2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

DynaInsRemover: A Real-time Dynamic Instance-Aware Static 3D LiDAR Mapping Framework for Dynamic Environment

Huanfeng Zhao, Meibao Yao*, Xueming Xiao and Bo Zheng

***Abstract*— Dynamic objects diversify the distribution of point cloud in the map, degrading the performance of the robotic downstream tasks. To address this problem, we present a novel real-time dynamic instance-aware static mapping framework called *DynaInsRemover*, which exploits the geometric discrepancies between instances to efficiently remove dynamic objects and preserve more details of static map. It contains the *Instance Occupancy Check* module for initial dynamic instance proposal and the *Instance Belief Update* module for reverting false positives. We quantitatively evaluate our approach performance on the SemanticKITTI dataset and validate it in a real-world environment. Experimental evaluations show that our method achieves very promising results in dynamic environments. The implementation of our method is available as open source at: `https://github.com/Zhaohuanfeng/DynaInsRemover.git`.**

I. Introduction and Related Works

Point cloud map is one of the common map representations that support advanced robotic tasks, such as path planning [1], and re-localization [2], life-long SLAM [3]. Using 3D LiDAR sensors and SLAM [4]–[9], we can easily reconstruct the spatial structure of the environment. As 3D snapshot of the environment, LiDAR scan records all moving objects, such as moving vehicles and pedestrians. As shown in Fig. 1, the point cloud map is the artifact of the scans stacked onto the trajectory, so moving objects continue to accumulate resulting in the ghost effect [10]. The dynamic objects trapped in the map diversify the spatial structure and thus deteriorate the robotic autonomous performance. Therefore, it is important to remove dynamic points from the map.

Among many schemes, learning-based approaches achieve impressive performance [11]–[20], allowing the segmentation of dynamic objects in a single scan. Learning-based approaches typically use labeled datasets to supervise the training of deep neural networks. Chen *et al*. [11] leveraged sequential range residual images as inputs to the point cloud semantic segmentation network to classify dynamic and static objects in LiDAR scans. Wu *et al*. [16] developed a novel framework for joint perception and motion prediction from LiDAR scans, using bird's eye view map of temporal sequences. Mersch *et al*. [17], [18] aggregated secutive past scans into a 4D point cloud map, incorporating computationally efficient sparse 4D convolutions and binary Bayes filter to improve moving object segmentation performance. Song *et al*. [20] proposed a novel method for unsupervised rigid dynamic point cloud segmentation by training object segmentation network and self-supervised scene flow network with the object geometry consistency constraint. However, learning-based approaches focus on the detection of dynamic points and not explicitly generate static map. Besides, they rely on high-quality data labeling, suffer class-imbalance during training, and cannot identify unlabeled classes. To this end, we focus on geometry-based approaches for static map construction.

Traditional geometry-based approaches reconstruct the static structure of environments without any semantic labels [21]–[32], which include two paradigms: (1) remove dynamic objects in post-processing of the complete point cloud map [23]–[27], [30], [32] and (2) preserve static parts

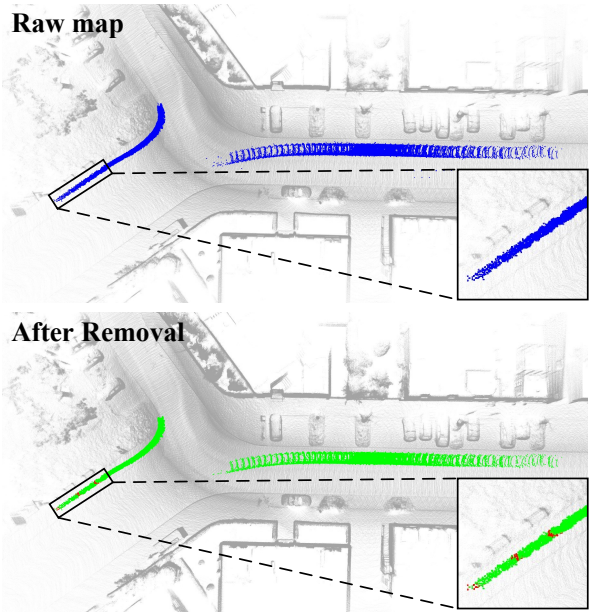


Fig. 1. *Top*: The raw map on Seq. 00 of SemanticKITTI around frame 4390. The blue points are the dynamic objects that need to be removed, and the gray points are the static parts that need to be preserved. *Bottom*: The map after applying our method. The green points are the dynamic objects of successful removal, and the red points are the false positives that remain in the static map.

Huanfeng Zhao and Meibao Yao are with the Intelligent Robotics Lab (IRL), School of Artificial Intelligence, Jilin University, Changchun 130012, China; Engineering Research Center of Knowledge-Driven Human-Machine Intelligence, Ministry of Education, China. Emails: hfzhao20@mails.jlu.edu, meibaoyao@jlu.edu.cn. Xueming Xiao is with the CVIR lab, Changchun University of Science and Technology, Changchun 130022, China; Key Lab of Opto-electronic Measurement and Optical Information Transmission Technology. Email: alexcapshow@gmail.com. Bo Zheng is with the Shanghai Aerospace Control Technology Institute, Shanghai 201109, China. Email: zhengbohit@gmail.com.

This work was sponsored by the National Natural Science Foundation of China (NSFC) through grants No. 62103163 and No. 62003055, Natural Science Foundation of Jilin Province through grant No. YDZJ202101ZYTS033, and Natural Science Foundation of Shanghai through grant No. 22ZR1479600. We thank the above mentioned funds for their financial support.

Meibao Yao* is Corresponding Author

with online filtering of dynamic objects [22], [31]. The latter is more promising as it not only generates static map online but also facilitates online robotic tasks such as localization and path planning. Therefore, this paper focuses on real-time dynamic object removal and static point cloud map construction from LiDAR scans.

Furthermore, the geometry-based approaches are mainly categorized as ray tracing-based, visibility-based, and other approaches. Ray tracing-based approaches assume that the voxel hit at the end of a ray to be occupied and the voxel through which the ray passes is free space. In 2D LiDAR perception and mapping, occupancy grid map [21] is widely used, capable of computing occupancy probability through checking whether a ray runs through voxel space or not. OctoMap [22] implements ray tracing and occupancy probability updating in 3D space using an octree representation of map. Schauer *et al*. [23] approximated occupied volume by a voxel grid and determine dynamic points by traversing the lines of sight from the sensor to the measured points through the voxel grid. Pfreundschuh *et al*. [24] proposed an offline dynamic objects labeling method for training deep neural network by occupancy grid map. Lukas *et al*. [25] used TSDF-based map representation to estimate high confidence free-space areas and detected dynamic points in point clouds. Although these approaches work well, ray tracing in 3D space is computationally expensive.

To reduce the computational burden, visibility-based approaches consider that if a query point is occluded in the line of sight of a previously point in map, then the previously acquired point to be dynamic. Kim *et al*. [27] proposed a novel remove-then-revert mechanism to construct static map using multi-resolution range images based on visibility. Qian *et al*. [28] suggested a combination of adaptive multi-resolution range images and tightly-coupled LiDAR inertial odometry to remove moving objects. Park *et al*. [29] achieved robust motion estimation in dynamic urban environments by leveraging multiple range residual images to estimate nonparametric background model. However, both ray tracing and visibility-based approaches degrade performance due to incidence angle ambiguity and occlusion issues.

To overcome the limitations of incidence angle ambiguity and occlusion, Lim *et al*. [30] proposed a visibility-free approach that compares the ratio of the relative height in regions between the query scan and the map. However, this approach requires limiting the height range to mitigate the object overlap issue, for instance, a dynamic object passes under a static object, as shown in Fig. 6 (d) (cyan box). Fan *et al*. [31] proposed an online dynamic object removal framework by map-based reverting algorithm for visibility issues and visibility check for the acceleration of ray tracing. The above geometry-based approaches apply voxel-wise, point-wise and region-wise dynamic object removal, which result in falsely removing parts of object in static map, as shown in Fig. 6. Lim *et al*. [32] considered these limitations and proposed an instance-aware dynamic object removal method based on [30], while this approach operates in an offline paradigm can hardly support online tasks.

In this paper, we propose a novel instance-aware static map construction framework with real-time performance, *DynaInsRemover*, to overcome the limitations of the above methods. Our approach applies instance-wise dynamic object removal while adding instance-level static objects to the map, which preserves more details of the static spatial structure.

This work makes the following contributions:

- We propose a new online static map construction framework, which utilizes instance-level information to remove dynamic objects and refine the details of static map. To our best knowledge, it is the first work focusing on the instance-aware dynamic object removal with real-time performance.
- We propose *Instance Occupancy Check* that computes the similarity score among occupancy descriptors encapsulating the same object in different viewpoints and identifies potential dynamic objects.
- We propose *Instance Belief Update* based on instance occupancy descriptor and binary Bayes filter, which reverts static points that are falsely classified to dynamic ones.
- We quantitatively evaluate the performance of the framework on the SemanticKITTI dataset, and validate the online performance of the algorithm in real world scenarios.

II. Methodology

This section presents our instance-aware dynamic object removal mapping framework, comprising three modules: *Instance Segmentation and Association*, *Instance Occupancy Check* and *Instance Belief Update*. Fig. 2 provides the schematic diagram of the framework.

A. Problem Statement and Notations

Our framework uses the consecutive scans and poses generated by the state-of-the-art SLAM or LiDAR odometry algorithms as input. Let $\mathcal{P}_t = \{\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_N\}$ be the t-th point cloud in the LiDAR frame that contains N points, where each point is expressed as $\mathbf{p} = [x, y, z]^\mathsf{T} \in \mathbb{R}^3$ in Cartesian coordinates. We first divide $\mathcal{P}_t$ into two parts: the assumed static ground points $\mathcal{G}_t$ and the non-ground points $\hat{\mathcal{P}}_t$, which satisfy $\mathcal{G}_t \cup \hat{\mathcal{P}}_t = \mathcal{P}_t$ and $\mathcal{G}_t \cap \hat{\mathcal{P}}_t = \varnothing$. Ground segmentation facilitates our focus on objects above the ground and overcomes the problems of incidence angle ambiguity and under-segmentation [33].

Thereafter, we use a clustering method to segment $\hat{\mathcal{P}}_t$ into M instances, $\mathcal{I}_t = \{\mathcal{S}_t^1, \mathcal{S}_t^2, \cdots, \mathcal{S}_t^M\}$, each of which contains a number of points and follows $\mathcal{S}_t^a \cap \mathcal{S}_t^b = \varnothing$ if $a \neq b$. For the same instance, it can be observed by multiple scans at different times. We define the observations of the same instance in a temporal window as a cluster. If an instance is dynamic, then the instances in the cluster to which it belongs exhibit geometric discrepancies. We utilize this property to estimate dynamic instances while accumulate static instances to construct static map.

Let $\mathbf{T}_t \in SE(3)$ be the estimated transformation matrix of the t-th LiDAR frame with respect to the world frame.

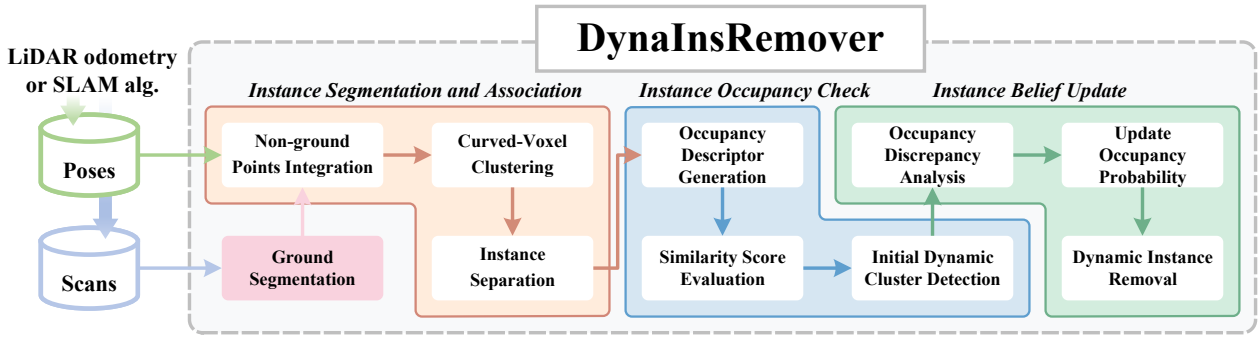
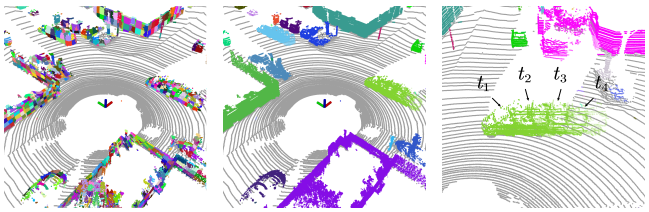


Fig. 2. The schematic of *DynaInsRemover* framework. (a) *Instance Segmentation and Association*: A limited time window of consecutive past scans are aggregated to current scan viewpoint after the ground segmentation preprocessing, a clustering algorithm is used to group the aggregated point cloud into multiple clusters, which contain multiple associated instances i.e. observations of the same object in different viewpoints during the time window. (b) *Instance Occupancy Check*: The instances are encapsulated as occupancy descriptors and the instance similarity scores are computed for initial dynamic instance proposal. (c) *Instance Belief Update* The probability that an instance is truly dynamic or not is updated by analyzing the occupancy discrepancies between the initial dynamic instances. Instances with high probability are removed while the rest of the instances are retrieved to the static map.



(a) (b) (c)

Fig. 3. Example of *Instance Segmentation and Association* for Seq. 07 of SemanticKITTI around frame 670. The gray points indicate extracted ground points. (a) Visualization of curved-voxelization of the aggregated scans, where different colors indicate different curved-voxels. (b) Instance proposal, different colors indicate different instance proposals. (c) In the same cluster, associated instances are distinguished and extracted by timestamps as labels.

By transforming the static point cloud to the world frame, the static map $\mathcal{M}_s$ can be formulated as follows:

$$\mathcal{M}_s = \bigcup_{t\in[T]} \mathbf{T}_t * \mathcal{P}_{s,t}, \text{ where} \tag{1}$$

$$\mathcal{P}_{s,t} = \mathcal{P}_t - \bigcup_{\mathcal{S}\in\hat{\mathcal{I}}_t} \mathcal{S}. \tag{2}$$

Here $[T] = \{1, 2, \cdots, T\}$ is the set of scan timestamps; $\mathcal{P}_{s,t}$ denotes the static points; $\hat{\mathcal{I}}_t$ means the estimated dynamic instances, which is a subset of $\mathcal{I}_t$, i.e., $\hat{\mathcal{I}}_t \subset \mathcal{I}_t$.

B. Instance Segmentation and Association

In the field of multi-object tracking [34], data association algorithms are often used to match objects in different frames. But this approach cannot be applied to our system for instance association, because we are concerned with all the instances in the scene rather than particular classes. Thus, we adopt the instance segmentation and association strategy of clustering followed by separation, which is based on the fact that the points in short sequential scans are continuously distributed. Given K temporally consecutive scans $\{\mathcal{P}_{t_1}, \cdots, \mathcal{P}_{t_K}\}$ as input, whose timestamps form a temporal window $[W] = \{t_1, \cdots, t_K\}$. We aim to segment the scans into multiple instance hypotheses and identify their correspondences. To achieve this, we first segment ground points by the off-the-shelf ground segmentation approach [35]. Next, we integrate the non-ground points of the scans to the t_K-th viewpoint by:

$$\mathcal{P}_{t_1:t_K} = \bigcup_{k\in[W]} \mathbf{T}_{t_K}^{-1}\mathbf{T}_k * \hat{\mathcal{P}}_k. \tag{3}$$

As shown in Fig. 3, we employ curved-voxel clustering algorithm [36] to perform instance segmentation and association. In the clustering, we use the spherical coordinates of the points to separate $\mathcal{P}_{t_1:t_K}$ into a number of curved-voxels. For efficiency, the indices of non-empty curved-voxel are mapped to a hash table. We utilize breadth-first search and hash-table based k-NN search to group the curved-voxels into multiple clusters, $\mathcal{I}_{t_1:t_K} = \{\mathcal{S}^1_{t_1:t_K}, \cdots, \mathcal{S}^M_{t_1:t_K}\}$, each of which consists of multiple instances during $[W]$. After the segmentation, instances in a cluster are coupled within each other, we use the point timestamp as label to classify a cluster into multiple instance hypothesis. Formally, we define the separated instance of $\mathcal{S}_{t_1:t_K}$ at $k \in [W]$ as follows:

$$\mathcal{S}_k = \{\mathbf{p} \in \mathcal{S}_{t_1:t_K} \mid t(\mathbf{p}) = k\}, \tag{4}$$

where $t(\mathbf{p})$ returns the timestamp of a point. Through this clustering with separation mechanism we can achieve instance segmentation and association.

C. Instance Occupancy Check

To efficiently obtain the initial dynamic cluster, we consider the utilization of instance-level geometric discrepancy and present *Instance Occupancy Check* inspired by Scan Context [37], [38]. We design the occupancy descriptor to encapsulate instances, the first step of which is to divide $\mathcal{S}_k$ into regular sector grids on the x-y plane:

$$\mathcal{S}_k = \bigcup_{i\in[N_r], j\in[N_\theta]} \mathcal{S}_k(i,j), \text{ with} \tag{5}$$

$$N_r = \lceil \frac{r_{\max} - r_{\min}}{s_r} \rceil, \tag{6}$$

$$N_\theta = \lceil \frac{\theta_{\max} - \theta_{\min}}{s_\theta} \rceil, \tag{7}$$

where $\mathcal{S}_k(i,j)$ indicates the set of points in the (i,j)-th sector; N_r and N_θ are the number of grids along over the

radial and azimuthal directions, respectively; $r_{\max}$, $r_{\min}$, $\theta_{\max}$ and $\theta_{\min}$ denote the radial and azimuthal boundary values of $\mathcal{S}_{t_1:t_K}$, respectively; s_r and s_θ denote the radial and azimuthal grid size, respectively. Each sector grid contains the points that satisfy the following condition:

$$\mathcal{S}_k(i,j) = \{\mathbf{p} \in \mathcal{S}_k \mid (i-1)\cdot s_r \le r(\mathbf{p}) - r_{\min} < i\cdot s_r, \\ (j-1)\cdot s_\theta \le \theta(\mathbf{p}) - y_{\min} < j\cdot s_\theta\}. \tag{8}$$

where $r(\mathbf{p}) = \sqrt{x^2+y^2}$ and $\theta(\mathbf{p}) = \arctan 2(y,x)$.

Let $h : \mathcal{S}_k(i,j) \mapsto \mathbb{R}$ be the function that encodes a grid with an occupancy value. We take the relative height of a grid as grid occupancy. Formally, the grid encoding function is defined as:

$$h(\mathcal{S}_k(i,j)) = \max_{\mathbf{p}\in\mathcal{S}_k(i,j)} z(\mathbf{p}) - z_{\min}, \tag{9}$$

where $z_{\min}$ denotes the minimum height of $\mathcal{S}_{t_1:t_K}$; $z(\mathbf{p})$ is the z-coordinate value of a point. We stack all the values of the grids into an $N_r \times N_\theta$ matrix as the occupancy descriptor for $\mathcal{S}_k$:

$$\mathbf{H}_k = (h_k^{ij}) \in \mathbb{R}^{N_r \times N_\theta},\ h_k^{ij} = h(\mathcal{S}_k(i,j)). \tag{10}$$

A grid may be empty if $\mathcal{S}_k(i,j)$ contains no points, in which case we assign zero to the grid, as $h_k^{ij} = 0$.

Next, we compute the occupancy descriptor similarity score between the instances in a cluster and consider the cluster with low similarity as the initial dynamic cluster proposals. For an object, whether it is static or dynamic, identifying its state using temporally close occupancy descriptors is difficult because the descriptors are too similar. In our case, the newest instance $\mathcal{S}_{t_K}$ and the oldest instance $\mathcal{S}_{t_1}$ are taken to generate instance occupancy descriptors and a cosine similarity is used to compute a similarity. Therefore, the similarity function can be defined as:

$$d(\mathcal{S}_{t_1:t_K}) = \frac{\mathbf{h}_{t_1}\cdot\mathbf{h}_{t_K}}{\|\mathbf{h}_{t_1}\|\|\mathbf{h}_{t_K}\|}, \tag{11}$$

where $\mathbf{h}$ is the descriptor vector flattened by a descriptor matrix $\mathbf{H}$, which allows for simple computations.

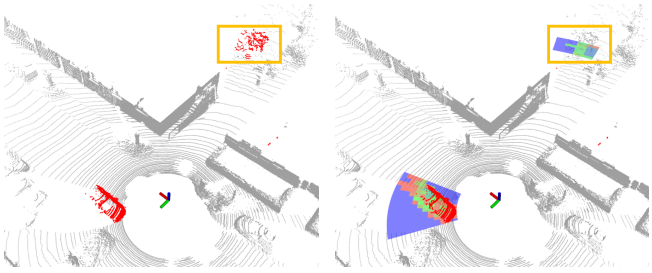
We consider the instances with low score as initial dynamic clusters $\hat{\mathcal{I}}^{init}_{t_1:t_K}$, which is defined as:

$$\hat{\mathcal{I}}^{init}_{t_1:t_K} = \{\hat{\mathcal{S}}_{t_1:t_K} \in \mathcal{I}_{t_1:t_K} \mid d(\hat{\mathcal{S}}_{t_1:t_K}) < \tau_s\}, \tag{12}$$

where τ_s is the threshold of occupancy similarity. As shown in Fig. 4 (a), through *Instance Occupancy Check*, the initial dynamic clusters can be successfully estimated.

D. Instance Belief Update

Ego-motion of LiDAR and occlusion result in incompletely visible object and low occupancy similarity score, which further leads to a static cluster of false positive. As shown by the yellow box in Fig. 4(a), a tree is falsely classified as dynamic. To address this problem, we propose the *Instance Belief Update* mechanism to validate whether an object is truly dynamic or not in a probabilistic method. The core idea is to take the differences in occupancy between the instance and the cluster it belongs to as observations to update the probability that the cluster is dynamic based on a binary Bayes filter.



(a) (b)

Fig. 4. (a) Example of *Instance Occupancy Check* for Seq. 00 of SemanticKITTI around frame 4470, where red points indicate initial dynamic instances. The static points in green box are falsely classified as dynamic points (b) Procedure of our *Instance Belief Update*, below the instance is the updated occupancy grid map, where light blue indicates empty grids, light green indicates grids with low dynamic probability and light red indicates grids with high dynamic probability.

To achieve this, we first build a grid map for $\hat{\mathcal{S}}_{t_1:t_K} \in \hat{\mathcal{I}}^{init}_{t_1:t_K}$ using the grid division in Section II-C, i.e., the sector grid. Next, we update the probability of the grid map based on a binary Bayes filter to fuse each measurement. The conditional probability distribution of the grid map state can be updated by:

$$p(\mathbf{m} \mid \mathbf{z}_{t_1:t_K}) = (1+p)^{-1} \tag{13}$$

$$p = \frac{1-p(\mathbf{m}\mid\mathbf{z}_{t_K})}{p(\mathbf{m}\mid\mathbf{z}_{t_K})}\frac{1-p(\mathbf{m}\mid\mathbf{z}_{t_1:t_{K-1}})}{p(\mathbf{m}\mid\mathbf{z}_{t_1:t_{K-1}})}\frac{p(\mathbf{m})}{1-p(\mathbf{m})} \tag{14}$$

where $\mathbf{m} = \{0,1\}$ denotes the binary state of whether $\hat{\mathcal{S}}_{t_1:t_K}$ is dynamic or not and $\mathbf{z}_{t_1:t_K}$ denotes the observed measurements during the temporal window, each of which is the occupancy discrepancy between the instance and the cluster; $p(\mathbf{m} \mid \mathbf{z}_{t_K})$ is the probability of $\hat{\mathcal{S}}_{t_1:t_K}$ to be dynamic given the measurement $\mathbf{z}_{t_K}$; $p(\mathbf{m} \mid \mathbf{z}_{t_1:t_{K-1}})$ is the previous prediction; $p(\mathbf{m}) = 0.5$ is a prior probability of the state;

By using the Bayes rule and the log-odd notation $l(x) = logit(p(x)) = \log\left(\frac{p(x)}{1-p(x)}\right)$, the Eq. (13) can be paraphrased as follows:

$$l(\mathbf{m} \mid \mathbf{z}_{t_1:t_K}) = l(\mathbf{m} \mid \mathbf{z}_{t_1:t_{K-1}}) + l(\mathbf{m} \mid \mathbf{z}_{t_K}). \tag{15}$$

The next step is to update $l(\mathbf{m} \mid \mathbf{z}_{t_K})$ from the occupancy discrepancies between $\hat{\mathcal{S}}_{t_K}$ and $\hat{\mathcal{S}}_{t_1:t_K}$. For the non-empty grid with indices (i,j), the occupancy discrepancy of the grid is defined as:

$$\rho(i,j) = \frac{\min\left(h^{ij}_{t_K}, h^{ij}_{t_1:t_K}\right)}{\max\left(h^{ij}_{t_K}, h^{ij}_{t_1:t_K}\right)}, \tag{16}$$

where $h^{ij}_{t_K}$ and $h^{ij}_{t_1:t_K}$ are the grid occupancies given by Eq. (9) and (10). Since all non-empty grids form the cluster, we need to aggregate the occupancy discrepancies of all non-empty grids to determine the state. Let $\mathcal{R}$ be the occupancy discrepancies set of non-empty grids for $\hat{\mathcal{S}}_{t_1:t_K}$, the update term is expressed as:

$$l(\mathbf{m} \mid \mathbf{z}_{t_K}) = \frac{1}{|\mathcal{R}|}\sum_{\rho\in\mathcal{R}} g(\rho), \tag{17}$$

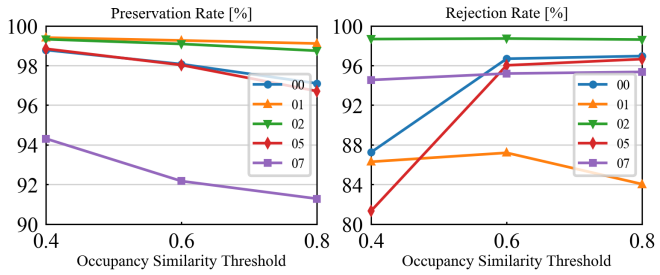


Fig. 5. The performance with changing occupancy similarity threshold τ_s.

where $g(\rho)$ can adaptively set a log-odd value for a grid, which is defind as:

$$g(\rho) = \begin{cases} 2 \cdot logit(p_{inc}), & \text{if } \rho = 0 \\ logit(p_{inc}), & \text{if } \rho < \tau_\rho \\ logit(p_{dec}), & \text{otherwise} \end{cases}, \tag{18}$$

where $p_{inc} > 0.5$ is the incremental update probability when the grid is dynamic; $p_{dec} < 0.5$ is the decremental update probability when the grid is static; τ_ρ is the discrepancy threshold. Finally, the estimated dynamic clusters, $\hat{\mathcal{I}}_{t_1:t_K}$, can be selected as:

$$\hat{\mathcal{I}}_{t_1:t_K} = \{\hat{\mathcal{S}}_{t_1:t_K} \in \hat{\mathcal{I}}^{init}_{t_1:t_K} \mid p(\mathbf{m} \mid \mathbf{z}_{t_1:t_K}) > \tau_p\}, \tag{19}$$

where the posterior probability is converted by $p(x) = \frac{e^{l(x)}}{1+e^{l(x)}}$; τ_p is a probability threshold to extract dynamic clusters. As shown in Fig. 4(b), *Instance Belief Update* accurately removes the truly dynamic objects in the initial dynamic instance proposal while preserving the misclassified false positives.

III. Experimental Evaluation

A. Experimental Setup

To analyze the performance of our framework, we consulted the static map construction benchmark proposed by Lim *et al.* [30]. We adopted the *preservation rate* (PR), *rejection rate* (RR) and F_1 score as evaluation metrics, which are voxel-wise and insensitive to the size of voxelization. The evaluation metrics are defined as follows:

- $\text{PR} = \frac{\#\text{ of preserved static points by the static map}}{\#\text{ of total static points on the raw map}}$
- $\text{RR} = 1 - \frac{\#\text{ of preserved dynamic points by the static map}}{\#\text{ of total dynamic points on the raw map}}$
- $\text{F}_1 = 2 \cdot \text{PR} \cdot \text{RR} / (\text{PR} + \text{RR})$

We chose the SemanticKITTI [39], [40] as the benchmark dataset, which provides point-wise manual label in urban environment. For experimental evaluation, we compared our algorithm with state-of-the-art approaches, including OctoMap (voxel size: $0.2m$) [22], Removert (3 remove stages) [27], ERASOR [30] and DynamicFilter (online approach) [31]. In addition, we recorded a dataset in campus environment containing vehicles and pedestrians to validate the effectiveness of our approach in the real-world.

B. Evaluation

In the quantitative experiment based on the SemanticKITTI dataset, we set the parameters as follows: $K = 4$, $p_{inc} = 0.7$, $p_{dec} = 0.4$, $\tau_\rho = 0.8$, $\tau_p = 0.5$ and the voxel size for evaluation metrics is $0.2m$.

TABLE I

Comparison with the state-of-the-art methods on the SemanticKITTI dataset with $\tau_s = 0.6$.

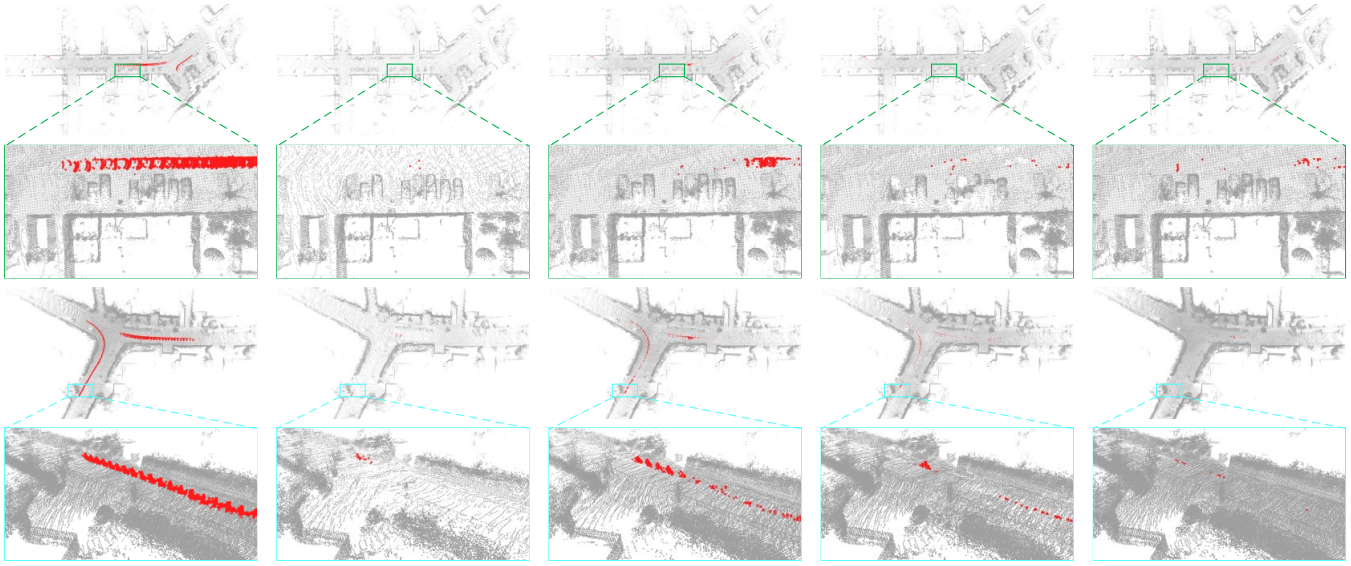
Seq.	Method	PR [%]	RR [%]	F_1 score
00	OctoMap-0.2 [22]	34.568	**99.979**	0.514
	Removert-RM3 [27]	85.502	99.354	0.919
	ERASOR [30]	93.980	97.081	0.955
	DynamicFilter* [31]	90.070	91.090	0.906
	DynaInsRemover(IOC)*	97.973	97.540	**0.977**
	DynaInsRemover(Proposed)*	**98.067**	96.679	0.974
01	OctoMap-0.2 [22]	20.777	**99.863**	0.344
	Removert-RM3 [27]	94.221	93.608	**0.939**
	ERASOR [30]	91.487	95.383	0.934
	DynamicFilter* [31]	87.950	87.690	0.878
	DynaInsRemover(IOC)*	89.498	84.663	0.870
	DynaInsRemover(Proposed)*	**99.262**	87.188	0.928
02	OctoMap-0.2 [22]	23.746	**99.792**	0.384
	Removert-RM3 [27]	76.319	96.799	0.853
	ERASOR [30]	87.731	97.008	0.921
	DynamicFilter* [31]	88.020	86.100	0.871
	DynaInsRemover(IOC)*	99.015	98.672	0.988
	DynaInsRemover(Proposed)*	**99.085**	98.716	**0.989**
05	OctoMap-0.2 [22]	33.904	**99.882**	0.506
	Removert-RM3 [27]	86.900	87.880	0.874
	ERASOR [30]	88.730	98.262	0.933
	DynamicFilter* [31]	90.170	84.650	0.873
	DynaInsRemover(IOC)*	97.909	96.373	**0.971**
	DynaInsRemover(Proposed)*	**98.021**	96.015	0.970
07	OctoMap-0.2 [22]	38.183	**99.565**	0.552
	Removert-RM3 [27]	80.689	98.822	0.888
	ERASOR [30]	90.624	99.271	**0.948**
	DynamicFilter* [31]	87.940	86.800	0.874
	DynaInsRemover(IOC)*	92.084	95.187	0.936
	DynaInsRemover(Proposed)*	**92.165**	95.187	0.936

* means online algorithm.

1) Impact of the Occupancy Similarity Threshold: We first examined the impact of occupancy similarity threshold τ_s on PR and RR. As shown in Fig. 5, when increasing τ_s, PR is decreased, but RR is increased. Therefore, we set $\tau_s = 0.6$ to yield the most reasonable performance throughout the experiment.

2) Quantitative Evaluation: We compared our framework with the existing state-of-the-art approaches. Table I presents the results of the quantitative evaluation experiment and Fig.6 shows the static maps after filtering dynamic objects.

As presented in Table I, OctoMap achieves the highest RR i.e. removes dynamic objects with the best performance. However, affected by incidence angle ambiguity, it produces sparse static maps, which means that a large number of static points are misclassified, especially ground points. Removert balances PR and RR well and achieves satisfactory results on most sequences. However, as shown in Fig. 6 (c), when the motion of the dynamic object is parallel to that of the sensor, the occlusion produces false positives, leading to loss of performance. ERASOR is a visibility-free approach and applies region-wise dynamic object removal, which achieves good dynamic object removal performance. But it cannot handle the dynamic object under a static object, as shown in Fig. 6(d). Compared to the online DynamicFilter, our method removes most of the dynamic points and achieves comparable performance to the offline methods. Because our



(a) Raw map (b) OctoMap (c) Removert (d) ERASOR (e) Proposed

Fig. 6. Comparison of the static map results produced by the state-of-the-art methods and our proposed method on the SemanticKITTI dataset Seq. 00 and 02. The red points are dynamic points that are not filtered and the gray points are static.

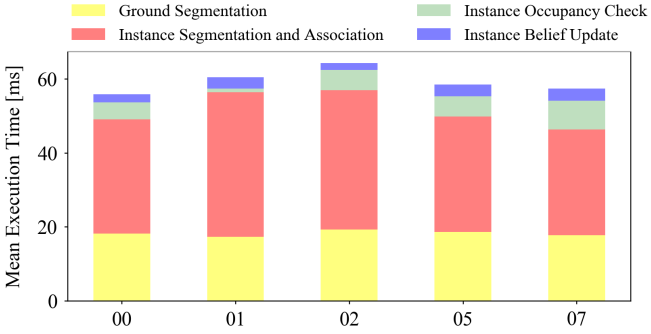
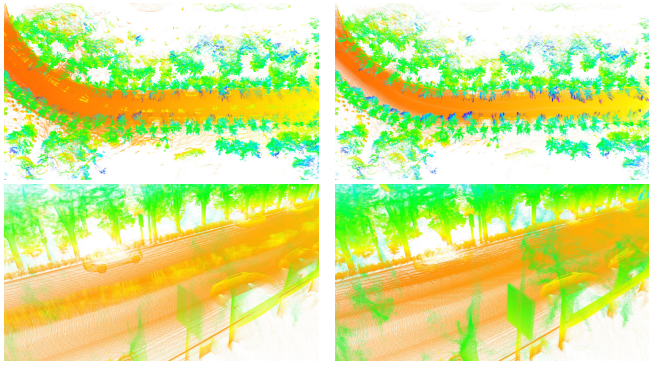


Fig. 7. The mean average execution times on the SemanticKITTI dataset, divided by algorithmic component.



(a) Raw map (b) After removal

Fig. 8. Our result in the campus experiment.

approach checks the instance-level geometric discrepancies and preserves as much detail in the map as possible.

3) Ablation Study: In ablation experiments, we shielded the *Instance Belief Update* to test its effectiveness. As shown in Table I, the *Instance Occupancy Check* (IOC) only approach achieved the highest F_1 score on Seq. 00 and 05. With the introduction of *Instance Belief Update*, PRs are all increased, although RRs are slightly lower.

C. Runtime of Our Approach

As our framework is developed for real-time static mapping, we measured the average runtime of each component of the proposed algorithm on the SemanticKITTI dataset with an AMD-5600X CPU only. From the Fig. 7, we can observe that our framework achieves the sum of average processing time around 60 ms, lessing than 100 ms for the LiDAR sensors to finish one scan.

D. Real world experiments

We also conducted a real-world experiment in campus using a wheel robot equipped with OS-64 LiDAR to validate the effectiveness of our approach. We use FAST-LIO [8] to estimate LiDAR motion trajectory. Fig. 8(a) shows the unprocessed raw map with ghost trail caused by pedestrians and vehicles. Fig. 8(b) displays the processed clean map. Our framework runs at $18Hz$ on the on-board PC with AMD-4800U CPU, which satisfies the real-time task requirements.

IV. Conclusion

In this paper, we present a novel dynamic instance-aware static map construction framework known as *DynaInsRemover*. Our method is based on instance geometric discrepancies to remove dynamic objects. In particular, we propose *Instance Occupancy Check* to identify potentially dynamic objects and *Instance Belief Update* to mitigate false positives. We quantitatively evaluate our approach on SemanticKITTI dataset, and the experimental results show that our method removes dynamic objects while preserving more details in static map compared to the state-of-the-art methods. For future work, we plan to utilize global information to deal with complex dynamic environments rather than just information within a temporal window.

REFERENCES

[1] H. Thomas, B. Agro, M. Gridseth, J. Zhang, and T. D. Barfoot, "Self-supervised learning of lidar segmentation for autonomous indoor navigation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 14 047–14 053.

[2] W. Ding, S. Hou, H. Gao, G. Wan, and S. Song, "Lidar inertial odometry aided robust lidar localization system in changing city scenes," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4322–4328.

[3] G. Kim and A. Kim, "Lt-mapper: A modular framework for lidar-based lifelong mapping," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 7995–8002.

[4] A. Wang, L. Wang, Y. Zhang, B. Hua, T. Li, Y. Liu, and D. Lin, "Landing site positioning and descent trajectory reconstruction of tianwen-1 on mars," *Astrodynamics*, vol. 6, pp. 69–79, 2022.

[5] R. Hu, X. Huang, and C. Xu, "Integrated visual navigation based on angles-only measurements for asteroid final landing phase," *Astrodynamics*, vol. 7, no. 1, pp. 69–82, 2023.

[6] H. Wei, W. Rao, G. Chen, G. Wang, X. Zou, Q. Li, and Y. Hu, "Tianwen-1 mars entry vehicle trajectory and atmosphere reconstruction preliminary analysis," *Astrodynamics*, vol. 6, pp. 81–91, 2022.

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

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

[9] K. Chen, R. Nemiroff, and B. T. Lopez, "Direct lidar-inertial odometry: Lightweight lio with continuous-time motion correction," *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[10] 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)*, 2020, pp. 10 765–10 771.

[11] X. Chen, S. Li, B. Mersch, L. Wiesmann, J. Gall, J. Behley, and C. Stachniss, "Moving object segmentation in 3d lidar data: A learning-based approach exploiting sequential data," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6529–6536, 2021.

[12] J. Kim, J. Woo, and S. Im, "Rvmos: Range-view moving object segmentation leveraged by semantic and motion features," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8044–8051, 2022.

[13] Z. He, X. Fan, Y. Peng, Z. Shen, J. Jiao, and M. Liu, "Empointmovseg: Sparse tensor-based moving-object segmentation in 3-d lidar point clouds for autonomous driving-embedded system," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 1, pp. 41–53, 2023.

[14] J. Sun, Y. Dai, X. Zhang, J. Xu, R. Ai, W. Gu, and X. Chen, "Efficient spatial-temporal information fusion for lidar-based 3d moving object segmentation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 11 456–11 463.

[15] S. Mohapatra, M. Hodaei, S. Yogamani, S. Milz, H. Gotzig, M. Simon, H. Rashed, and P. Maeder, "Limoseg: Real-time bird's eye view based lidar motion segmentation," *arXiv preprint arXiv:2111.04875*, 2021.

[16] P. Wu, S. Chen, and D. N. Metaxas, "Motionnet: Joint perception and motion prediction for autonomous driving based on bird's eye view maps," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 382–11 392.

[17] B. Mersch, X. Chen, I. Vizzo, L. Nunes, J. Behley, and C. Stachniss, "Receding moving object segmentation in 3d lidar data using sparse 4d convolutions," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7503–7510, 2022.

[18] B. Mersch, T. Guadagnino, X. Chen, I. Vizzo, J. Behley, and C. Stachniss, "Building volumetric beliefs for dynamic environments exploiting map-based moving object segmentation," *IEEE Robotics and Automation Letters*, 2023.

[19] N. Wang, C. Shi, R. Guo, H. Lu, Z. Zheng, and X. Chen, "Insmos: Instance-aware moving object segmentation in lidar data," *arXiv preprint arXiv:2303.03909*, 2023.

[20] Z. Song and B. Yang, "Ogc: Unsupervised 3d object segmentation from rigid dynamics of point clouds," *Advances in Neural Information Processing Systems*, vol. 35, pp. 30 798–30 812, 2022.

[21] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[22] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

[23] 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.

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

[25] L. Schmid, O. Andersson, A. Sulser, P. Pfreundschuh, and R. Siegwart, "Dynablox: Real-time detection of diverse dynamic objects in complex environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 10, pp. 6259–6266, 2023.

[26] 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.

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

[28] C. Qian, Z. Xiang, Z. Wu, and H. Sun, "Rf-lio: Removal-first tightly-coupled lidar inertial odometry in high dynamic environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 4421–4428.

[29] J. Park, Y. Cho, and Y.-S. Shin, "Nonparametric background model-based lidar slam in highly dynamic urban environments," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 190–24 205, 2022.

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

[31] T. Fan, B. Shen, H. Chen, W. Zhang, and J. Pan, "Dynamicfilter: an online dynamic objects removal framework for highly dynamic environments," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 7988–7994.

[32] H. Lim, L. Nunes, B. Mersch, X. Chen, J. Behley, H. Myung, and C. Stachniss, "Erasor2: Instance-aware robust 3d mapping of the static world in dynamic scenes," in *Robotics: Science and Systems (RSS 2023)*. IEEE, 2023.

[33] M. Oh, E. Jung, H. Lim, W. Song, S. Hu, E. M. Lee, J. Park, J. Kim, J. Lee, and H. Myung, "Travel: Traversable ground and above-ground object segmentation using graph representation of 3d lidar scans," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7255–7262, 2022.

[34] S. Cheng, M. Yao, and X. Xiao, "Dc-mot: Motion deblurring and compensation for multi-object tracking in uav videos," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 789–795.

[35] H. Lim, M. Oh, and H. Myung, "Patchwork: Concentric zone-based region-wise ground segmentation with ground likelihood estimation using a 3d lidar sensor," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6458–6465, 2021.

[36] S. Park, S. Wang, H. Lim, and U. Kang, "Curved-voxel clustering for accurate segmentation of 3d lidar point clouds with real-time performance," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6459–6464.

[37] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4802–4809.

[38] G. Kim, S. Choi, and A. Kim, "Scan context++: Structural place recognition robust to rotation and lateral variations in urban environments," *IEEE Transactions on Robotics*, vol. 38, no. 3, pp. 1856–1874, 2022.

[39] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[40] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9296–9306.