Automatic Labeling to Generate Training Data for Online LiDAR-Based Moving Object Segmentation

Xieyuanli Chen, Benedikt Mersch, *Graduate Student Member, IEEE*, Lucas Nunes, *Graduate Student Member, IEEE*, Rodrigo Marcuzzi, *Graduate Student Member, IEEE*, Ignacio Vizzo, *Student Member, IEEE*, Jens Behley, and Cyrill Stachniss, *Member, IEEE*

***Abstract*—Understanding the scene is key for autonomously navigating vehicles, and the ability to segment the surroundings online into moving and non-moving objects is a central ingredient of this task. Often, deep learning-based methods are used to perform moving object segmentation (MOS). The performance of these networks, however, strongly depends on the diversity and amount of *labeled* training data—information that may be costly to obtain. In this letter, we propose an automatic data labeling pipeline for 3D LiDAR data to save the extensive manual labeling effort and to improve the performance of existing learning-based MOS systems by automatically annotation training data. Our proposed approach achieves this by processing the data offline in batches, i.e., it is not designed to run online on a vehicle. It labels the actually moving objects such as driving cars and pedestrians as moving. In contrast, the non-moving objects, e.g., parked cars, lamps, roads, or buildings, are labeled as static. We show that this approach allows us to label LiDAR data highly effectively and compare our results to those of other label generation methods. We also train a deep neural network with our automatically generated labels and achieve comparable performance to the one trained with manual labels on the same data—and an even better performance when using additional datasets with labels generated by our approach. Furthermore, we evaluate our method on multiple datasets using different sensors, and our experiments indicate that our method can generate labels in different outdoor environments.**

***Index Terms*—Deep learning methods, object detection, segmentation and categorization, semantic scene understanding.**

I. Introduction

MOVING object segmentation (MOS) is a fundamental task for autonomous vehicles as it separates the *actually moving* objects such as driving cars and pedestrians from static or non-moving objects such as buildings, parked cars, etc. This is an important processing step needed in many applications, such as pose estimation [1], collision avoidance [2], or robot path planning [3]. This knowledge can also robustify and improve future surroundings prediction [4], [5] and simultaneous localization and mapping (SLAM) [6]. Thus, accurate and reliable MOS available at frame rate is relevant for most autonomous mobile systems.

3D LiDAR-based moving object segmentation (LiDAR-MOS) is challenging due to the distance-dependent sparsity and uneven distribution of the range measurements. Our recent work [6] proposes a deep neural network, called LMNet, to achieve LiDAR-MOS faster than sensor frame rate by exploiting sequential LiDAR range images. Such supervised learning-based methods rely on manually labeled data, which is often limited in size and cannot easily be generated for new or unseen environments. Despite a large amount of publicly available LiDAR datasets nowadays [7]–[9], labeling LiDAR data is still a tedious process and one of the bottlenecks in supervised learning. Automatic label generation can alleviate this problem by exploiting the temporal-spatial relations in the dataset and, compared to online operation, by processing the data in batches. For example, the labels at a specific timestamp can be easier determined when exploiting preceding *and* succeeding scans compared to online MOS.

The main contribution of this letter is a novel modularized approach to generate MOS labels from 3D LiDAR scans automatically. Given a sequence of LiDAR scans, our approach first exploits a SLAM method to estimate the vehicle poses. It then applies dynamic object removal techniques to detect possible dynamic objects coarsely. After that, we cluster the proposals into instances and track them using a Kalman filter. Based on the tracked trajectories, we label the actually moving objects (driving cars, pedestrians, etc.) as moving. In contrast, the non-moving objects, including parked cars, are labeled as static, as shown in Fig. 1. Using the labels automatically generated offline, we train a LiDAR-MOS network for online detection. Note that no manually labeled data or other sensor information is needed for this form of training. Once having a sequence of LiDAR scans, our method can automatically generate MOS labels.

For evaluation, we automatically label LiDAR scans in the training sequences of the KITTI odometry dataset [9] using different methods and compare the quality of generated MOS labels using the LiDAR-MOS labels from SemanticKITTI [6],

Manuscript received December 16, 2021; accepted April 5, 2022. Date of publication April 12, 2022; date of current version April 21, 2022. This letter was recommended for publication by Associate Editor J. R. Martinez-de Dios and Editor M. Vincze upon evaluation of the reviewers' comments. This work was supported in part by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) through Germany's Excellence Strategy under Grant EXC-2070 - 390732324 - PhenoRob, in part by EC through Horizon Europe under Grant Agreement 101017008 (Harmony), and in part by Chinese Scholarship Committee. *(Corresponding author: Xieyuanli Chen.)*

Xieyuanli Chen, Benedikt Mersch, Lucas Nunes, Rodrigo Marcuzzi, Ignacio Vizzo, and Jens Behley are with the University of Bonn, 53115 Bonn, Germany (e-mail: xieyuanli.chen@igg.uni-bonn.de; mersch@igg.uni-bonn.de; lucas.nunes@uni-bonn.de; rodrigomarcuzzi@gmail.com; ivizzo@uni-bonn.de; jens.behley@igg.uni-bonn.de).

Cyrill Stachniss is with the University of Bonn, 53115 Bonn, Germany, and also with the Department of Engineering Science, University of Oxford, Oxford OX1 2JD, U.K. (e-mail: cyrill.stachniss@igg.uni-bonn.de).

Digital Object Identifier 10.1109/LRA.2022.3166544

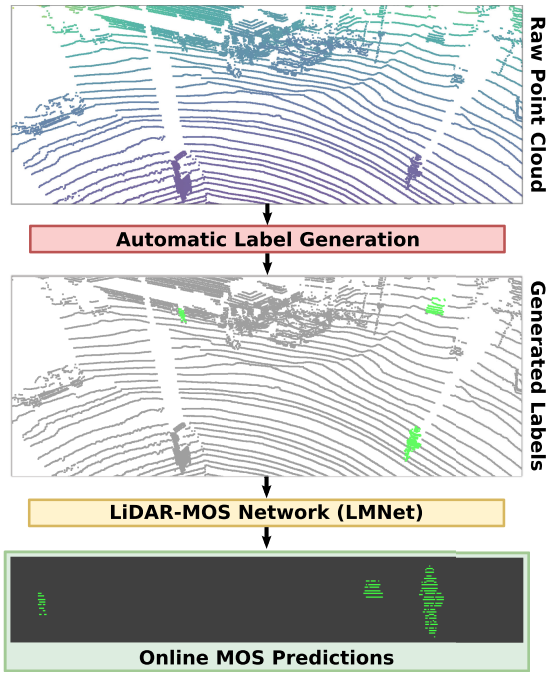


Fig. 1. Moving object segmentation trained with the labels automatically generated by our approach. Our method can detect and segment the moving objects given sequential point cloud data and generate labels for LiDAR-based moving object segmentation. Based on the automatically generated labels, we re-train our LiDAR-MOS network, which can be deployed on an autonomous vehicle to perform LiDAR-MOS online in unseen environments.

[10]. Next, we automatically generate more labels on additional data from the KITTI road dataset, to train LMNet further. Compared to the network trained on manually labeled ground truth data, the evaluation results suggest that the network trained on labels automatically generated by our method achieves a similar performance, and is superior when using more automatically generated data from additional scans. Furthermore, our approach generalizes well to different, unseen environments, which we show for MOS on three different datasets.

In sum, we make the following claims: i) Compared to existing methods, our approach generates better labels for LiDAR-MOS. ii) Based on our generated labels, a network achieves a similar performance compared to the same network trained with manual labels on the same data and better performance with additional automatically labeled training data. iii) Our method generates effective labels for different LiDAR scanners and in different environments. We released our code at.[1]

II. Related Work

With the advent of deep neural networks, the task of moving object segmentation has achieved great success for image and video-based approaches [11], [12]. However, it is still challenging for the LiDAR-based approach due to the sparsity of the sensor data and the lack of publicly available large-scale datasets with point-wise moving and static labels. Here, we concentrate on LiDAR-based MOS and methods that can generate LiDAR-MOS labels.

[1][Online]. Available: https://github.com/PRBonn/auto-mos

There are both geometric model-based and deep neural network-based methods to address the problem of *online* LiDAR-MOS. For example, Yoon *et al.* [13] detect dynamic objects in LiDAR scans exploiting geometric heuristics, e.g., the residual between LiDAR scans, free space checking, and region growing to find moving objects. Dewan *et al.* [14] propose a LiDAR-based scene flow method that estimates motion vectors for rigid bodies. Such geometric-based methods need no training but occasionally result in incomplete or inaccurate detection of moving objects.

Besides geometry-based approaches, there are also deep network-based methods [15]–[17], which use generic end-to-end trainable models to learn local and global statistical relationships directly from data. Such scene flow methods usually estimate motion vectors between two consecutive scans, which may not differentiate between slowly moving objects and sensor noise. Semantic segmentation can be seen as a relevant step towards moving object segmentation. Based on the online semantic segmentation [18]–[20], SuMa++ [1] exploits semantics to detect and filter out dynamic objects to improve the LiDAR SLAM performance. There are also several 3D point cloud-based semantic segmentation approaches [21], [22], which exploit sequential point clouds and predict moving objects. However, these methods require a large number of semantic labels, which are not always available and make it difficult to train and generalize to unseen data. Recently, we proposed a deep learning-based LiDAR-MOS approach [6] exploiting short sequences of range images from a rotating 3D LiDAR sensor, which achieves accurate online performance on a novel LiDAR-MOS benchmark. Despite good performance and simplified binary labels, it still relies on manual labels and does not generalize well to different environments with varying appearances.

Unlike online LiDAR-MOS, label generation can be done *offline*, which enables better performance in perception tasks, using sequential future and past observations. Multiple works have already been proposed to clean the point cloud map, which removes dynamic objects during the mapping procedure, resulting in a clean static map [23]–[26]. For example, Kim *et al.* [24] proposed a range image-based method, which exploits the consistency check between the query scan and the pre-built map to remove dynamic points. Pagad *et al.* [26] proposed an occupancy map-based method to remove dynamic points in LiDAR scans. They first build occupancy maps using object detection and then use a voxel traversal method to remove the moving objects. Recently, Lim *et al.* [25] proposed a method to first remove dynamic objects by checking the occupancy of each sector of LiDAR scans and then revert the ground plane by region growing. In contrast, Arora *et al.* [23] first detect the ground plane and then remove the "ghost effect" caused by the moving object during mapping. Even though map cleaning methods can distinguish unstable points from the static map, they usually remove not only dynamic objects but also static parts due to noisy poses or occlusions. Since the objective of map cleaning is different from MOS, it may not influence the mapping results when removing more points that belong to static objects. In contrast, MOS accurately separates moving and static objects in each LiDAR scan.

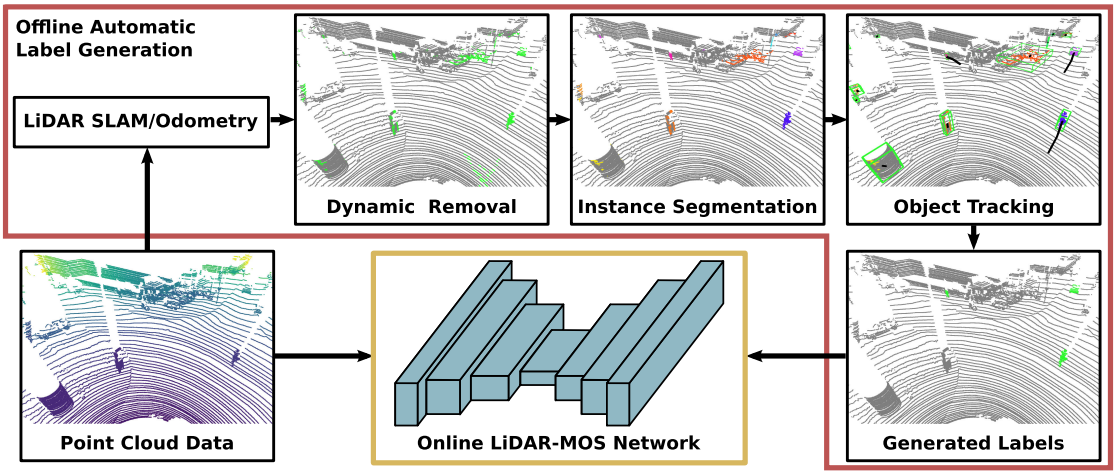


Fig. 2. Overview of our method. It uses sequential LiDAR scans as input, and first conducts a LiDAR odometry / SLAM step to estimate the poses. With the estimated poses, it then applies a map cleaning method to coarsely detect the moving objects (colored in green). A clustering method is then used to extract instances (in different colors) based on the detected moving object proposals. After that, a multi-object tracking method is applied to associate instances (with bounding boxes) and decide the final labels of instances based on the tracked trajectories (colored in black). With the generated labels, we train LMNet that can be later deployed for online LiDAR-MOS.

The approach closest to our work is the one by Pfreundschuh *et al.* [27]. It generates labels automatically to train a network for moving object-aware SLAM working in small areas, e.g., a hall or a train station. Different from that, our approach does not require ground truth poses and exploits tracking to distinguish between moving and non-moving objects.

III. Our Approach

We are interested in automatically labeling LiDAR points offline with two classes, *static* or *dynamic*, that can be later used for training a network for online MOS. Our approach consists of five serialized modules as shown in Fig. 2. Our proposed method only uses sequential LiDAR scans as input, and first uses a LiDAR odometry / SLAM approach to estimate the poses for each scan (see Section III-A). With the estimated poses, we apply a map cleaning method to coarsely detect moving objects (see Section III-B). We then use a clustering method to extract instances based on the detected moving object proposals (see Section III-C). After that, we apply a multi-object tracking method to associate instances and determine the final labels of instances based on the tracked trajectories (see Section III-D). Once we have generated the labels offline, we train LMNet that can be later deployed on an autonomous vehicle to perform LiDAR-MOS online in an unseen environment (see Section III-E). Note that we can easily replace the individual methods for each step due to the highly modularized nature of our proposed framework. We will now discuss the individual steps of our approach.

A. LiDAR Odometry

First, we estimate the poses of each scan in order to get globally aligned point clouds in the next step. Instead of requiring given ground-truth poses or information from other sensors, such as RTK-GPS, our method uses only sequential LiDAR scans as input. In the following, we denote the estimated absolute pose of a scan at time t by $\mathsf{T}_t \in \mathbb{R}^{4\times4}$. Note that there might be noise in the estimated poses, which may influence the performance of visibility or ray tracing-based methods [24], [27]. With increasing noise, more objects will be detected as moving due to the misalignment caused by inaccurate poses. However, this does not affect our method, since we use the estimated poses for map cleaning and generate only coarse dynamic object proposals, which are later verified and typical false positives are filtered out via tracking. We use the off-the-shelf SLAM approach, SuMa [28], to estimate the pose of each LiDAR scan. SuMa exploits a spherical projection of the point cloud and estimates the relative pose between the current LiDAR scan and the maintained world model via projective ICP.

B. Coarse Dynamic Object Removal

Instead of using semantic information to pre-define movable objects [1], e.g., cars, pedestrians, our method only exploits the sequential temporal-spatial information to coarsely detect the moving objects in a class-agnostic manner. Since the sequential LiDAR data is available for offline label generation, we can detect dynamic objects with an aggregated map, which is also referred as map cleaning. We first aggregate all local LiDAR points to obtain a single point cloud map $\mathcal{M}$:

$$\mathcal{M} = \bigcup_{t\in\mathcal{T}} \{\mathsf{T}_t\mathbf{p} | \mathbf{p} \in \mathcal{P}_t\}, \tag{1}$$

where $\mathcal{T} = \{1, 2, \ldots, N\}$ is the set of timestamps and N is the number of scans. $\mathcal{P}_t = \{\mathbf{p}_j\}_{j=1}^{N_t}$ is the scan at time t with N_t points. The transformations T_t are the aforementioned estimated poses from the SLAM method.

Let $\hat{\mathcal{M}}$ be the estimated static map. The problem of map cleaning can be defined as follows:

$$\hat{\mathcal{M}} = \mathcal{M} - \bigcup_{t\in\mathcal{T}} \hat{\mathcal{M}}_t^{\text{dyn}}, \tag{2}$$

where $\hat{\mathcal{M}}_t^{\text{dyn}}$ refers to the estimated dynamic points at timestamp t. In this work, we are interested in the set $\hat{\mathcal{M}}_t^{\text{dyn}}$ of moving objects instead of the static map $\hat{\mathcal{M}}$.

Even though such map cleaning methods can distinguish moving objects from the static map, we observe a substantial number of false positive detections, possibly caused by noisy points or inaccuracies in the estimated poses. Since the goal of map cleaning is to obtain a static map, it is reasonable for such methods to be more conservative and remove some more points to guarantee a clean map. Moreover, exploiting both, past and future data sequentially, there are also redundant observations. Thus, it may not influence the mapping results when cleaning methods remove more points even though they belong to static objects. However, the objective of MOS training data generation is different from that of map cleaning. Here, we aim to accurately separate actual moving objects, e.g., driving cars, from static or non-moving objects, e.g., buildings or parked cars, in every LiDAR scan.

To show that our framework can employ different methods, we test three map cleaning methods, including ERASOR [25], Removert [24], and an Octomap-based method [23]. All these variants of our method achieve good performance in MOS label generation as shown in Section IV-B. Among them, ERASOR [25] achieves the best performance. We refer to their original letters for more details of these map cleaning methods [23]–[25].

C. Class-Agnostic Instance Segmentation

To compute accurate segments of moving objects, we apply instance segmentation on the dynamic proposals $\hat{\mathcal{M}}_{\text{dyn},t}$ provided by the map cleaning method. The goal of a segmentation $\mathcal{S}$ is to partition the point cloud into disjoint subsets:

$$\mathcal{S} = \bigcup_{k \in \{1,\dots,N_S\}} \mathcal{S}_k, \tag{3}$$

where $\mathcal{S}_k \subset \hat{\mathcal{M}}_{\text{dyn},t}$ is a segment, and N_S denotes the number of segments. Every point $\mathbf{p} \in \hat{\mathcal{M}}_{\text{dyn},t}$ exists in one and only one segment $\mathcal{S}_k$, i.e., $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset, \forall\, i \neq j$.

Once we obtain the final clustering results using an off-the-shelf class-agnostic segmentation approach, we generate a bounding box $\boldsymbol{b}_k = (\boldsymbol{c}_k, \theta, l, w, h, s)$ for each segment $\mathcal{S}_k$, including the center coordinates $\boldsymbol{c}_k \in \mathbb{R}^3$, the length l, width w, and height h, heading angle θ, and confidence score s. We filter out segments that are too small or too large, i.e., segments with less than N_{min} points or with a maximum side length of the bounding box larger than a threshold T_{size} and get the final set of instances $\mathcal{B} = \{\boldsymbol{b}_m\}_{m=1}^{N_B}$ with $N_B \leq N_S$. There are many methods for class-agnostic segmentation [29]–[31]. In this letter, we choose HDBSCAN [31], which is an extension of DBSCAN [32]. DBSCAN is density-based and provides a clustering hierarchy from which a simplified tree of significant clusters can be constructed. HDBSCAN performs DBSCAN over varying density thresholds ϵ and integrates the result yielding a clustering that gives the best score over ϵ. This allows HDBSCAN to find clusters of varying densities (unlike DBSCAN), and to be more robust to parameter selection.

D. Multiple Dynamic Object Tracking

After clustering, there can still be static objects inside the set of instances $\mathcal{B}$. To verify the real dynamic objects, we use a multi-object tracking method to obtain trajectories of object instances and determine the label for each tracked instance based on its movement.

To this end, we use multiple extended Kalman filters to track instance bounding boxes as proposed by Weng *et al.* [33]. For the motion model, we first compensate the ego-motion using the poses estimated by the SLAM system. Then, for each instance, we apply a constant velocity model as the state prediction. For the observation model, we find associations between instances in consecutive scans. We compute a cost matrix $\boldsymbol{C} \in \mathbb{R}^{N_B^t \times N_B^{t-1}}$ between all N_B^t instances in the LiDAR scan at timestamp t and all N_B^{t-1} instances still tracked in the previous scan at timestamp $t-1$, by means of the similarity. The association problem can be formulated as a bipartite graph matching problem that can be solved using the Hungarian method [34], an optimal assignment method, to determine the pairs of associations between currently detected and previously tracked instances.

To compute the instance similarity, we take three different geometric features into account, the center distance c_d, the overlap between bounding box volumes c_o, and the change of the volume between each pair of instances based on their bounding boxes c_v:

$$c_d(i,j) = \|\boldsymbol{c}_i - \boldsymbol{c}_j\|_2, \tag{4}$$

$$c_o(i,j) = 1 - \text{IoU}(\boldsymbol{b}_i, \boldsymbol{b}_j), \tag{5}$$

$$c_v(i,j) = 1 - \frac{\min(\boldsymbol{v}_i, \boldsymbol{v}_j)}{\max(\boldsymbol{v}_i, \boldsymbol{v}_j)}, \tag{6}$$

where the center of the bounding box $\boldsymbol{b}_k$ is denoted as $\boldsymbol{c}_k$ and $\|\cdot\|_2$ is the Euclidean distance. The intersection over union $\text{IoU}(\cdot)$ between two instance bounding boxes is used to account for the volume overlap with $\boldsymbol{v}_k = l \times w \times h$ representing the volume of the bounding box $\boldsymbol{b}_k$.

Each entry of the cost matrix $\boldsymbol{C}$ is calculated as the association cost by a linear combination of these three features between a new instance i and the prediction of a previous tracked instance j:

$$\boldsymbol{C}_{i,j} = \alpha_d\, c_d(i,j) + \alpha_o\, c_o(i,j) + \alpha_v\, c_v(i,j), \tag{7}$$

where $\alpha_d, \alpha_o, \alpha_v \in \mathbb{R}$ are importance weights.

There are multiple challenging cases during tracking. For example, newly detected objects might not be tracked in the previous scans, a tracked instance might leave the scene, or a tracked instance might be occluded or missed. To avoid wrong associations, we add a new EKF in case of newly detected objects by determining Flag_{add}:

$$\text{Flag}_{\text{add}} = (c_d > T_d) \vee (c_o > T_o) \vee (c_v > T_v), \tag{8}$$

where T_d, T_o, T_v are the thresholds for the three instance features respectively determined based on the validation data. We store the deactivated trajectories for a fixed number of timestamps n_{old}. We associate or re-identify instances if the similarity between a newly detected object and the still tracked

or deactivated one is high, resulting in $\text{Flag}_{\text{add}} = false$. For re-identification, the motion model is applied continuously also to the deactivated targets (not matched with any newly detected instance) to estimate their position in case of occlusions or missed detections.

After tracking, we label an instance as moving if its accumulated trajectory is larger than its maximum side length of the associated bounding box. We label the instance point-wise, which means all points inside the bounding box will be labeled as moving once we determine the instance as moving. In case of misdetection or occlusion, we can still generate temporally consistent labels by re-identifying instances using the EKFs.

E. Training a Neural Network for Online LiDAR-MOS

After running our pipeline, we obtain binary labels for all points. Based on that, we can train a deep learning-based MOS network, which can be later deployed for online LiDAR-MOS in unseen environments. Instead of using the manual labels provided by the LiDAR-MOS benchmark [6], we train the network with the automatically generated labels by our proposed offline pipeline. Since our method can generate labels automatically for sequential LiDAR data, we can train with more data than that provided by the LiDAR-MOS benchmark, which further boosts the performance of our network for online LiDAR-MOS. Furthermore, the proposed auto-labeling method also generates labels for LiDAR data from other datasets, which were collected in different environments. Using these additional generated labels, the deep learning-based MOS network can better generalize to different environments.

In this letter, we train the state-of-the-art LiDAR-MOS network LMNet [6]. LMNet exploits sequential range images from a LiDAR sensor as an intermediate representation combined with a typical encoder-decoder CNN and runs faster than the frame rate of the sensor. By using residual images, the network obtains temporal information and can differentiate between moving and static objects. For more details, we refer to the LMNet letter [6].

F. Parameters and Implementation Details

Our method consists of individual modules that are carried out sequentially. We use SuMa [28] with the default parameters as the LiDAR SLAM approach. We then use ERASOR [25] with the default parameters to coarsely detect the dynamic objects. Based on the dynamic object proposals, we apply HDBSCAN [31] with multiple density thresholds $\epsilon = \{2.0, 1.0, 0.5, 0.25\}$ to generate dynamic object instances. After filtering out instances with less than $N_{\text{min}} = 5$ points or with a maximum side length larger than $T_{\text{size}} = 20\,\text{m}$, we track the remaining instances using a multi-object tracking method [33]. We use the same importance weights for three different association terms, i.e., $\alpha_d = \alpha_o = \alpha_v = 1$, and keep track of deactivated instances for $n_{\text{old}} = 5$ timestamps. The thresholds are $T_d = 2\,\text{m}$, $T_o = 0.95$, and $T_v = 0.7$. For training LMNet [6], we use the default parameters with 8 residual images without semantic information and train for 150 epochs.

IV. Experimental Evaluation

We present our experiments to illustrate the capabilities of our method. They furthermore support our key claims, that: i) Our approach generates better labels for moving object segmentation using only 3D LiDAR scans; ii) Based on our automatically generated labels, the network achieves a similar performance in LiDAR-MOS compared to the one trained with manual labels, and a better performance with additional automatically generated training data; iii) Our method generates LiDAR-MOS labels for different LiDAR data collected from different environments.

A. Experimental Setup

For quantifying the LiDAR-MOS performance, we use the intersection-over-union (IoU) metric over moving objects, which is given by

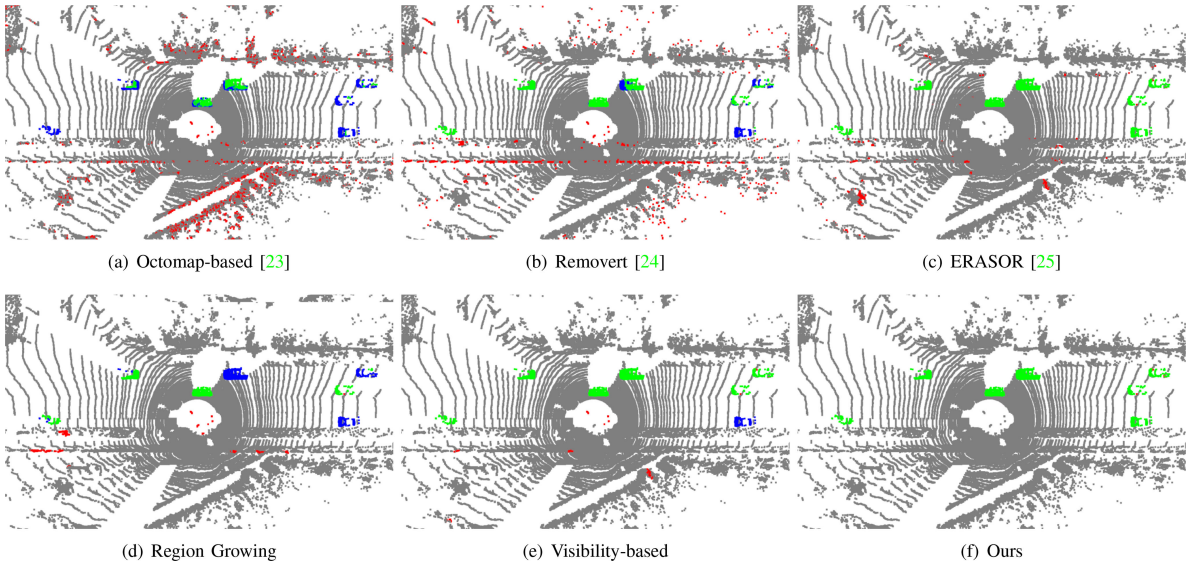
$$\text{IoU}_{\text{MOS}} = \frac{\text{TD}}{\text{TD} + \text{FD} + \text{FS}}, \tag{9}$$

where TD, FD, and FS correspond to the number of true dynamics, false dynamics, and false statics points.

We evaluate our method on four different datasets. The first one is the LiDAR-MOS benchmark [6] of SemanticKITTI [7], which separates all classes into moving and static. It contains 22 sequences, from 00 to 07 and 09 to 10 for training with ground truth labels, 08 for validation, and 11 to 21 for testing. In this work, we first evaluate the automatic label generation methods on the training data in Section IV-B. After generating the labels automatically, we re-train the LMNet with the generated labels and test the re-trained model on the hidden test set. Since there is more unlabeled LiDAR data available in the original KITTI dataset [9], we generate more labels automatically and train the network with extra data. We also test the further trained model on the hidden test set, see Section IV-C. To evaluate the generalization ability of the proposed method, we also test our approach on three other datasets, Apollo [35], MulRan [36], and IPB-Car [37], [38], shown in Section IV-D. KITTI and Apollo have been recorded with Velodyne-64 LiDAR scanners, while IPB-Car and MulRan offer LiDAR data from Ouster sensors.

B. Automatic Data Labeling Results

The experiment in this section supports our claim that our approach generates better labels for moving object segmentation using only 3D LiDAR scans than baseline methods. We test multiple methods that can distinguish moving and non-moving objects on 3D LiDAR data, as shown in Table I. We compare our method to map cleaning-based methods like Removert [24], ERASOR [25], and an Octomap-based [23] method. Those methods are open-source and have been evaluated on the KITTI dataset. Therefore, we directly use the provided implementation with the default parameters. We also re-implement two other methods whose source code is not publicly available. The first one is inspired by Yoon *et al.* [13] and detects dynamic objects in LiDAR scans exploiting geometric heuristics, including residuals and region growing, which we denote as "Region Growing". The other one is inspired by the label generation method of Pfreundschuh *et al.* [27], which uses range images for visibility



(a) Octomap-based [23] (b) Removert [24] (c) ERASOR [25]

(d) Region Growing (e) Visibility-based (f) Ours

Fig. 3. Qualitative results of (a) Octomap-based [23], (b) Removert [24], (c) ERASOR [25], (d) Region growing-based method inspired by [13], (e) Visibility-based method inspired by [27], and (f) ours. The green points represent the true dynamics (TD), red points are false dynamics (FD), blue points are false statics (FS), and the gray background are true statics (TS).

TABLE I
EVALUATION ON AUTOMATIC LABEL GENERATION

Methods	Precision	Recall	IoU_{MOS}
Octomap-based [23]	14.4	69.9	13.6
Removert [24]	18.4	55.4	16.0
ERASOR [25]	20.0	81.6	19.1
Region Growing	17.6	54.1	15.4
Visibility-based	76.8	41.2	36.6
Ours (Octomap)	77.4	70.8	58.7
Ours (Removert)	79.4	62.1	53.5
Ours	**88.1**	**82.5**	**74.2**

checking and later uses clustering to verify candidates by voting. We implemented this method using the range image-based visibility checking part from Removert, the HDBSCAN for clustering, and the same voting used in the original letter. In the following, we refer to this approach as "Visibility-based". For more details, we refer to the original letters [13], [27].

We show the results of our method with different setups. We denote our method using Octomap as "Ours (Octomap)," using Removert as "Ours (Removert)," and our best performance with ERASOR as "Ours". All methods only use geometric information without any learning or semantic information, which makes it possible to generate labels fully unsupervised.

As shown in Table I, our method outperforms other baseline methods on generating LiDAR-MOS labels in terms of IoU over moving objects, which is in line with the qualitative results shown in Fig. 3. As can be seen, the map-cleaning methods, Octomap-based, Removert, and ERASOR, show many false dynamics (colored in red). This also leads to a high recall of moving objects but to a low precision. For other label generation methods, "Region Growing" and "Visibility-based," they often miss moving objects leading to many false static points (colored in blue), which results in a low recall. In comparison, all variants

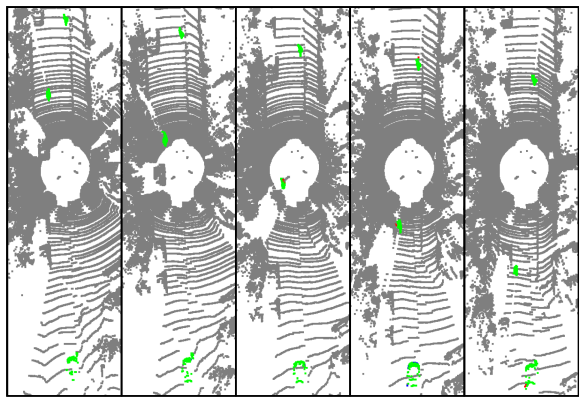


Fig. 4. Qualitative results on consecutive scans. Our method detects different types of moving objects, i.e., vehicles and cyclists, consistently across a sequence of scans (green=TD, red=FD, blue=FS, gray=TS).

of our method outperform the baseline methods in terms of MOS IoU, which shows the effectiveness of the proposed framework. Our best setup using ERASOR outperforms others in terms of all metrics.

We further provide qualitative results of our method on consecutive scans in Fig. 4 . As can be seen, our method detects different types of moving objects, i.e., vehicles and cyclists, consistently across a sequence of scans.

C. MOS Performance Using Auto-Generated Labels

In the second experiment, we re-train LMNet with the automatically generated labels and evaluate the model on the hidden test set of the LiDAR-MOS benchmark, as shown in Table II. We compare the network trained on manual ground truth labels (LMNet+Manual), with the network re-trained on

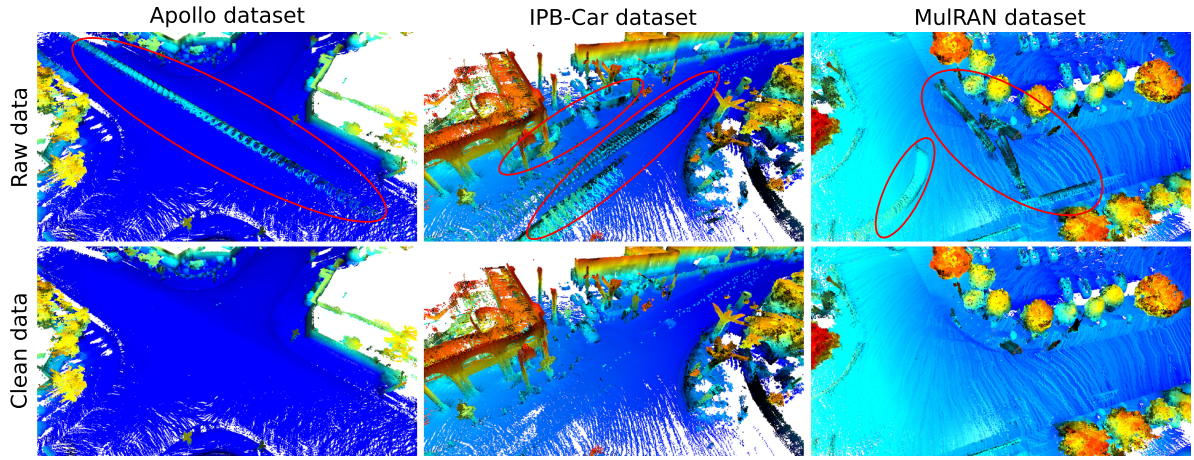


Fig. 5. Map cleaning results on three different datasets. The color from blue to red represents the height from low to high. Artifacts caused by moving objects are effectively removed by our method (red circles).

TABLE II
ONLINE LIDAR-MOS PERFORMANCE ON BENCHMARK

Methods	IoU_{MOS}
SalsaNext [19] (movable classes)	4.4
SceneFlow [17]	4.8
SqSequence [21]	43.2
KPConv [22]	60.9
LMNet+Manual	58.3
LMNet+Ours	54.3
LMNet+Ours+Extra	**62.3**

TABLE III
ONLINE LIDAR-MOS PERFORMANCE ON APOLLO

Methods	IoU_{MOS}
LMNet+Manual (trained on KITTI)	16.9
LMNet+Ours	45.7
LMNet+Ours+Fine-Tuned	**65.9**

the labels generated by the proposed method (LMNet+Ours). As can be seen, LMNet+Ours achieves a similar IoU on the benchmark as LMNet+Manual. We also show the results of LMNet re-trained with extra automatically generated labels (LMNet+Ours+Extra). In this experiment, we use the road sequences of KITTI raw dataset [9], generate additional labels using the proposed method, and re-train LMNet with labels automatically generated on both KITTI odometry and road data. Boosted with extra data, we see that LMNet+Ours+Extra outperforms LMNet+Manual trained with manual labels, and is the best performing strategy.

We additionally compare our method to other online LiDAR-MOS methods that only use the current and past observations as required for real-world self-driving applications. For example, one uses all movable objects as dynamic depending on the semantics from a semantic segmentation network, such as SalsaNext [19], or multi-scan networks to learn to distinguish moving and non-moving object classes, such as SqSequence [21] or KPConv [22], or using the flow vectors with a threshold to determine the moving objects, such as SceneFlow [17]. Our method also outperforms all the baseline methods.

D. Generalization Capabilities

The third experiment investigates the generalization capabilities of our approach, and it supports our last claim that our method can generate LiDAR-MOS labels for different LiDAR data collected from different environments. To this end, we test our method on three more datasets, Apollo [35], MulRan [36], and IPB-Car [37], [38]. To quantitatively evaluate the generalization capabilities of the proposed method, we manually label a small test set on the Apollo-ColumbiaPark-MapData [35], sequence 2 frame 22300–24300 and sequence 3 frame 3100–3600, which contain many dynamic objects. We use the same LMNet and compare three different setups: the pre-trained model with SemanticKITTI LiDAR-MOS manual labels (LMNet+Manual trained on KITTI), the re-trained model with automatically generated labels on Apollo-ColumbiaPark-MapData sequence 1 frame 6500–7500 and sequence 4 frame 1500–3100 (LMNet+Ours), and the pre-trained model with automatically generated labels on KITTI fine-tuned with automatically generated labels on the Apollo dataset (LMNet+Ours+Tuned).

As shown in Table III, the model pre-trained with SemanticKITTI LiDAR-MOS manual labels cannot generalize very well to different environments. The re-trained model with our approach achieves a better performance, even when using only a small set of training data, in this case only 2600 LiDAR scans. If we fine-tune the pre-trained model with our automatically generated labels, i.e., using automatic generated labels from both KITTI and Apollo, it performs best with only 10 fine-tuning epochs.

We also test our method on the MulRan and IPB-Car datasets. Since there are no manual labels available, we only show qualitative mapping results of Apollo, IPB-Car, and MulRan datasets in Fig. 5. As can be seen, using the automatically generated labels, we can effectively remove dynamics during mapping (shown in the upper row) and obtain comparably clean maps (shown in the bottom row).

V. Conclusion

In this letter, we presented a novel approach to automatically generate labels on point clouds for moving object segmentation in 3D LiDAR data. Our method combines LiDAR odometry, map cleaning, instance segmentation, and multi-object tracking to determine labels for moving and non-moving objects in LiDAR data offline. Based on the labels generated automatically by our approach, we re-train a network to achieve an efficient online LiDAR-MOS system. We provided comparisons to other existing techniques and supported all our claims made in this letter. The experiments suggest that our method achieves a solid performance on LiDAR-MOS label generation and substantially boosts the online performance by additional automatically generating labels. We evaluated our method on four different datasets, showing strong generalization capabilities for our approach.

References

[1] X. Chen, A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss, "SuMa++: Efficient LiDAR-based semantic SLAM," in *Proc. IEEE/RSJ Intl. Conf. Intell. Robots Syst.*, 2019, pp. 4530–4537.

[2] L. Peters, D. Fridovich-Keil, V. Rubies-Royo, C. J. Tomlin, and C. Stachniss, "Inferring objectives in continuous dynamic games from noise-corrupted partial state observations," in *Proc. Robot, Sci. Syst.*, 2021.

[3] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard, "Autonomous robot navigation in highly populated pedestrian zones," *J. Field Robot.*, vol. 32, no. 4, pp. 565–589, 2014.

[4] B. Mersch, X. Chen, J. Behley, and C. Stachniss, "Self-supervised point cloud prediction using 3D spatio-temporal convolutional networks," in *Proc. Conf. Robot Learn.*, 2021, pp. 1444–1454.

[5] M. Toyungyernsub, M. Itkina, R. Senanayake, and M. J. Kochenderfer, "Double-prong ConvLSTM for spatiotemporal occupancy prediction in dynamic environments," in *Proc. IEEE Intl. Conf. Robot. Automat.*, 2021, pp. 13 931–13 937.

[6] X. Chen *et al.*, "Moving object segmentation in 3D LiDAR data: A. learning-based approach exploiting sequential data," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 6529–6536, Oct. 2021.

[7] J. Behley *et al.*, "Towards 3D LiDAR-based semantic scene understanding of 3D point cloud sequences: The SemanticKITTI dataset," *Int. J. Robot. Res.*, vol. 40, no. 8/9, pp. 959–967, 2021.

[8] H. Caesar *et al.*, "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11 621–11 631.

[9] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *Int. J. Robot. Res.*, vol. 32, no. 11, pp. 1231–1237, 2013.

[10] J. Behley *et al.*, "SemanticKITTI: A. dataset for semantic scene understanding of LiDAR sequences," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9297–9307.

[11] J. H. Giraldo, S. Javed, and T. Bouwmans, "Graph moving object segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 5, pp. 2485–2503, May 2022.

[12] P. W. Patil, K. M. Biradar, A. Dudhane, and S. Murala, "An end-to-end edge aggregation network for moving object segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 8146–8155.

[13] D. Yoon, T. Tang, and T. Barfoot, "Mapless online detection of dynamic objects in 3D LiDAR," in *Proc. Conf. Comput. Robot Vis.*, 2019, pp. 113–120.

[14] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Rigid scene flow for 3D LiDAR scans," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 1765–1770.

[15] S. A. Baur, D. J. Emmerichs, F. Moosmann, P. Pinggera, B. Ommer, and A. Geiger, "SLIM: Self-supervised LiDAR scene flow and motion segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13 126–13 136.

[16] Z. Gojcic, O. Litany, A. Wieser, L. J. Guibas, and T. Birdal, "Weakly supervised learning of rigid 3D scene flow," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 5692–5703.

[17] X. Liu, C. R. Qi, and L. J. Guibas, "FlowNet3D: Learning scene flow in 3D point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 529–537.

[18] 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.*, 2019, pp. 4213–4220.

[19] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "SalsaNext: Fast, uncertainty-aware semantic segmentation of LiDAR point clouds," in *Proc. IEEE Adv. Vis. Comput.*, 2020, pp. 207–222.

[20] S. Li, X. Chen, Y. Liu, D. Dai, C. Stachniss, and J. Gall, "Multi-scale interaction for real-time LiDAR data segmentation on an embedded platform," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 738–745, Apr. 2022.

[21] H. Shi, G. Lin, H. Wang, T.-Y. Hung, and Z. Wang, "SpSequenceNet: Semantic segmentation network on 4D point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4573–4582.

[22] H. Thomas, C. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6411–6420.

[23] 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 Robot.*, 2021, pp. 1–6.

[24] G. Kim and A. Kim, "Remove, then revert: Static point cloud map construction using multiresolution range images," in *Proc. IEEE/RSJ Intl. Conf. Intell. Robots Syst.*, 2020, pp. 10758–10765.

[25] 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. Automat. Lett.*, vol. 6, no. 2, pp. 2272–2279, Apr. 2021.

[26] 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 Intl. Conf. Robot. Automat.*, 2020, pp. 10 765–10 771.

[27] 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 Intl. Conf. Robot. Automat.*, 2021, pp. 11 641–11 647.

[28] J. Behley and C. Stachniss, "Efficient Surfel-based SLAM using 3D laser range data in urban environments," in *Proc. Robot., Sci. Syst.*, 2018.

[29] J. Behley, V. Steinhage, and A. Cremers, "Laser-based segment classification using a mixture of bag-of-words," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2013, pp. 4195–4200.

[30] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3D laser scans for online operation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 163–169.

[31] R. J. Campello, D. Moulavi, A. Zimek, and J. Sander, "Hierarchical density estimates for data clustering, visualization, and outlier detection," *ACM Trans. Knowl. Discov. Data*, vol. 10, no. 1, pp. 1–51, 2015.

[32] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Proc. Pacific-Asia. Conf. Knowl. Discov. Data Mining*, 2013, pp. 160–172.

[33] X. Weng, J. Wang, D. Held, and K. Kitani, "3D multi-object tracking: A baseline and new evaluation metrics," in *Proc. IEEE/RSJ Intl. Conf. Intell. Robots Syst.*, 2020, pp. 10 359–10 366.

[34] H. Kuhn, "The hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, no. 1/2, pp. 83–97, 1955.

[35] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-Net: Towards learning based LiDAR localization for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6389–6398.

[36] G. Kim, Y. Park, Y. Cho, J. Jeong, and A. Kim, "Mulran: Multimodal range dataset for urban place recognition," in *Proc. IEEE Intl. Conf. Robot. Automat.*, 2020, pp. 6246–6253.

[37] X. Chen, T. Läbe, A. Milioto, T. Röhling, J. Behley, and C. Stachniss, "OverlapNet: A. siamese network for computing LiDAR scan similarity with applications to loop closing and localization," *Auton. Robots*, vol. 46, pp. 61–81, 2021.

[38] X. Chen, T. Läbe, L. Nardi, J. Behley, and C. Stachniss, "Learning an overlap-based observation model for 3D LiDAR localization," in *Proc. IEEE/RSJ Intl. Conf. Intell. Robots Syst.*, 2020, pp. 4602–4608.