2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

RF-LIO: Removal-First Tightly-coupled Lidar Inertial Odometry in High Dynamic Environments

2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-1714-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IROS51168.2021.9636624

***Abstract*— Simultaneous Localization and Mapping (SLAM) is considered to be an essential capability for intelligent vehicles and mobile robots. However, most of the current lidar SLAM approaches are based on the assumption of a static environment. Hence the localization in a dynamic environment with multiple moving objects is actually unreliable. The paper proposes a dynamic SLAM framework RF-LIO, building on LIO-SAM, which adds adaptive multi-resolution range images and uses tightly-coupled lidar inertial odometry to first remove moving objects, and then match lidar scan to the submap. Thus, it can obtain accurate poses even in high dynamic environments. The proposed RF-LIO is evaluated on both self-collected datasets and open Urbanloco datasets. The experimental results in high dynamic environments demonstrate that, compared with LOAM and LIO-SAM, the absolute trajectory accuracy of the proposed RF-LIO can be improved by 90% and 70%, respectively. RF-LIO is one of the state-of-the-art SLAM systems in high dynamic environments.**

I. INTRODUCTION

Robust and accurate localization is the premise for an intelligent vehicle or mobile robot. SLAM technology based on lidar can provide robust centimeter-level state estimation and a high-precision point cloud map without GPS, hence receives much attention recently. For example, LOAM [1] proposes a feature extraction strategy based on edge and plane, which has become the most widely used method for low-drift and real-time state estimation and mapping. On the basis of LOAM, LIO-SAM [2] adopts tightly-coupled lidar inertial odometry (LIO) and keyframe strategy to further improve the processing speed and trajectory accuracy. Nevertheless, these mainstream SLAM systems and point cloud registration methods [3], [4] are based on the assumption of a static environment, i.e. there are no moving objects in the background. In the matter of fact, autonomous systems often work in a realistic environment with a lot of moving objects, such as vehicles, pedestrians, etc. Due to the sensor's field of view (FOV) is blocked, most of the feature points fall on the moving objects instead of the static map, as illustrated in Fig. 1. The previous approaches based on the static environment assumption will fail. Moreover, it is also challenging to extract road markers, traffic signs, and other critical static features in the point cloud map, as ghost tracks of moving objects may occlude them [5]. Therefore, it is essential to improve the performance of SLAM in dynamic environments.

A naive solution for dynamic SLAM is to build a map containing only static objects, i.e. to remove the moving object points from the point cloud map. For example, Suma++ [6] first obtain the accurate poses of multiple measurements by scan-matching and then iteratively update the state (i.e. static or dynamic) of a map voxel by fusing the semantic information. However, this premise is often not tenable in a high dynamic environment because a large number of moving objects will make the existing scan-matching method ineffective. Hence, these methods fall into a chicken-and-egg problem: moving object points removal relies on accurate pose, while accurate pose can not be obtained when the point cloud map contains moving objects. The above problem makes these SLAM methods can not achieve good results in high dynamic environments. In addition, deep-learning methods relies heavily on training data, which also limits the application of Suma++ from one scene to other scenes.

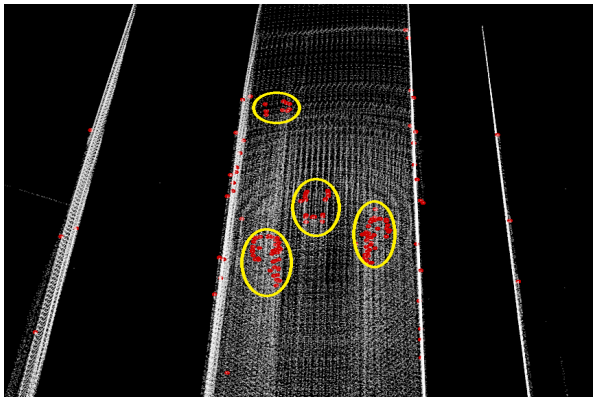


Fig. 1. Distribution of feature points under the moving objects occlusion. The white points are the static environment. The red points are the edge feature points. The red points in the yellow circle fall on the moving vehicles.

This paper proposes a new removal-first tightly-coupled lidar inertial odometry framework, i.e. RF-LIO, to solve the SLAM problem in high dynamic environments. The removal-first refers to that the proposed RF-LIO first removes moving object without an accurate pose and then employ scan-matching. When a new scan arrives, RF-LIO does not immediately perform scan-matching to obtain an accurate pose, as it is easily affected by the dynamic environment. Instead, we use the tightly-coupled inertial measurement unit (IMU) odometry to obtain a rough initial state estimation. Then, RF-LIO can preliminarily remove the moving points in the environment by using the adaptive resolution range image. After preliminary moving points removal, RF-LIO uses scan-matching to obtain a relatively more accurate pose. Through these iterative removal and scan-matching steps, RF-LIO can finally obtain accurate poses in high dynamic environments. We use the removal rate of moving objects and absolute trajectory accuracy to evaluate RF-LIO on both

self-collected datasets and open Urbanloco datasets [7]. The experiments demonstrate that the average moving objects removal rate of RF-LIO is 96.1% and the absolute trajectory accuracy of RF-LIO can be improved by 90% and 70%, compared with LOAM and LIO-SAM, respectively.

II. RELATED WORKS

Moving objects such as pedestrians and vehicles widely exist in real-world scenes. Therefore, most state-of-the-art SLAM approaches that initially designed in static environments cannot handle these severe dynamic scenes [8]. To address this problem, moving objects need to be recognized from the background and removed. The related approaches can be summarized as follows:

Statistical approaches: These approaches first calculate a geometric model, and then use a statistical method, such as Random Sample Consensus (RANSAC) [9], to remove the feature points that do not conform to the geometric model. However, these approaches will fail if the moving features are in the majority.

Voxel-based approaches: These methods are based on voxel ray casting [10]–[12]. Voxel-based approaches construct a huge voxel map and use the simple prior knowledge: when a lidar beam hits a voxel, the voxels along the way must be empty. But the premise of using these methods is to have very accurate localization information, which is contrary to the original intention of dynamic SLAM. In addition, these methods consume a lot of memory and computing resources. Even if the latest method [5] uses deep-learning and GPU acceleration, it can only maintain a maximum octree depth of 16 and voxel size of 0.3 meters.

Visibility-based approaches: These visibility-based approaches [13], [14] do not need to maintain a vast voxel map. Compared with voxel-based approaches, these methods can build a larger map with higher computational efficiency. These methods are based on the principle that when there is a closer point in a narrow FOV, the farther point will be occluded. On the contrary, if we observe the map's far point, then the corresponding closer point in the query scan must be moving.

Segmentation-based approaches: DynaSLAM [15] use semantic information to assist moving objects detection. These methods often need to be combined with other traditional methods. Because even the object is identified as a vehicle, we can not judge whether it is stopping or driving. For example, DS-SLAM [8] uses a statistical method to calculate the motion consistency of feature points to select moving points and fixed points. The proportion of moving points in the same object is used to judge whether the object is moving or stationary. Moreover, segmentation-based approaches rely heavily on supervised labels and training data and are limited by the accuracy and category of segmentation.

III. REMOVAL-FIRST LIDAR INERTIAL ODOMETRY

A. System Overview

Fig. 2 shows an overall framework of RF-LIO, which consists of three major modules: IMU preintegration, feature extraction, and mapping. First of all, the IMU preintegration module is used to infer the system motion and generates IMU odometry. Then, the feature extraction module compensates the motion distortion of the point cloud [16]. The edge and plane features are extracted by evaluating the roughness of points.

The mapping module is the critical module of our proposed approach. To achieve the moving objects removal-first without an accurate pose, there are several key steps: (i) The initial pose is obtained by IMU odometry. Then the error between IMU preintegration and scan-matching is used to determine an initial resolution (i.e. how many angles of FOV does each pixel correspond to). (ii) RF-LIO use this initial resolution to construct the range image from the current lidar scan and the corresponding submap separately. (iii) Through comparing their visibility, the main moving points of the submap are removed. (iv) RF-LIO match the lidar scan to the submap and judge whether scan-matching is convergent. If it is convergent, after graph optimization, the final fine resolution is used to remove the remaining moving points in the current keyframe. otherwise, a new resolution will be generated, and steps (ii), (iii), and (iv) will be repeated.

B. IMU Preintegration and Initial Pose

We first denote the world frame as W and the IMU frame, which coincides with the robot body frame as B. The state of the system can be represented as follows:

$$x = \left[R^T, p^T, v^T, b^T\right]^T \tag{1}$$

where $R \in \mathrm{SO}(3)$ is the rotation matrix, $p \in \mathbb{R}^3$ is the position vector, v is the speed vector, and b is the IMU bias vector. b consists of slowly varying accelerometer bias b^{a} and gyroscope bias b^{g}. The IMU measurement model can be expressed as follows:

$$\hat{\omega}_{\mathrm{B}}(t) = \omega_{\mathrm{B}}(t) + b^{\mathrm{g}}(t) + \eta^{\mathrm{g}}(t) \tag{2}$$

$$\hat{a}_{\mathrm{B}}(t) = R^{\mathrm{T}}_{\mathrm{WB}}(t)\left(a_{\mathrm{B}}(t) - g_{\mathrm{W}}\right) + b^{\mathrm{a}}(t) + \eta^{\mathrm{a}}(t) \tag{3}$$

Where $\hat{\omega}_{\mathrm{B}}$ and $\hat{a}_{\mathrm{B}}$ are the raw IMU measurements in B and are affected by the bias b and white noise η.

When IMU measurements come, we apply the IMU preintegration method proposed in [17] to obtain the initial pose of the current keyframe $k+1$ from the previous keyframe k:

$$R^{k+1}_{\mathrm{WB}} = R^{k}_{\mathrm{WB}} \Delta R_{k,k+1} \operatorname{Exp}\left(\left(J^{\mathrm{g}}_{\Delta R} b^{\mathrm{g}}_{k}\right)\right) \tag{4}$$

$$\begin{aligned} v^{k+1}_{\mathrm{B}} &= v^{k}_{\mathrm{B}} + g_{\mathrm{W}} \Delta t_{k,k+1} \\ &\quad + R^{k}_{\mathrm{WB}}\left(\Delta v^{k,k+1}_{\mathrm{B}} + J^{\mathrm{g}}_{\Delta v} b^{\mathrm{g}}_{k} + J^{\mathrm{a}}_{\Delta v} b^{\mathrm{a}}_{k}\right) \end{aligned} \tag{5}$$

$$\begin{aligned} p^{k+1}_{\mathrm{B}} &= p^{k}_{\mathrm{B}} + v^{k}_{\mathrm{B}} \Delta t_{k,k+1} + \frac{1}{2} g_{\mathrm{W}} \Delta t^2_{k,k+1} \\ &\quad + R^{k}_{\mathrm{WB}}\left(\Delta p_{k,k+1} + J^{g}_{\Delta p} b^{\mathrm{g}}_{k} + J^{a}_{\Delta p} b^{\mathrm{a}}_{k}\right) \end{aligned} \tag{6}$$

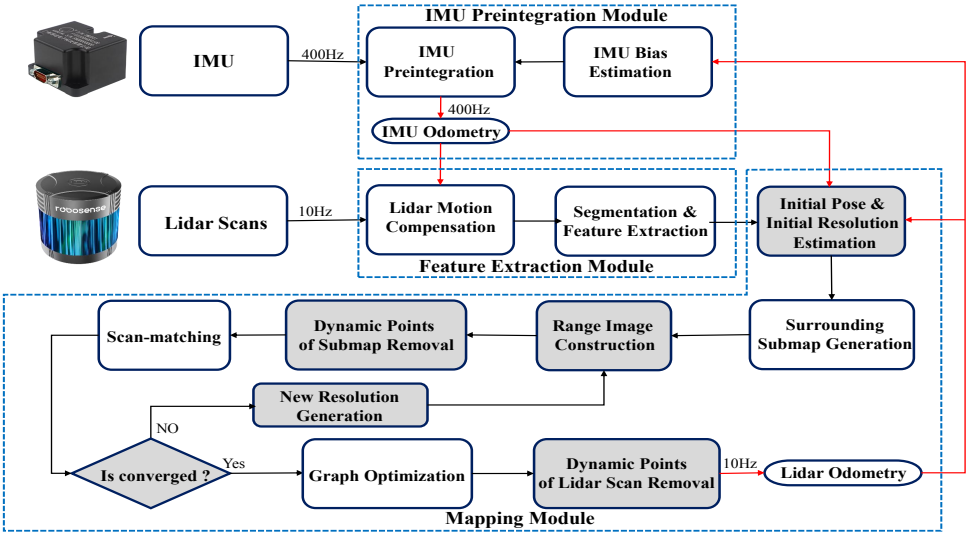


Fig. 2. Overall framework of RF-LIO. The IMU preintegration module is used to infer the system motion and generate IMU odometry. The feature extraction module compensates the motion distortion of point cloud and extracts feature points by evaluating the roughness of points. The mapping module outputs a refined pose estimation and a global 3D mapping, and removes the moving objects from the point cloud map.

Where the Jacobians $J^{a}_{(\cdot)}b^{\mathrm{a}}$ and $J^{g}_{(\cdot)}b^{\mathrm{g}}$ represent a first-order approximation of the effect of changing the biases without explicitly recomputing the preintegration.

C. IMU Preintegration Error and Initial Resolution

In the process of using IMU measurements to infer system motion, there will inevitably be a deviation from the ground truth, which makes the query scan points to the corresponding map points are ambiguous. To solve this problem, Palazzolo and Stachniss [18] proposed a window-based method (i.e. not pixel-to-pixel, but pixel-to-window comparison). A more convenient method was proposed in Removert [19] that uses multiple range images having different resolutions. However, Removert uses fixed resolutions, as it is based on accurate localization information. But RF-LIO needs to remove the dynamic points before scan-matching (i.e. without an accurate pose). Hence we use the pose error between IMU preintegration and scan-matching to generate the initial resolution dynamically.

When scan-matching is performed, the translation and orientation errors of IMU preintegration can be computed as follows:

$$E^{k}_{\mathrm{R}} = \mathrm{Log}\left(\left(\Delta R_{k-1,k}\,\mathrm{Exp}\left(J^{\mathrm{g}}_{\Delta R}b^{\mathrm{g}}_{k-1}\right)\right)^{T} R^{k-1}_{\mathrm{BW}}R^{k}_{\mathrm{WB}}\right) \tag{7}$$

$$E^{k}_{\mathrm{v}} = R^{k-1}_{\mathrm{BW}}\left(v^{k}_{\mathrm{B}} - v^{k-1}_{\mathrm{B}} - g_{\mathrm{W}}\Delta t_{k-1,k}\right) - \left(\Delta v_{k-1,k} + J^{\mathrm{g}}_{\Delta v}b^{\mathrm{g}}_{k-1} + J^{\mathrm{a}}_{\Delta v}b^{\mathrm{a}}_{k-1}\right) \tag{8}$$

$$E^{k}_{\mathrm{p}} = R^{k-1}_{\mathrm{BW}}\left(p^{k}_{\mathrm{B}} - p^{k-1}_{\mathrm{B}} - v^{k-1}_{\mathrm{B}}\Delta t_{k-1,k} - \frac{1}{2}g_{\mathrm{W}}\Delta t^{2}_{k-1,k}\right) - \left(\Delta p_{k-1,k} + J^{\mathrm{g}}_{\Delta p}b^{\mathrm{g}}_{k-1} + J^{a}_{\Delta p}b^{\mathrm{a}}_{k-1}\right) \tag{9}$$

$$E^{k}_{\mathrm{b}} = b_{k} - b_{k-1} \tag{10}$$

Through (7), (8), (9), and (10), we can get the pose error of the previous keyframe k. However, before scan-matching, the IMU preintegration error of the current keyframe $k+1$ can not be obtained by using the above method. To get the error of the current keyframe $k+1$, we use the error transfer relation of the nonlinear system:

$$\delta X_{k} = \delta X_{k-1} + \delta \dot{X}_{k-1}\Delta t \tag{11}$$

We are only interested in the $\delta\theta$ and δp of δX, hence our approach can be written as follows:

$$\begin{bmatrix} \delta\theta^{k+1} \\ \delta p^{k+1} \end{bmatrix} = A \begin{bmatrix} \delta\theta^{k} \\ \delta p^{k} \\ \delta v^{k} \\ \delta b^{\mathrm{a}}_{k} \\ \delta b^{\mathrm{g}}_{k} \end{bmatrix} + B \begin{bmatrix} \eta^{\mathrm{g}} \\ \eta^{\mathrm{a}} \\ \eta_{\mathrm{b^g}} \\ \eta_{\mathrm{b^a}} \end{bmatrix} \tag{12}$$

$$A = \begin{bmatrix} I - [\omega]_{\times}\Delta t & 0 & 0 & 0 & -I\Delta t \\ 0 & I & I\Delta t & 0 & 0 \end{bmatrix} \tag{13}$$

$$B = \begin{bmatrix} I\Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{14}$$

By predicting the IMU odometry localization error, we can obtain the initial resolution of the range image. We use the following empirical formula to convert translation and orientation error into resolution:

$$r = \alpha\delta p + \delta\theta \tag{15}$$

Where α is a value between 0 and 1, which is used to balance the contribution of translation error and orientation error to resolution.

To balance the real-time performance and removal rate, and to avoid the influence caused by the wrong prediction resolution, we set a minimum resolution r_0 and appropriately

enlarge the predicted resolution r. The final initial resolution r_{f} is defined as follows:

$$r_{\mathrm{f}} = max(\beta r, r_0) \tag{16}$$

Where β is a coefficient greater than 1. $r_0 = \frac{\text{vertical FOV}}{\text{vertical rays}}$, which can effectively avoid empty pixel values in range image.

D. Range Image Construction and Moving Points Removal

We first define F_{k+1} as the current keyframe scan and M_k as a corresponding submap, which is a point cloud map created by sliding window method around F_{k+1}. In addition, to balance the removal rate of moving points and the real-time performance, we use a full query scan to compare with the feature submap. This is because a feature submap with multiple keyframes has a similar density to the full query scan and has fewer points than a full submap. Then, we divide the point cloud into two mutually exclusive subsets: dynamic points set $(\cdot)^{\mathrm{D}}$, and static points set $(\cdot)^{\mathrm{S}}$. Formally, the aforementioned problem is expressed as:

$$M = M^{\mathrm{D}} \cup M^{\mathrm{S}} \qquad F = F^{\mathrm{D}} \cup F^{\mathrm{S}} \tag{17}$$

while $(\cdot)^{\mathrm{D}} \cap (\cdot)^{\mathrm{S}} = \varnothing$.

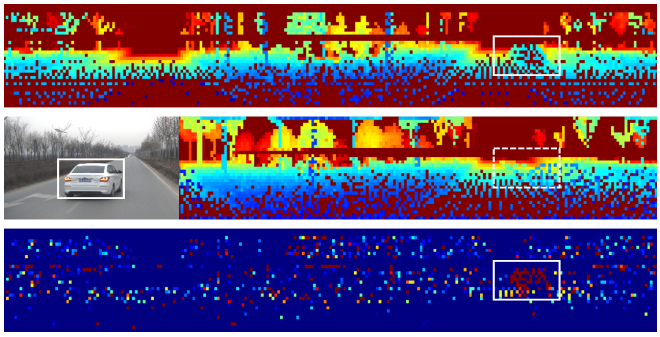


Fig. 3. Adaptive resolution range image. The top row is the range image of the current scan I^{F}_{k+1}. The middle row is the range image of the surrounding submap I^{M}_{k}. The bottom row is the difference range image I^{Diff}_{k+1}. The color of a pixel indicates the distance of a pixel's corresponding 3D point to the sensor keyframe $k+1$; blue indicates the closer distance, and red indicates the farther distance. Therefore, the red pixels in the bottom range image represent the moving points in the scan.

Sharing the analogous philosophy to existing approaches [18], [19], we use the projected range image's visibility to perform moving points recognition. As shown in Fig. 3, The pixel (i, j) value $I_{k+1,ij}$ of the range image is defined as the distance of a point $p \in \mathbb{R}^3$ to the $k+1$th keyframe's local coordinate system B_{k+1}:

$$I^{\mathrm{M}}_{k,ij} = \min_{p \in P^{\mathrm{M}}_{ij}} \mathrm{dist}(p) \qquad I^{\mathrm{F}}_{k+1,ij} = \min_{p \in P^{\mathrm{F}}_{ij}} \mathrm{dist}(p) \tag{18}$$

Where the range image size is determined by the given resolution and the lidar's horizontal and vertical FOV range. $P^{\mathrm{M\,or\,F}}_{ij}$ is the spherical coordinate of the point $P^{\mathrm{M\,or\,F}}$ (i.e. azimuth angle i and elevation angle j).

Then, the visibility of submap points and scan points are calculated via their matrix element-wise subtraction as:

$$I^{\mathrm{Diff}}_{k+1} = I^{\mathrm{F}}_{k+1} - I^{\mathrm{M}}_{k} \tag{19}$$

We determine a point $p \in P^{\mathrm{M\,or\,F}}_{ij}$ is dynamic if its corresponding pixel value of $I^{\mathrm{Diff}}_{k+1,ij}$ is larger than an adaptive threshold τ. Finally, the dynamic points are defined as:

$$M^{\mathrm{D}}_{k} = \left\{ M_k \mid I^{\mathrm{Diff}}_{k+1} > \tau \right\} \tag{20}$$

$$F^{\mathrm{D}}_{k+1} = \left\{ F_{k+1} \mid I^{\mathrm{Diff}}_{k+1} < -\tau \right\} \tag{21}$$

$$\tau = \gamma \mathrm{dist}(p) \tag{22}$$

Where γ is the sensitivity with respect to the point's distance. We suggest setting γ to the maximum of the final resolution shown in Fig. 4(c), which can effectively avoid mistaking the static points for the dynamic points due to the wrong prediction resolution.

E. Lidar Odometry and New Resolution

The lidar odometry is used to estimate the sensor motion between two consecutive scans. Various scan-matching methods, such as [3] and [4], can be utilized for this purpose. We opt to use the method proposed in LOAM [1], as a lot of related works [20], [21] has proved that it has an excellent and robust performance in various challenging environments. Then the edge and plane features are extracted for each new lidar scan by evaluating the roughness of points over a local area. We denote the edge and plane features from the keyframe $k+1$ as F^{e}_{k+1} and F^{p}_{k+1}, respectively. Then we transform them from B_{k+1} to W and obtain $\left\{ {}^{w}F^{\mathrm{e}}_{k+1}, {}^{w}F^{\mathrm{p}}_{k+1} \right\}$. This initial transformation is obtained by IMU preintegration (see Sec. III-B for details). For each feature point in $\left\{ {}^{w}F^{\mathrm{e}}_{k+1}, {}^{w}F^{\mathrm{p}}_{k+1} \right\}$, we can find its correspondence feature points in $\left\{ M^{\mathrm{e}}_{k}, M^{\mathrm{p}}_{k} \right\}$ via nearest neighbor search. Then the distance between a feature point and its corresponding edge or planar patch can be calculated using the following equations:

$$d^{\mathrm{e}}_{k} = \frac{\left| \left(p^{\mathrm{e}}_{k+1,i} - p^{\mathrm{e}}_{k,j} \right) \times \left(p^{\mathrm{e}}_{k+1,i} - p^{\mathrm{e}}_{k,l} \right) \right|}{\left| p^{\mathrm{e}}_{k,j} - p^{\mathrm{e}}_{k,l} \right|} \tag{23}$$

$$d^{\mathrm{p}}_{k} = \frac{\left| \begin{array}{c} \left(p^{\mathrm{p}}_{k+1,i} - p^{\mathrm{p}}_{k,j} \right) \\ \left(p^{\mathrm{p}}_{k,j} - p^{\mathrm{p}}_{k,l} \right) \times \left(p^{\mathrm{p}}_{k,j} - p^{\mathrm{p}}_{k,m} \right) \end{array} \right|}{\left| \left(p^{\mathrm{p}}_{k,j} - p^{\mathrm{p}}_{k,l} \right) \times \left(p^{\mathrm{p}}_{k,j} - p^{\mathrm{p}}_{k,m} \right) \right|} \tag{24}$$

Where i, j, l and m are the feature indices. $p^{\mathrm{e}}_{k+1,i} \in {}^{w}F^{\mathrm{e}}_{k+1}$ is a edge feature point, and $p^{\mathrm{e}}_{k,j}, p^{\mathrm{e}}_{k,l} \in M^{\mathrm{e}}_{k}$ are corresponding edge-line points. $p^{\mathrm{p}}_{k+1,i} \in {}^{w}F^{\mathrm{p}}_{k+1}$ is a plane feature point, and $p^{\mathrm{p}}_{k,j}, p^{\mathrm{p}}_{k,l}, p^{\mathrm{p}}_{k,m} \in M^{\mathrm{p}}_{k}$ are corresponding planar patch points. Finally. we can obtain the transformation by solving the optimization problem:

$$\min \left\{ \Delta T_{k,k+1} \right\} = \min_{\Delta T_{k,k+1}} \left\{ \sum_{p^{\mathrm{e}}_{k+1,i} \in {}^{w}F^{\mathrm{e}}_{k+1}} d^{\mathrm{e}}_{k} + \sum_{p^{\mathrm{p}}_{k+1,i} \in {}^{w}F^{\mathrm{p}}_{k+1}} d^{\mathrm{p}}_{k} \right\} \tag{25}$$

When the scan-matching is convergent, we can get the relative transformation $\Delta T_{k,k+1}$ between the two keyframes k and $k+1$. So the lidar odometry can be written as:

$$T_{k+1} = T_k \Delta T_{k,k+1} \tag{26}$$

However, in a high dynamic environment with multiple moving objects, the lidar odometry will drift. As shown in Fig. 2, we have preliminarily removed the moving points in the steps of Sec. III-C and Sec. III-D, but it still can not guarantee the reliability of scan-matching. Hence we need to judge its convergence. We describe and implement a naive but effective approach, which is judged by the Euler distance-based of the edge points F^{e}_{k+1}. It can be naturally embedded into the our scan-matching method without extra computation. Besides, compared with using all scan points, the sparsity of edge points can ensure that the method is robust enough. We also note that our framework can be compatible with other methods, such as [22] and [23], but these methods need other time to calculate. Our experiments show that our approach is effective enough. It is summarized in Alg. 1.

Algorithm 1 Convergence Score Calculation

Input:

Edge points of current scan: $\{F^{\mathrm{e}}_{k+1}\}$

Edge points of corresponding submap: $\{M^{\mathrm{e}}_{k}\}$

Output:

Convergence Score

Procedure:

Transform $\{F^{\mathrm{e}}_{k+1}\}$ from body frame B_{k+1} to world frame W

Initialize score $= 0$ and number $n_{\mathrm{score}} = 0$

for p^{e}_i in $\{F^{\mathrm{e}}_{k+1}\}$ **do**

  Find its nearest neighbours in $\{M^{\mathrm{e}}_{k}\}$ by KDtree, then calculate their distance

  if distance $\leqslant \tau_{\mathrm{D}}$ **then**

    score$+=$ distance

    $n_{\mathrm{score}} += 1$

if $n_{\mathrm{score}} > 0$ **then**

  score $= \frac{\mathrm{score}}{n_{\mathrm{score}}}$

else

  score $= \infty$

Where τ_{D} is a threshold used to remove outliers that are too far away. When score $<$ score$_0$, it is judged to be convergent, otherwise it is not convergent.

When the scan-matching is not convergent, RF-LIO use (15) and (16) to generate a new resolution better than the initial resolution based on the pose error of lidar Odometry, then removes the dynamic points again and repeats the steps of Set. III-E.

IV. EXPERIMENTS

A. Experimental Setup

TABLE I

PARAMETERS USED IN RF-LIO FOR ALL EXPERIMENTS.

Parameter	α	β	r_0	γ	score$_0$
Value	0.1	2.0	0.02	0.02	0.25

We evaluate our RF-LIO via a series of experiments and compare it with LOAM [1] and LIO-SAM [2]. We note that both RF-LIO and LIO-SAM use the same feature extraction and loop closure detection methods, and both use GTSAM [24] to optimize the factor graph. The ablation experiment with LIO-SAM shows the effect of removal-first in RF-LIO. In all experiments, RF-LIO uses the same parameters shown in Table I. All the methods are implemented in C++ and executed on a computer with an Intel i7-10700k CPU using the robot operating system (ROS) [25] in Ubuntu Linux. For validation, we use self-collected datasets and open UrbanLoco datasets [7], where the Urbanloco datasets contain a large number of moving objects. The details of these datasets are shown in Table II. For a dataset with a few moving objects, we define it as a low dynamic dataset. For a dataset with a large number of moving objects, we define it as a high dynamic dataset. And medium dynamic datasets are between low dynamic datasets and high dynamic datasets. All the methods only use lidar and IMU without GPS. The GPS data is only used as the ground truth.

TABLE II

DATASETS DETAILS

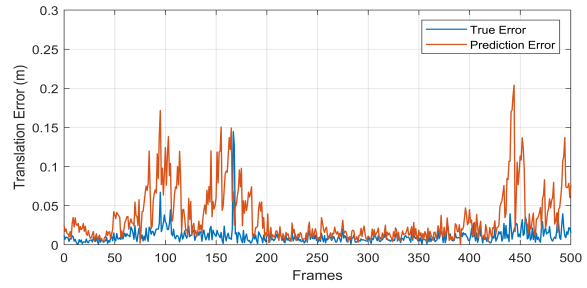
Dataset	Scans	Trajectory Length (m)	Max Speed (m/s)	Average Speed (m/s)	Dynamic Level
Urban	16131	6390.33	5.29	3.89	Low
Campus	5883	1007.97	1.96	1.56	Medium
Suburban	4198	1890.44	6.20	4.64	Medium
CAMarketStreet	14471	5690.98	13.44	7.07	High
CARussianHill	15860	3570.38	10.27	4.98	High

B. Initial Resolution Analysis

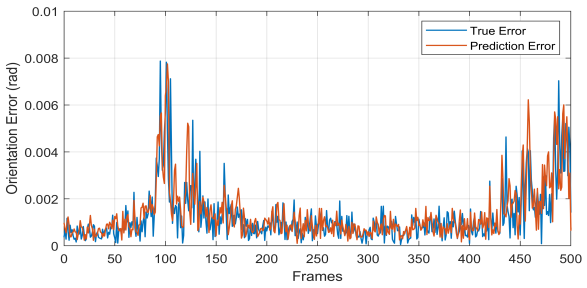
Setting the correct initial resolution of the range image can effectively remove moving points and avoid the mistake of removing fixed points. Therefore, this experiment is designed to prove the correctness of the method proposed in Sec. III-C. From Fig. 4(b), we can see that the predicted orientation error is in good agreement with the true error. Because the translation is the double-integrating of acceleration, the predicted translation error is not as accurate as the predicted orientation error. Fig. 4(a) shows that the deviation between the predicted translation error and the true translation error is always within an acceptable range. As shown in Fig. 4(c), the prediction resolution is almost consistent with the true resolution, and the final resolution is always bigger than the true resolution. It makes our method can effectively reduce the mistake of identifying moving points.

C. Moving Objects Removal Test

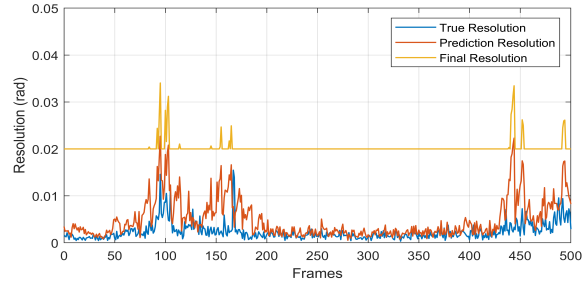
In this test, the removal rate of moving points is used to evaluate RF-LIO. For this purpose, we collected three datasets with multiple moving objects in different environments. Fig. 5(a) shows a snapshot of the suburban dataset with many moving vehicles. RF-LIO has the same feature extraction method as LIO-SAM. Therefore, we use LIO-SAM as the evaluation standard to evaluate the moving points removal rate of RF-LIO. The maps obtained from LIO-SAM and RF-LIO are shown in Figs. 5(b) and 5(c), respectively. As is shown, LIO-SAM renders a lot of ghost tracks in the



(a) Translation Error



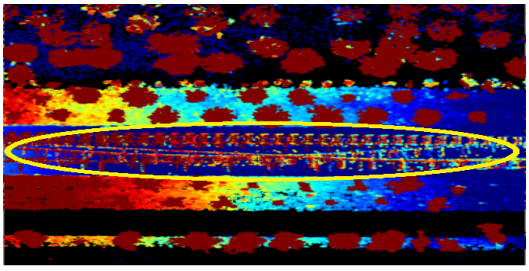
(b) Orientation Error



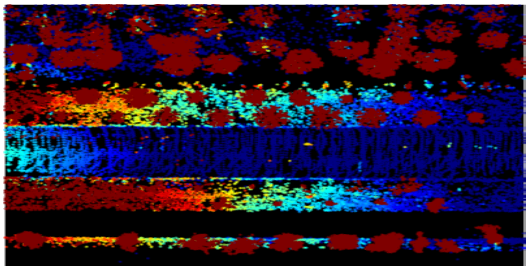
(c) Initial Resolution

Fig. 4. Translation error, orientation error and initial resolution analysis. (a) The translation error of IMU preintegration. (b) The orientation error of IMU preintegration. (c) The resolution curve. The true resolution and the prediction resolution are calculated by the true error and the prediction error, respectively. The final resolution is an appropriate amplification of the prediction resolution to reduce the impact of a wrong prediction.

(a) Test environment



(b) LIO-SAM



(c) RF-LIO

Fig. 5. Mapping results of LIO-SAM and RF-LIO in the dynamic environment of the suburban dataset.

TABLE III
REMOVAL RATE OF MOVING OBJECT POINTS

Dataset	LIO-SAM	RF-LIO	Removal Rate
Urban	85890	1803	97.9%
Campus	134092	5766	95.7%
Suburban	198113	8898	95.5%
Average	139365	5489	96.1%

point cloud map. Compared with it, RF-LIO can get a purer map. We count the number of residual moving points in the maps of LIO-SAM and RF-LIO, respectively, and then calculate the removal rate. The results are shown in Table III. We can see that RF-LIO achieves an average removal rate of 96.1% compared with LIO-SAM. We note that RF-LIO does not completely remove all the moving points. Because some moving points are too close to the ground ($distance < 0.5$, which is less than the threshold we set), and others are generated by lidar beams parallel to the ground, which makes it difficult to find the corresponding far points in the submap to remove them.

D. Results on Low and Medium Dynamic Datasets

In this experiment, we use three self-collected datasets with only a few moving objects. Since LOAM and LIO-SAM are designed in a static environment, we compare with them to show the performance of RF-LIO in general cases. The urban dataset includes a wide variety of urban terrains: residential area, overpass, construction area, etc. The details and final point cloud map from RF-LIO are shown in Fig. 6. For an intuitive display, the map of RF-LIO is overlaid on a satellite image. The campus dataset is collected from the XJTU campus with multiple pedestrians. As shown in Fig. 5(a), the suburban dataset contains various moving vehicles.

To show the benefits of removing moving points before scan-matching, we divide RF-LIO into three different types. When removing moving points first and scan-matching later, our method is referred to as RF-LIO (First). Similarly, when the moving points are removed after scan-matching, our method is referred to as RF-LIO (After). When first removing moving points, then scan-matching, and finally removing moving points again, our method is referred to as RF-LIO (FA). The absolute trajectory error of all methods is shown in Table IV. LOAM does not perform well on

Fig. 6. RF-LIO map aligned with satellite image, and some snapshots of the environment.

all three datasets. Therefore here we only compare with LIO-SAM. From the results, it can be seen that RF-LIO (After) and LIO-SAM have similar performance, while RF-LIO (First) and RF-LIO (FA), using removal-first, have a significant improvement. Compared with LIO-SAM, RF-LIO (FA) improves the absolute trajectory accuracy by 36.7%, 3%, and 18.3%, respectively.

TABLE IV
ABSOLUTE TRAJECTORY RMSE [M] FOR ALL METHODS USING LOW AND MEDIUM DYNAMIC DATASETS

Dataset	LOAM	LIO-SAM	RF-LIO (After)	RF-LIO (First)	RF-LIO (FA)
Urban	244.19	10.21	10.79	7.72	**6.46**
Campus	118.49	0.66	0.66	0.64	**0.61**
Suburban	Fail	1.53	1.51	1.42	**1.25**

E. Results on High Dynamic Datasets

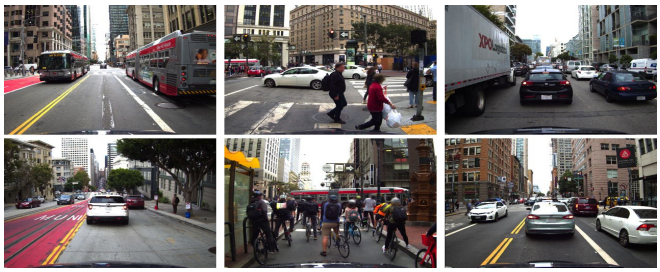


Fig. 7. Some screenshots of the urbanloco datasets. There are many moving objects, which block lidar's FOV and introduce outliers.

This experiment is designed to demonstrate the superiority of our method in a challenging high dynamic environment. We select the open UrbanLoco datasets as the test dataset, which is collected in highly urbanized scenes with numerous moving objects. Some screenshots of the test environment are shown in Fig. 7. In this experiment, when the feature points fall on the moving object, the LOAM map shown in Fig. 8(b) diverges in multiple locations. LIO-SAM outperforms LOAM in this test, and its map is shown in Fig. 8(c). Compared with LIO-SAM, RF-LIO has smaller drift and better closed-loop detection performance due to the removal of moving objects. The absolute trajectory error of all methods is shown in Table V.

TABLE V
ABSOLUTE TRAJECTORY RMSE [M] FOR ALL METHODS USING HIGH DYNAMIC DATASETS

Dataset	LOAM	LIO-SAM	RF-LIO (After)	RF-LIO (First)	RF-LIO (FA)
CAMarktStreet	203.78	62.43	23.98	**15.83**	15.89
CARussianHill	175.69	36.98	12.91	12.44	**12.17**

In the high dynamic datasets, RF-LIO has a more prominent effect on SLAM performance improvement. Compared with LIO-SAM, RF-LIO (After) improves the absolute trajectory accuracy by 61.6% and 65.1%, respectively. The experimental results demonstrate that our moving objects removal method can effectively remove dynamic feature points and improve SLAM performance. On these two datasets, RF-LIO (First) outperforms RF-LIO (After) by 34.0% and 3.6%, respectively. The results are consistent with the results on low and medium dynamic datasets, but more significant. So we can conclude that removal-first is another effective method to improve SLAM performance in dynamic environments. Because removal-first has removed most of the dynamic points, the trajectory error of RF-LIO (First) and RF-LIO (FA) are not significantly different. In summary, RF-LIO (FA) achieves the best results among all methods.

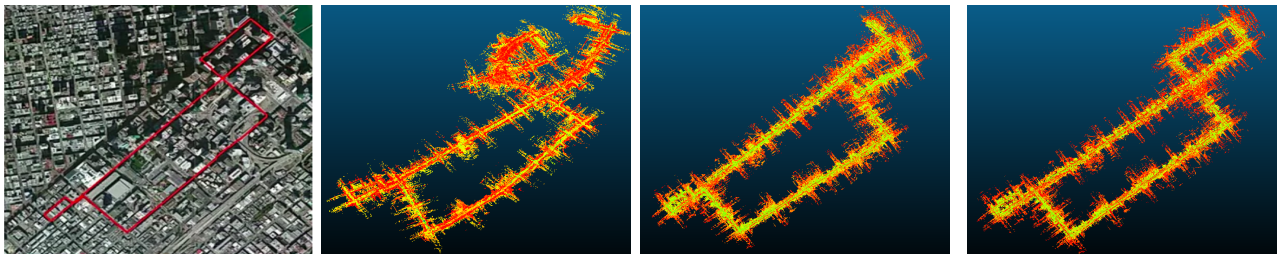
In practical application, real-time performance is another crucial indicator to evaluate SLAM systems. We test the runtime of RF-LIO (After), RF-LIO (First), and RF-LIO (FA) on all five datasets. The results are shown in Table VI. We see that the runtime of RF-LIO (FA) is less than 100 ms in low dynamic environments, while in high dynamic environments, it is also less than 121 ms. In addition, we note that the runtime of RF-LIO (First) is significantly less than RF-LIO (After), and even the runtime of RF-LIO (FA) is also less than RF-LIO (After). The experimental results show that removal-first takes extra time to remove moving objects, but a clean point cloud can reduce scan-matching time. Although removal-after also removes the moving points, the scan-matching has been completed at this time. Therefore the effect is not as good as removal-first.

TABLE VI
RUNTIME OF RF-LIO FOR PROCESSING ONE SCAN

Dataset	RF-LIO (After)	RF-LIO (First)	RF-LIO (FA)
Urban	86 ms	55 ms	61 ms
Campus	97 ms	68 ms	74 ms
Suburban	118 ms	97 ms	112 ms
CAMarktStreet	105 ms	93 ms	96 ms
CARussianHill	134 ms	107 ms	121 ms

V. CONCLUSIONS

We propose RF-LIO to perform real-time and robust state estimation and mapping in high dynamic environments. RF-LIO employs moving objects removal-first algorithm com-



(a) Ground truth (b) LOAM (c) LIO-SAM (d) RF-LIO

Fig. 8. Mapping results of LOAM, LIO-SAM, and RF-LIO using the CAMarketStreet dataset.

bined with tightly-coupled LIO to solve the chicken-and-egg problem of first removing the dynamic points or first scan-matching in high dynamic environments. The proposed adaptive range image moving points removal algorithm does not rely on any prior training data, nor is it limited by the category and number of moving objects. Hence RF-LIO can be applied to various scenes robustly.

However, RF-LIO still has some ongoing works. In a very open environment, if there are no corresponding far points in the surrounding environment, the visibility-based range image method can not remove moving points. Another problem is that when moving objects completely block the FOV of our sensor, the method is not suitable.

REFERENCES

[1] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time." *Robotics: Science and Systems*, vol. 2, no. 9, 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," *arXiv preprint arXiv:2007.00258*, 2020.

[3] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp." *Robotics: science and systems*, vol. 2, no. 4, p. 435, 2009.

[4] P. Biber and W. Straßer, "The normal distributions transform: A new approach to laser scan matching," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 3. IEEE, 2003, pp. 2743–2748.

[5] S. Pagad, D. Agarwal, S. Narayanan, K. Rangan, H. Kim, and G. Yalla, "Robust method for removing dynamic objects from point clouds," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 765–10 771.

[6] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma++: Efficient lidar-based semantic slam," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4530–4537.

[7] W. Wen, Y. Zhou, G. Zhang, S. Fahandezh-Saadi, X. Bai, W. Zhan, M. Tomizuka, and L.-T. Hsu, "Urbanloco: a full sensor suite dataset for mapping and localization in urban scenes," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2310–2316.

[8] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1168–1174.

[9] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[10] C. Chen and B. Yang, "Dynamic occlusion detection and inpainting of in situ captured terrestrial laser scanning point clouds sequence," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 119, pp. 90–107, 2016.

[11] J. Gehrung, M. Hebel, M. Arens, and U. Stilla, "An approach to extract moving objects from mls data using a volumetric background representation." *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 4, 2017.

[12] 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.

[13] R. Ambruş, N. Bore, J. Folkesson, and P. Jensfelt, "Meta-rooms: Building and maintaining long term spatial models in a dynamic world," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 1854–1861.

[14] W. Xiao, B. Vallet, M. Brédif, and N. Paparoditis, "Street environment change detection from mobile laser scanning point clouds," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 107, pp. 38–49, 2015.

[15] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[16] C. Le Gentil, T. Vidal-Calleja, and S. Huang, "In2lama: Inertial lidar localisation and mapping," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6388–6394.

[17] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration theory for fast and accurate visual-inertial navigation," *IEEE Transactions on Robotics*, pp. 1–18, 2015.

[18] E. Palazzolo and C. Stachniss, "Fast image-based geometric change detection given a 3d model," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6308–6315.

[19] G. Kim and A. Kim, "Remove, then revert: Static point cloud map construction using multiresolution range images," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE/RSJ, 2020.

[20] 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.

[21] C. Qin, H. Ye, C. E. Pranata, J. Han, S. Zhang, and M. Liu, "Lins: A lidar-inertial state estimator for robust and efficient navigation," *arXiv preprint arXiv:1907.02233*, 2019.

[22] X. Chen, T. Läbe, A. Milioto, T. Röhling, O. Vysotska, A. Haag, J. Behley, C. Stachniss, and F. Fraunhofer, "Overlapnet: Loop closing for lidar-based slam," *Robotics: Science and Systems (RSS)*, 2020.

[23] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, and R. Siegwart, "Long-term 3d map maintenance in dynamic environments," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 3712–3719.

[24] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "isam2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[25] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.