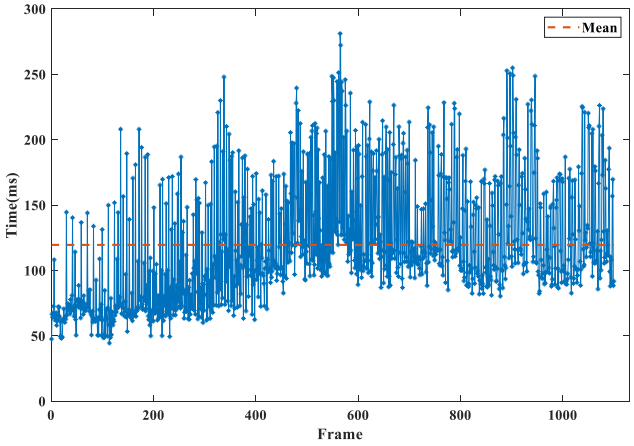


Fig. 10. Running time of each frame in the KITTI dataset with 64-beam LiDAR.

in the KITTI dataset to test the proposed method [32] (see Fig. 10), in which 45 626 points on average were processed in each frame. The average processing time of each frame is approximately 119.59 ms, verifying the real-time performance of the proposed method.

V. CONCLUSION

In this article, we proposed a real-time LiDAR-inertial-based dynamic region removal odometry (DRR-LIO). Our method effectively eliminated the ghost trail effect left by dynamic objects and improved the localization accuracy by an average of 60%. The DRR-LIO used a vertical voxel height descriptor to guarantee a good real-time performance. To verify the dynamic removal accuracy, we manually marked the dynamic point cloud and the static point cloud. The average static points' preservation rate was 83.75%, and the average dynamic points' removal rate reached 92.36%, indicating a good accuracy of dynamic objects' removal. Tests on different datasets show that DRR-LIO is robust to various scenarios. In further works, the proposed method will be improved and extended to enhance its localization accuracy and robustness under sloping pavement.

REFERENCES

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

[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 *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 5135–5142, doi: 10.1109/IROS45743.2020.9341176.

[3] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized LiDAR odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 4758–4765, doi: 10.1109/IROS.2018.8594299.

[4] S. Zhao, Z. Fang, H. Li, and S. Scherer, "A robust laser-inertial odometry and mapping method for large-scale highway environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 1285–1292, doi: 10.1109/IROS40897.2019.8967880.

[5] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, and R. Siegwart, "Long-term 3D map maintenance in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2014, pp. 3712–3719, doi: 10.1109/ICRA.2014.6907397.

[6] D. Yoon, T. Tang, and T. Barfoot, "Mapless online detection of dynamic objects in 3D LiDAR," in *Proc. 16th Conf. Comput. Robot Vis. (CRV)*, May 2019, pp. 113–120, doi: 10.1109/CRV.2019.00023.

[7] P. Pfreundschuh, H. F. C. Hendrikx, V. Reijgwart, R. Dubé, R. Siegwart, and A. Cramariuc, "Dynamic object aware LiDAR SLAM based on automatic generation of training data," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 11641–11647, doi: 10.1109/ICRA48506.2021.9560730.

[8] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss, "SuMa++: Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4530–4537, doi: 10.1109/IROS40897.2019.8967704.

[9] C. Qian, Z. Xiang, Z. Wu, and H. Sun, "RF-LIO: Removal-first tightly-coupled LiDAR inertial odometry in high dynamic environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2021, pp. 4421–4428, doi: 10.1109/IROS51168.2021.9636624.

[10] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "RangeNet++: Fast and accurate LiDAR semantic segmentation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 4213–4220, doi: 10.1109/IROS40897.2019.8967762.

[11] P. Ruchti and W. Burgard, "Mapping with dynamic-object probabilities calculated from single 3D range scans," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 6331–6336, doi: 10.1109/ICRA.2018.8463149.

[12] Z. Zhao, W. Zhang, J. Gu, J. Yang, and K. Huang, "LiDAR mapping optimization based on lightweight semantic segmentation," *IEEE Trans. Intell. Vehicles*, vol. 4, no. 3, pp. 353–362, Sep. 2019.

[13] K. Wong, S. Wang, M. Ren, M. Liang, and R. Urtasun, "Identifying unknown instances for autonomous driving," in *Proc. Conf. Robot Learn. (CoRL)*, 2019, pp. 384–393.

[14] G. Kim and A. Kim, "Remove, then revert: Static point cloud map construction using multiresolution range images," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 10758–10765, doi: 10.1109/IROS45743.2020.9340856.

[15] W. Xiao, B. Vallet, M. Brédif, and N. Paparoditis, "Street environment change detection from mobile laser scanning point clouds," *ISPRS J. Photogramm. Remote Sens.*, vol. 107, pp. 38–49, Sep. 2015.

[16] X. Li, P. Ye, J. Li, Z. Liu, L. Cao, and F.-Y. Wang, "From features engineering to scenarios engineering for trustworthy AI: I&I, C&C, and V&V," *IEEE Intell. Syst.*, vol. 37, no. 4, pp. 18–26, Jul./Aug. 2022, doi: 10.1109/MIS.2022.3197950.

[17] R. Ambrus, N. Bore, J. Folkesson, and P. Jensfelt, "Meta-rooms: Building and maintaining long term spatial models in a dynamic world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 1854–1861, doi: 10.1109/IROS.2014.6942806.

[18] J. Schauer and A. Nuchter, "The peopleremover—Removing dynamic objects from 3-D point cloud data by traversing a voxel occupancy grid," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 1679–1686, Jul. 2018.

[19] 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 Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2272–2279, Apr. 2021.

[20] J. Fu, W. Yao, G. Sun, H. Tian, and L. Wu, "An FTSA trajectory elliptical homotopy for unmanned vehicles path planning with multi-objective constraints," *IEEE Trans. Intell. Vehicles*, vol. 8, no. 3, pp. 2415–2425, Mar. 2023.

[21] J. Fu, G. Sun, W. Yao, and L. Wu, "On trajectory homotopy to explore and penetrate dynamically of multi-UAV," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24008–24019, Dec. 2022, doi: 10.1109/TITS.2022.3195521.

[22] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Robot.*, vol. 34, no. 3, pp. 189–206, 2013.

[23] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based detection and tracking in 3D LiDAR scans," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2016, pp. 10765–10771, doi: 10.1109/ICRA.2016.7487649.

[24] S. Pagad, D. Agarwal, S. Narayanan, K. Rangan, H. Kim, and G. Yalla, "Robust method for removing dynamic objects from point clouds," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 6331–6336, doi: 10.1109/ICRA40945.2020.9197168.

[25] X. Li, K. Wang, Y. Tian, L. Yan, F. Deng, and F.-Y. Wang, "The ParallelEye dataset: A large collection of virtual images for traffic vision research," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 6, pp. 2072–2084, Jun. 2019, doi: 10.1109/TITS.2018.2857566.

[26] M. Arora, L. Wiesmann, X. Chen, and C. Stachniss, "Mapping the static parts of dynamic scenes from 3D LiDAR point clouds exploiting ground segmentation," in *Proc. Eur. Conf. Mobile Robots (ECMR)*, 2021, pp. 1–6, doi: 10.1109/ECMR50962.2021.9568799.

[27] H. Wang, C. Wang, C.-L. Chen, and L. Xie, "F-LOAM: Fast LiDAR odometry and mapping," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 4390–4396, doi: 10.1109/IROS51168.2021.9636655.

[28] Z. Zhou, M. Yang, C. Wang, and B. Wang, "ROI-cloud: A key region extraction method for LiDAR odometry and localization," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 3312–3318, doi: 10.1109/ICRA40945.2020.9197059.

[29] J. Jung, C. Stachniss, and C. Kim, "Automatic room segmentation of 3D laser data using morphological processing," *ISPRS Int. J. Geo-Inf.*, vol. 6, no. 7, p. 206, 2017.

[30] T. D. Barfoot, *State Estimation for Robotics: A Matrix Lie Group Approach*. Cambridge, U.K.: Cambridge Univ. Press, 2016.

[31] W. Wen et al., "UrbanLoco: A full sensor suite dataset for mapping and localization in urban scenes," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 2310–2316, doi: 10.1109/ICRA40945.2020.9196526.

[32] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3354–3361, doi: 10.1109/CVPR.2012.6248074.

Yankun Wang received the B.S. and M.S. degrees from Northeastern University, Shenyang, China, in 2016 and 2020, respectively. He is currently pursuing the Ph.D. degree with the Department of Control Science and Engineering, Harbin Institute of Technology University, Harbin, China.

His research interests include multisensor fusion, simultaneous localization and mapping (SLAM), and mobile multisensor mapping systems.

Weiran Yao (Member, IEEE) received the bachelor's (Hons.), master's, and Ph.D. degrees in aeronautical and astronautical science and technology from the School of Astronautics, Harbin Institute of Technology (HIT), Harbin, China in 2013, 2015, and 2020, respectively.

From 2017 to 2018, he was a Visiting Ph.D. Student with the Department of Mechanical and Industrial Engineering, University of Toronto (UofT), Toronto, ON, Canada. He is currently an Associate Professor with the School of Astronautics, HIT. His research interests include unmanned vehicles, multirobot mission planning, and multiagent control systems.

Bing Zhang received the B.S. degree from North Central University, Taiyuan, China, in 2017, and the M.S. degree from the Harbin Institute of Technology, Harbin, China, in 2020, where he is currently pursuing the Ph.D. degree with the Department of Control Science and Engineering.

His research interests include multimodal feature fusion in perception and mapping, 3-D reconstruction, and simultaneous localization and mapping (SLAM).

Jinyu Fu (Graduate Student Member, IEEE) received the B.S. degree in automation from Dalian University, Dalian, China, in 2016, and the M.S. degree in navigation science and technology from Dalian Maritime University, Dalian, in 2019. He is currently pursuing the Ph.D. degree with the Department of Control Science and Engineering, School of Astronautics, Harbin Institute of Technology, Harbin, China.

His current research interests include intelligent vehicles systems and mission planning of multirobot.

Jian Yang is currently working with Beijing Spacecrafts, China Academy of Space Technology, Beijing, China. His research interests include high-precision control and processing technology of large-scale equipment.

Dr. Yang was awarded the title of Senior Technician in 2019.

Guanghui Sun (Senior Member, IEEE) received the B.S. degree in automation and the M.S. and Ph.D. degrees in control science and engineering from the Harbin Institute of Technology, Harbin, China, in 2005, 2007, and 2010, respectively.

He is currently a Professor with the Department of Control Science and Engineering, Harbin Institute of Technology. His research interests include fractional order systems, nonlinear control systems, and sliding mode control.