RVMOS: Range-View Moving Object Segmentation Leveraged by Semantic and Motion Features

Jaeyeul Kim, Jungwan Woo, and Sunghoon Im

***Abstract*—Detecting traffic participants is an essential and age-old problem in autonomous driving. Recently, the recognition of moving objects has emerged as a major issue in this field for safe driving. In this paper, we present RVMOS, a LiDAR Range-View-based Moving Object Segmentation framework that segments moving objects given a sequence of range-view images. In contrast to the conventional method, our network incorporates both motion and semantic features, each of which encodes the motion of objects and the surrounding circumstance of the objects. In addition, we design a new feature extraction module suitably designed for range-view images. Lastly, we introduce simple yet effective data augmentation methods: time interval modulation and zero residual image synthesis. With these contributions, we achieve a 19% higher performance (mIoU) with 10% faster computational time (34 FPS on RTX 3090) than the state-of-the-art method with the SemanticKitti benchmark. Extensive experiments demonstrate the effectiveness of our network design and data augmentation scheme.**

***Index Terms*—Autonomous driving, LiDAR, moving object segmentation, perception, range-view.**

I. Introduction

3D LIDAR is considered an essential sensor for safe autonomous driving because of its accurate and long-range 3D scanning ability, and large field of view [1]. Various studies on LiDAR-based scene understanding have been conducted including 3D object detection [2], segmentation [3], and moving object segmentation [4]. Early studies have begun to investigate on 3D LiDAR-based perception whose representations are processed in 3D space using point-based [5] or voxel-based [6] representations. These approahces have shown competitive performance but are computationally expensive because of the complexity of 3D point cloud processing methods. Recent works [4], [7] have adopted projection-based point cloud representations, such as a Range-View (RV) and Bird's-Eye-View (BEV), to take advantage of the computational efficiency of 2D space. They project a 3D point cloud onto a 2D plane and perform moving object segmentation with a lightweight 2D convolution operation.

Manuscript received 24 February 2022; accepted 14 June 2022. Date of publication 24 June 2022; date of current version 6 July 2022. This letter was recommended for publication by Associate Editor C. Wang and Editor C. C. Lerma upon evaluation of the reviewers' comments. This work was supported by the National Research Foundation of Korea (NRF) Grant, Korea government (MSIT) under Grant 2020R1C1C1013210, and in part by the DGIST R&D Program of the Ministry of Science and ICT (20-CoE-IT-01). *(Jaeyeul Kim and Jungwan Woo contributed equally to this work.) (Corresponding author: Sunghoon Im.)*

The authors are with the Department of Electrical Engineering and Computer Science, DGIST, Daegu 42988, Republic of Korea (e-mail: jykim94@dgist.ac.kr; friendship1@dgist.ac.kr; sunghoonim@dgist.ac.kr).

Digital Object Identifier 10.1109/LRA.2022.3186080

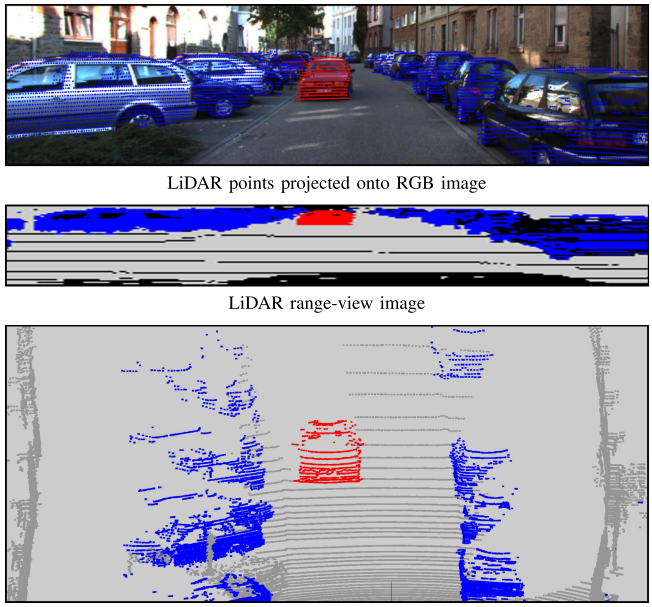


Fig. 1. Our result of moving objects (red-colored) and parked vehicles (blue-colored) segmentation in SemanticKitti MOS datasets [4].

Among the various perception tasks, Moving Object Segmentation (MOS) has recently emerged as a major topic of interest in autonomous driving [4], [7], largely because the MOS data can both provide the collision avoidance signals as well as indicate the objects that should be continuously tracked to drive safely. The MOS task aims to distinguish between parked and moving vehicles, as shown in Fig. 1, including those temporally stopped on the road (e.g., cars in front of intersections in Fig. 2). The approach also segments other moving traffic participants, such as pedestrians and cyclists. To achieve these goals, it is essential to perceive not only the motion of objects but also the circumstances surrounding the objects. However, most recent works [4], [7] have relied on motion cues to segment the objects and distinguish the moving and parked objects. These methods lack semantic representations and have limited performance, producing undesirable results. Although LMNet [4] post-processes with the binary mask from the pre-trained semantic network [8], it still suffers the inaccurate initial estimation, which is rarely corrected by post-processing.

To tackle the issue, we present the novel Range-View-based Moving Object Segmentation (RVMOS), which is explicitly guided by both semantics and motion using an Attention-based

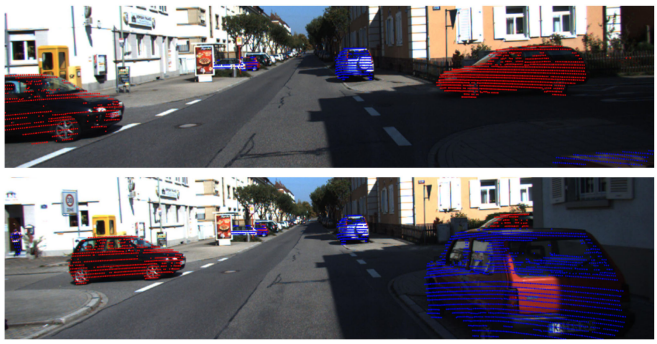


Fig. 2. SemanticKitti MOS datasets [4]. Projected GT LiDAR points into corresponding RGB images (top: seq0-index3659, bottom: seq0-index3654). Red and blue points indicate moving and parked vehicles, respectively.

Fusion Module (AFM). This design inspiration allows our network to not only catch the movement of objects but also effectively distinguish moving objects and parked vehicles, as shown in Fig. 1. In addition, we newly design the RV-based Feature Extraction Module (FEM) alleviating the geometric distortion of the range-view image. Lastly, we introduce simple yet effective data augmentation methods for RV-based MOS. As a result, our method achieves higher performance and lighter computational cost than the current state-of-the-art method.

In summary, the contributions of our work can be summarized as follows:

- We propose a Moving Object Segmentation framework (RVMOS) that clearly distinguishes moving objects from sequential range-view images.
- We design our network tailored for RV-based MOS considering of the characteristics of range-view images.
- We present simple yet effective data augmentation methods that increases the amount of range-view datasets.
- The proposed method achieves a 19% higher performance (mIoU) with 10% faster computational time (34 FPS on RTX 3090) than the state-of-the-art method [4] for the SemanticKitti benchmark.

II. Related Work

A. LiDAR Point Cloud Processing in 3D Space

LiDAR 3D point clouds have sparse and unordered characteristics, so two representative structures of Deep Neural Networks (DNNs) have been newly designed for 3D point clouds. Point-based methods [5], [9]–[13] directly pass point clouds into point-wise MLP to extract the representation of 3D point clouds. Several works [5], [12], [13] sample local region points with farthest point sampling or k-nearest neighborhood to aggregate local information. Voxel-based methods [3], [6], [14] voxelize unordered and sparse 3D point clouds into pre-defined grid cubes to process a 3D convolutional operation. Deng *et al.* [14] propose Voxel R-CNN, a framework for 3D object detection using both a 3D backbone and a 2D backbone. Zhu *et al.* [3] propose a cylindrical partitioning method for the LiDAR point cloud to solve the issues of irregular LiDAR density varying with the distance. These processing techniques have shown promising results for classification, detection, or segmentation, but these are computationally expensive. The point-based method consumes much time in the sampling process and the voxel-based approach requires heavy 3D convolution operations.

B. Projection-Based LiDAR Point Cloud Processing

Projection-based methods have been proposed to address the issues of the point cloud processing techniques described in Section II-A. These projection-based approaches are relatively cheaper in computational cost and memory cost because they are processed in lower dimensional space. They project 3D points onto a specific 2D image plane to generate RV [2], [8], [15]–[19] or BEV [20]–[24]. Wu *et al.* [15] use RV images to refine predicted semantic segmentation outputs with conditional random fields. RangeNet++ [17] further improves the range-view representation with the k-nearest neighbor to handle the misaligns in segmentation outputs. Cortinhal *et al.* [8] propose SalsaNext for LiDAR segmentation task with range-view input by extending SalsaNet [21]. Gerdzhev *et al.* [20] combine BEV and RV representation by further aggregating the features in multiple views. Some of recent works have proposed LiDAR BEV and RGB image fusion for 3D object detection [25] and end-to-end autonomous driving [26].

C. LiDAR-Based Moving Object Segmentation

The LiDAR-based moving object segmentation aims to segment moving objects given a sequence of 3D LiDAR point clouds. Dewan *et al.* [27] segment moving objects by putting trained semantic and motion features into a Bayes filter. Rashed *et al.* [28] present FuseMODNet to segment moving objects using both camera and LiDAR. Pagad *et al.* [29] introduce a method to remove the moving objects from the point cloud with an occupancy map. The work by [30] segments moving points and constructs a static point cloud map using range image-based point discrepancy computation. Pfreundschuh *et al.* [31] present an occupancy grid-based approach to segment moving points. Recently, Chen *et al.* [4] release a moving object segmentation dataset based on SemanticKitti [32], and open the benchmark site. In addition, they propose a moving object segmentation network, called LMNet, using residual range-view images as inputs. Mohapatra *et al.* [7] introduce a framework with a low computational load based on BEV representation. The work in [33] proposes an auto-labeling pipeline moving object segmentation using occupancy-based dynamic object removal and a Kalman filter [34].

III. Proposed Method

In this section, we propose our Range-View-based framework for Moving Object Segmentation (RVMOS) whose overall pipeline is illustrated in Fig. 3. We describe the projection of LiDAR 3D points into range-view images in Section III-A. We present our RVMOS network structure and modules in Section III-B. We introduce our augmentation schemes and describe training details in Section III-C and Section III-D, respectively.

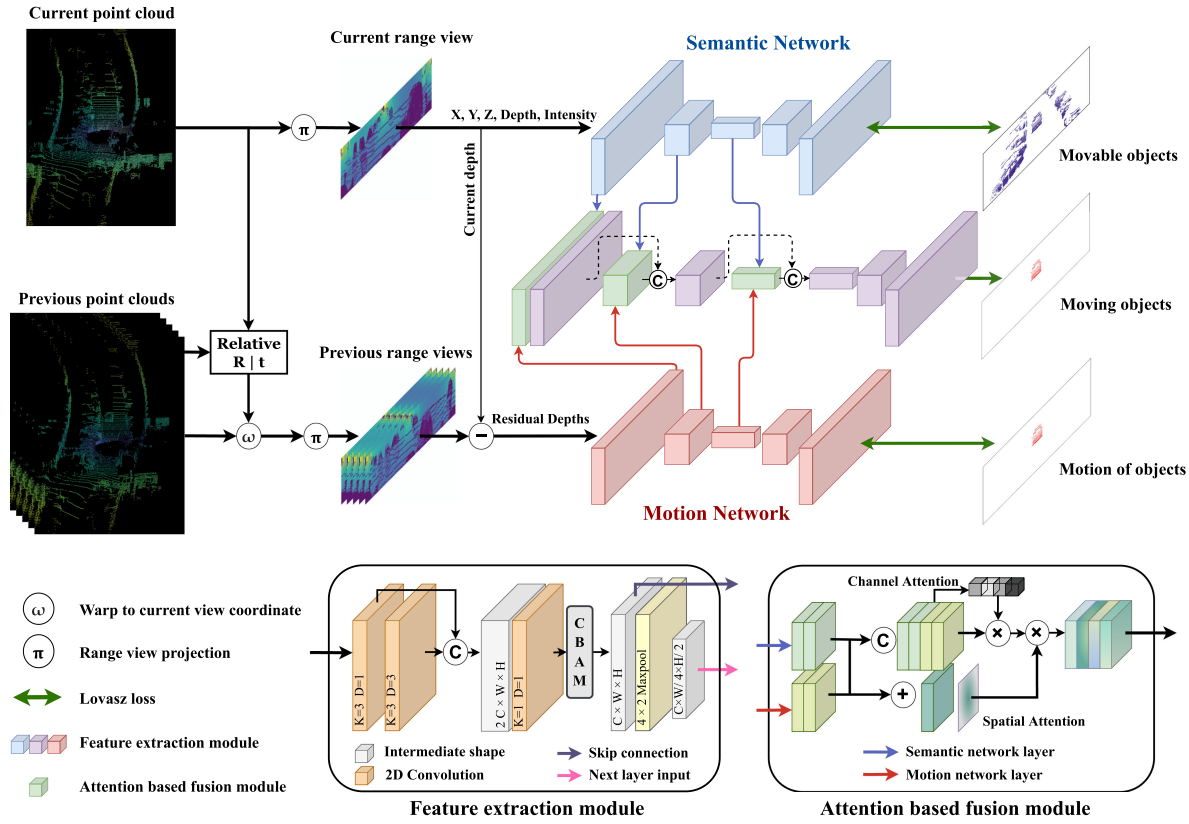


Fig. 3. Schematic overview of the RVMOS. Upper blue blocks refer to Semantic Network which segments movable objects. Bottom red blocks mean Motion Network which segments movement of objects. Middle purple blocks refer to Fusion Network which predicts final moving object segmentation output. Middle green blocks refer to Attention based fusion module. Each network actually consists of a total of 9 blocks, but is represented by 5 blocks for better visual understanding.

A. Data Preprocessing

We aim to obtain 2D range-view images $\mathbf{I}_{RV}$ and residual images $\mathbf{I}_{res}$, given N number of 3D point clouds $\mathbf{X} \in \mathbb{R}^{N\times 3}$ and their remission values $e \in \mathbb{R}^{N\times 1}$. Following previous works [4], [17], [19], we define the transformation from 3D LiDAR points $\mathbf{X}_i = [X_i, Y_i, Z_i] \in \mathbb{R}^3$ to 2D image coordinates in range-view $\mathbf{x}_i = [u_i, v_i] \in \mathbb{R}^2$ as follows:

$$\mathbf{x}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} \frac{1}{2}\left(1 - \arctan(Y_i, X_i)\pi^{-1}\right) W \\ \left(1 - \left(\arcsin\left(Z_i r_i^{-1}\right) + \mathrm{f}_{\mathrm{up}}\right)\mathrm{f}^{-1}\right) H \end{bmatrix},$$
$$r_i = \sqrt{X_i^2 + Y_i^2 + Z_i^2},\ i \in \{1, \ldots, N\}, \tag{1}$$

where f and f_{up} are the sensor field-of-view and upper part field-of-view when f is divided by plane $z = 0$. We set the range-view image $\mathbf{I}_{RV} \in \mathbb{R}^{W\times H\times 5}$ with the size of (W, H) whose intensities are initialized as zero.

We obtain the image by projecting the 3D LiDAR data $\mathbf{L} = [X; Y; Z; r; e] \in \mathbb{R}^{N\times 5}$ into the corresponding 2D image coordinates $\mathbf{x} = [\mathbf{u}; \mathbf{v}] \in \mathbb{R}^{N\times 2}$ as follows:

$$\mathbf{I}_{RV}(\mathbf{u}, \mathbf{v}) = \mathbf{L}. \tag{2}$$

Then, we warp past K point cloud $\{\mathbf{X}^k \mid k = 1, 2, \ldots, K\}$ into the current frame coordinates $\mathbf{X}^0$ using the known relative poses $\mathbf{P}^k$. We obtain residual images $\mathbf{I}_{res}$ by calculating the normalized absolute difference between past and current range-view images as follows:

$$\mathbf{I}_{res}^k(\mathbf{u}, \mathbf{v}) = \left| \frac{\mathbf{I}_{RV}^k(\mathbf{u}, \mathbf{v}) - \mathbf{I}_{RV}^0(\mathbf{u}, \mathbf{v})}{\mathbf{I}_{RV}^0(\mathbf{u}, \mathbf{v})} \right|,$$
$$\mathbf{I}_{RV}^k(\mathbf{u}, \mathbf{v}) = \mathbf{L}^k = \left[\mathbf{P}^k \mathbf{X}^k; r^k, e^k\right]. \tag{3}$$

We ignore invalid pixels with distance values out of range between 0.2 m to 50 m for this computation in (3), which means they set to zero. We use the provided relative pose $\mathbf{P}^k$ from the SemanticKitti dataset. We concatenate the K residual range images as an input of the motion network. For residual images $\mathbf{I}_{res}^k \in \mathbb{R}^{W\times H}$, we only use depth channel r^k from among the five channel LiDAR data. We use five residual images $K = 5$ for the motion network.

B. Network Structure

1) Multi-Branch Networks: The proposed framework consists of three networks: semantic, motion, and fusion networks in training time as shown in Fig. 3. We separate current and past residuals and then feed them into the semantic and motion networks to encode semantic information and motion cues, respectively. The semantic network segments movable objects regardless of movement given a current range image. Because the current single frame is sufficient to detect a movable object,

the semantic network only uses the current frame as input. The motion network segments the motion of objects given the residual range images of current and previous frames. The residual image can be used to detect motion because the residual has a non-zero value at the boundary of the moving objects.

We also design an attention-based fusion module to fuse the intermediate feature layers from the semantic and motion network using spatial and channel-wise attention [35] as shown in Fig. 3. The fusion network predicts a moving object probability map given the fused features, encapsulating both the movable objects and their motion information. In this way a moving vehicle can be inferred from a motion cue, and a temporarily stationary vehicle can be inferred through the semantic features encoding the surrounding circumstance of objects. We train all these networks simultaneously. After training, the fusion network with the semantic and motion features is only used for the moving object segmentation.

2) Feature Extraction Module: We design the RV-based Feature Extraction Module (FEM) alleviating the geometric distortion of the range-view image. The range-view image is formed by projecting 360° Lidar points into a 2D plane so that it captures the distorted 3D geometry. In particular, objects in the range-view are expressed in various sizes, making it difficult to segment objects. In addition, the moving object segmentation task should robustly segment regardless of the magnitude of the object motion. To encode various motions from the residual image, the feature extractor requires an adaptive receptive field. To accomplish this, we design Feature Extraction Module (FEM) with the stacked multiple dilated rate convolution layers. It consists of two convolution layers of kernel size 3×3 with dilated rates 1 and 3, followed by a 1×1 convolution layer for feature refinement with attention module [35].

Moreover, A range-view image is a projection of a 3D LiDAR point with an elevation angle wider than the azimuth angle per index. Due to this intrinsic characteristic, the range-view captures a vertically elongated 3D space per unit pixel area. It means that the typical convolutional operations tend to extract range-view features with distorted 3D geometry. To alleviate the issue, we use vertically elongated pooling (i.e. 4×2 max-pooling) in the feature extraction module instead of a common rectangular-shaped pooling operation. This operation compensates for the distorted aspect ratio and helps extract features that reflect the 3D geometry closer to the real world. With broader max-pooling the amount of computation is also reduced.

C. Data Augmentation

We introduce two simple yet effective data augmentation schemes to improve the performance of the proposed network without additional computational load. In contrast to general data augmentation strategies in the spatial domain, such as adjusting scale, rotation, or the translation of points used in previous works [3], [4], [8], [20], [36], the proposed schemes achieve additional performance improvement with a simple transformation in the temporal domain.

1) Time Interval Modulation: In the training phase, we generate more residual frames using the range-view image at shorter time intervals. We modulate the index k in (3) by multiplying constant $\tau \in \{1, 2\}$ as follows:

$$\mathbf{I}^k_{res}(\mathbf{u}, \mathbf{v}) = \left| \frac{\mathbf{I}^{\tau k}_{RV}(\mathbf{u}, \mathbf{v}) - \mathbf{I}^0_{RV}(\mathbf{u}, \mathbf{v})}{\mathbf{I}^0_{RV}(\mathbf{u}, \mathbf{v})} \right|. \tag{4}$$

In the inference phase, we use $\tau = 2$ as the default time interval. We use either the interval $\tau = 1$ or 2 for this augmentation scheme in the training stage. This data augmentation technique simulates the speed of the foreground object more slowly than the actual speed. It increases the number of residual images so that the network effectively handles the slower moving objects in the test phase.

2) Zero Residual Image Synthesis: Objects moving at fairly slow speeds have few motion signals, making it difficult for the model to segment objects with motion features. We tackle the issue by data augmentation of residual images and the corresponding motion maps with all zero values. We use the synthesized residual images as an input of the motion network and the motion maps for supervision of the network. This augmentation scheme allows the network to experience more situations where the object is stationary in the last five frames during training. Moreover, this prevents the fusion network from relying excessively on the motion network features and allows the features of the semantic network to be taken into account. We train the motion network using synthetic data with a probability of 0.2, otherwise we do it with non-augmented data.

D. Training Details

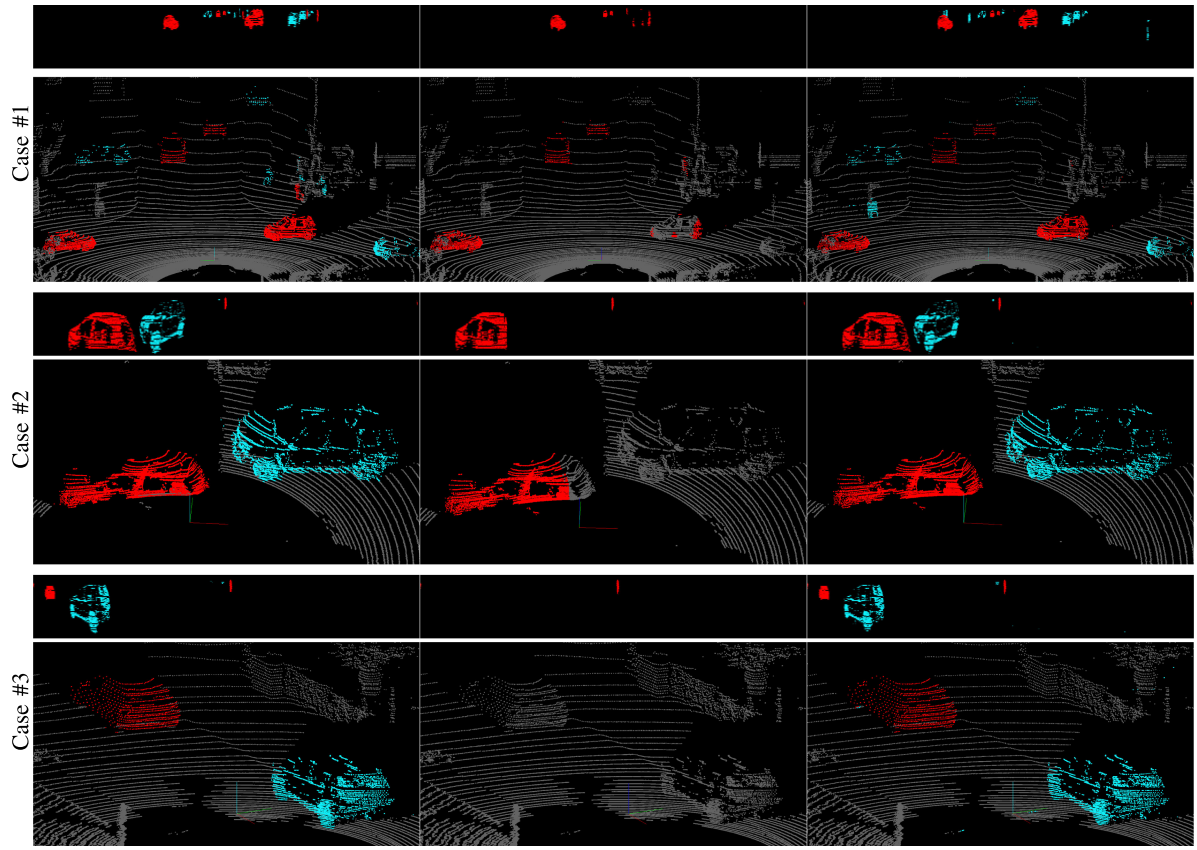
We train our networks with the sum of semantic, motion, and fusion losses as follows:

$$\mathcal{L}_{Total} = \mathcal{L}_{Semantic} + \mathcal{L}_{Motion} + \mathcal{L}_{Fusion}. \tag{5}$$

We impose supervisory losses using Lovasz-Softmax loss [37] for all networks using the ground truth labels of the movable and moving object segmentation. We use Adam optimizer with a batch size of 16. We set 0.005 as the initial learning rate, and decay the rates by a factor of 0.99 for every epoch. We train the entire network for 200 epochs. For the SemanticKitti dataset, we use sequences 00 to 10 except sequence 08 for training, and sequence 08 for validation. We also augment the data by flipping the range-view images left and right with a probability of 0.25. All semantic, motion and moving object segmentation networks are trained simultaneously. We do not use any pre-trained weights. We train all networks from scratch.

IV. Experiments

In this section we show the results of our method for the moving segmentation task. First in Section IV-A, we compares our results qualitatively and quantitatively with other state-of-the-art methods. Then we perform ablation studies and demonstrate how each component contributes to total performance in Section IV-B. Also in Section IV-D, we show that our method



(a) Ground-truth (b) LMNet [4] (c) Ours

Fig. 4. Qualitative comparison of moving (red-colored) and movable (cyan-colored) object segmentation on the SemanticKitti. (top: range-view images, bottom: 3D point clouds.).

TABLE I
QUANTITATIVE COMPARISON OF SEMANTIC SEGMENTATION PERFORMANCE ON SEMANTICKITTI'S TEST SET

Method	mIoU
Flownet3d [38]	0.287
SpSequenceNet [39]	0.432
SalsaNext [8]	0.466
KPConv [36]	0.609
LMNet [4]	0.625
OURS	**0.747**

Bold represents the best performance.

TABLE II
QUANTITATIVE COMPARISON OF SEMANTIC SEGMENTATION PERFORMANCE ON SEMANTICKITTI'S VALIDATION SET

Method	mIoU
MINet [19]	0.369
RangeNet++ [17]	0.395
LiMoSeg [7]	0.526
SalsaNext [8]	0.534
LMNet (with 5 residual imgs) [4]	0.643
LMNet (with 8 residual imgs) [4]	0.671
OURS	**0.712**

Bold represents the best performance.

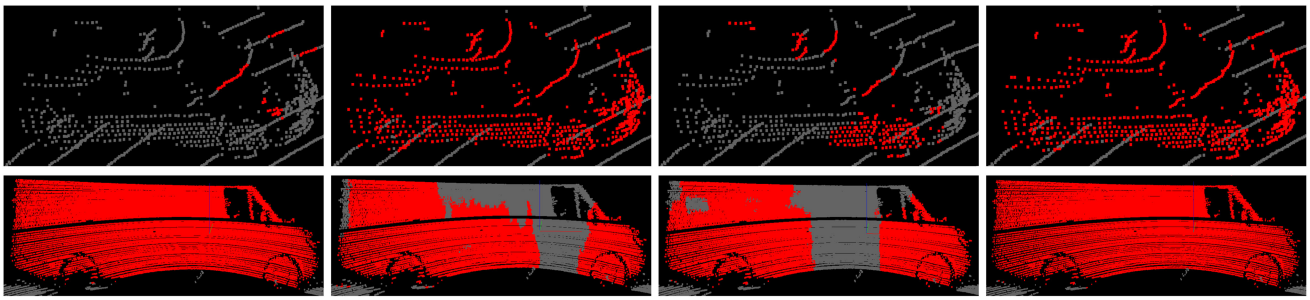
is lightweight and fast enough for real-time autonomous driving system. Finally, in Section IV-E, we present instance segmentation results beyond moving object semantic segmentation.

A. Comparison to State-of-The-Arts Methods

We compare the proposed method to state-of-the-art MOS methods [4], [7]. We measured the mIoU of our approach by uploading the moving object segmentation results of the test sequences to the SemanticKITTI benchmark server [4]. Table I shows a comparison of the semantic segmentation performance result with state-of-the-art algorithms on the SemanticKitti test set. Table II shows the results of the validation set. Our RVMOS outperforms all of the competitive methods in both validation and test sets of the SemanticKitti datasets. We achieved a 19% higher mIoU score than LMNet [4] on the test benchmark. The qualitative results in Fig. 4 show that our method segments the moving vehicles more accurately than LMNet [4], while also segmenting parked vehicles. Our RVMOS outperforms the state-of-the-art method for objects of various sizes and low-speed thanks to the proposed feature extraction module (FEM) and data augmentation methods.

B. Ablation Studies

We conduct various ablation studies to verify how each component affects performance in Table III. As shown in Table III-(a),(b),(c), our data augmentation schemes slightly



(a) w/o AUG (b) w/o FEM (c) LMNet [4] (d) Ours

Fig. 5. Qualitative results of ablation study on the SemanticKitti. Our augmentation method shows visually good results in slow-speed and large-size vehicles. (top: a vehicle moving at a slow speed of about 10 km/h, bottom: a large-sized van.)

TABLE III
ABLATION STUDY. (A) WITHOUT ALL AUGMENTATIONS. (B) WITHOUT ZERO RESIDUAL AUGMENTATION. (C) WITHOUT FRAME INTERVAL AUGMENTATION. (D) WITHOUT OUR FEATURE EXTRACTION MODULE(FEM). (E) OUR RVMOS

	Error metric			
Method	mIoU	TP	FP	FN
(a) w/o All aug	0.643	1.16M	**0.15M**	0.50M
(b) w/o Residual aug	0.683	1.25M	0.16M	0.42M
(c) w/o Interval aug	0.686	1.26M	0.17M	0.40M
(d) w/o FEM	0.673	1.25M	0.19M	0.41M
(e) Ours	**0.712**	**1.32M**	0.19M	**0.34M**

Bold represents the best performance.

TABLE IV
COMPUTATION RESOURCE COMPARISON

Method	FPS	ms	params	size (MB)
(a) LMNet (RangeNet++) [4]	25	40	50.37M	192.3
(b) LMNet (SalsaNext) [4]	31	32	6.71M	25.6
(c) LiMoSeg [7]	**125**	**8**	-	35.0
(d) Ours	34	29	**2.63M**	**10.0**

Bold represents the best performance.

increase the number of false-positives while significantly reducing the number of false-negatives. With both augmentation schemes, the performance is more improved. The results shows the effectiveness of the proposed data augmentation methods. In particular, the performance of low-speed object segmentation is considerably improved as shown in the first column of Fig. 5. We believe that our data augmentation helps RVMOS to robustly segment a low-speed object by relying on semantic information. We also compare the performance with the conventional feature extractor in Table III-(d) and the proposed feature extraction module (FEM) in Table III-(e). We set the conventional extractor with 2×2 max-pooling size and a dilation rate of 1 without the attention module, following SalsaNext [8]. The results show that the proposed FEM decreases both FP and FN and improves the overall performance. As shown in the second column of Fig. 5, the proposed method with FEM shows better segmentation ability than LMNet [4] for large vehicles requiring an enormous receptive field. In addition, when we apply our augmentation methods to the LMNet we reproduce, the performance improves from 0.638 to 0.654, which shows that our augmentation methods can be applied effectively even to the other frameworks.

C. Generalization Ability

We conduct experiments to demonstrate the generalization ability of the proposed method. We train our RVMOS on SemanticKitti dataset [4] and test on the Waymo dataset [40]. For the network training and test, we exclude the intensity values provided by two datasets because the intensity ranges of the datasets are different. Note that the range of SemanticKitti (Velodyne LiDAR) is from zero to one while Waymo (Waymo LiDAR) is from zero to infinity. We only conduct qualitative evaluation due to the absence of GT MOS labels on Waymo datasets. The results in Fig. 6 show that the proposed method reasonably segments the moving and movable objects in other environments although the sensor and the driving environment are different. This experiment shows the generality of our RVMOS.

D. Computational Resources

We measure the inference time (FPS, ms), memory usage (size), and the number of learnable parameters (params) for ours and state-of-the-art methods in Table IV. We use author-provided numbers for all measurements in LiMoSeg [7], which is computed on an NVIDIA Jetson Xavier. LiMoSeg [7] achieves the fastest computational time, but requires a larger memory and underperforms all competitive methods by a large margin in Table II. We conduct the experiments on ours and LMNet [4] on NVIDIA RTX 3090. Compared to LMNet [4], our method achieves a 19% higher mIoU with 10% higher operating speed and 60.8% fewer parameters. The results in Table II and Table IV show our method runs in real-time with a reasonable computational load and achieves the best performance.

E. Application: Instance Segmentation

We briefly introduce one application of our method, Moving Instance Segmentation (MIS) which makes meaningful use of the extracted segmentation labels. We design a simple instance segmentation head to distinguish each traffic participant. We utilize the semantic features from RVMOS and

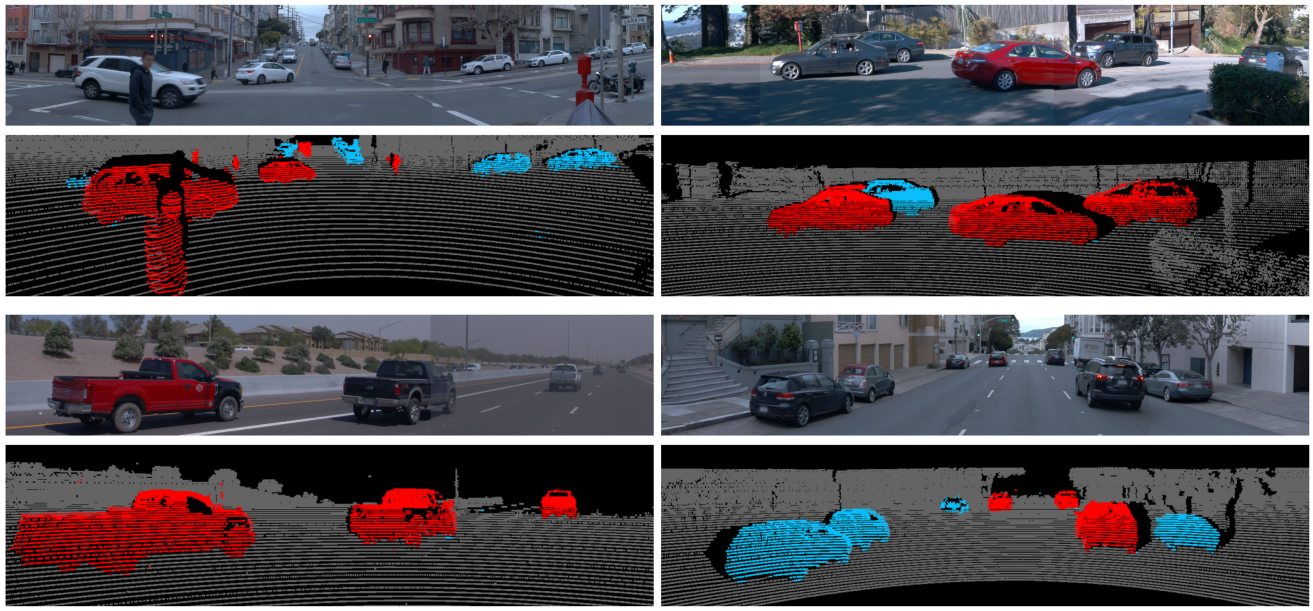


Fig. 6. Qualitative results of moving (red-colored) and movable (cyan-colored) object segmentation on the Waymo dataset. RGB images from different views were manually stitched for better visual understanding.

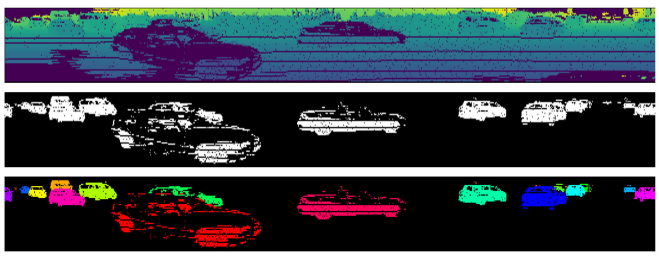


Fig. 7. The result of moving object instance segmentation through the lightweight instance module on the SemanticKitti test set. Raw RV (top), moving object mask from RVMOS (middle), and the resulting instance mask (bottom).

the additional positional feature encoded from each 3D point. We train the instance head by minimizing the discriminative loss [41] with ground truth moving object instance labels. In test time, we extract features from the instance head and cluster the features using HDBSCAN [42]. The instance head consists of only about 0.3 M parameters, and just 30 epochs of additional training are required using pre-trained RVMOS weights. Fig. 7 shows visually pleasing clustering results, even for complex scenes with dozens of vehicles. This experiment demonstrates that our RVMOS can be practically utilized for instance discrimination.

V. Conclusion

In this paper, we present a LiDAR Range-View-based Moving Object Segmentation method. The framework obtains current RV and multiple residual images while subtracting current and previous RV images, and outputs the moving and movable objects. The proposed network is explicitly guided by the motion and semantic features, each of which encapsulates the object motion information and the surrounding circumstance of the objects. We also design a suitable feature extraction module considering the range-view input setup. Lastly, we introduce effective data augmentation methods that improves the segmentation performance. Extensive experiments show that RVMOS outperforms state-of-the-art methods and that each of our technical contributions is effective. As future work, we plan to study a framework that not only tracks moving objects but also predicts their velocity.

Appendix

In the proposed structure, feature fusion is performed only in the encoder except for the decoder. We perform additional experiments on the case of feature fusion in both the encoder and decoder, as shown in Table V. Also, we attach performance comparisons with varying loss weights of semantic, motion, and fusion networks in Table VI.

TABLE V
Performance Difference Depending on Where the Feature Fusion Takes Place

Fusion layers	mIoU	TP	FP	FN
(a) Enc. (Ours)	0.712	1.32M	0.19M	0.34M
(b) Enc.+Dec.	0.684	1.27M	0.20M	0.39M

TABLE VI
Performance Difference According to the Loss Weights of the Three Networks (Semantic, Motion, and Fusion)

Loss weight (S : M : F)	mIoU	TP	FP	FN
(a) 1:1:1	0.712	1.32M	0.19M	0.34M
(b) 8:1:1	0.688	1.28M	0.19M	0.39M
(c) 1:8:1	0.682	1.28M	0.21M	0.39M
(d) 1:1:8	0.707	1.30M	0.17M	0.37M

REFERENCES

[1] 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.

[2] L. Fan, X. Xiong, F. Wang, N. Wang, and Z. Zhang, "RangeDet: In defense of range view for LiDAR-based 3D object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 2918–2927.

[3] X. Zhu *et al.*, "Cylindrical and asymmetrical 3D convolution networks for LiDAR segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9939–9948.

[4] 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.

[5] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet: Deep hierarchical feature learning on point sets in a metric space," in *Neural Inf. Process. Syst.*, 2017, pp. 4490–4499.

[6] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 4490–4499.

[7] S. Mohapatra *et al.*, "LiMoSeg: Real-time bird's eye view based LiDAR motion segmentation," in *Proc. Int. Conf. Comput. Vis. Theory Appl.*, 2022, pp. 828–835.

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

[9] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2017, pp. 652–660.

[10] J. Li, B. M. Chen, and G. H. Lee, "So-net: Self-organizing network for point cloud analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9397–9406.

[11] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2sequence: Learning the shape representation of 3D point clouds with an attention-based sequence to sequence network," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 01, pp. 8778–8785.

[12] N. Luo *et al.*, "KVGCN: A KNN searching and VLAD combined graph convolutional network for point cloud segmentation," *Remote Sens.*, vol. 13, no. 5, 2021, Art. no. 1003.

[13] L. Ge, Z. Ren, and J. Yuan, "Point-to-point regression PointNet for 3D hand pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 475–491.

[14] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li, "Voxel R-CNN: Towards high performance voxel-based 3D object detection," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 1201–1209.

[15] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1887–1893.

[16] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 4376–4382.

[17] 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.

[18] C. Xu *et al.*, "Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 1–19.

[19] 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.

[20] M. Gerdzhev, R. Razani, E. Taghavi, and B. Liu, "Tornado-Net: Multiview total variation semantic segmentation with diamond inception module," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 9543–9549.

[21] E. E. Aksoy, S. Baci, and S. Cavdar, "SalsaNet: Fast road and vehicle segmentation in LiDAR point clouds for autonomous driving," in *Proc. IEEE Intell. Veh. Symp.*, 2020, pp. 926–932.

[22] L. Caltagirone, S. Scheidegger, L. Svensson, and M. Wahde, "Fast LiDAR-based road detection using fully convolutional neural networks," in *Proc. IEEE Intell. Veh. Symp.*, 2017, pp. 1019–1024.

[23] Y. Zeng *et al.*, "RT3D: Real-time 3-D vehicle detection in LiDAR point cloud for autonomous driving," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 3434–3440, Oct. 2018.

[24] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-YOLO: An Euler-region-proposal for real-time 3D object detection on point clouds," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2018, pp. 197–209.

[25] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, "Multi-task multi-sensor fusion for 3D object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7345–7353.

[26] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7077–7087.

[27] A. Dewan, G. L. Oliveira, and W. Burgard, "Deep semantic classification for 3D LiDAR data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 3544–3549.

[28] H. Rashed, M. Ramzy, V. Vaquero, A. E. Sallab, G. Sistu, and S. Yogamani, "FuseMODNet: Real-time camera and LiDAR based moving object detection for robust low-light autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2019.

[29] S. Pagad, D. Agarwal, S. N. K. Rangan, H. Kim, and G. Yalla, "Robust method for removing dynamic objects from point clouds," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 10765–10771.

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

[31] P. Pfreundschuh, H. F. Hendrikx, V. Reijgwart, R. Dube, R. Siegwart, and A. Cramariuc, "Dynamic object aware LiDAR SLAM based on automatic generation of training data," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 11641–11647.

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

[33] X. Chen *et al.*, "Automatic labeling to generate training data for online LiDAR-based moving object segmentation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 6107–6114, 2022.

[34] R. E. Kalman, "A new approach to linear filtering and prediction problems," *J. Fluids Eng.*, vol. 82, no. 1, pp. 35–45, 1960.

[35] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*., 2018, pp. 3–19.

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

[37] M. Berman, A. R. Triki, and M. B. Blaschko, "The Lovasz-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4413–4421.

[38] X. Liu, C. R. Qi, and L. J. Guibas, "Flownet3d: Learning scene flow in 3D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 529–537.

[39] H. Shi, G. Lin, H. Wang, T.-Y. Hung, and Z. Wang, "SpSequenceNet: Semantic segmentation network on 4D point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4574–4583.

[40] P. Sun *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2446–2454.

[41] B. D. Brabandere, D. Neven, and L. V. Gool, "Semantic instance segmentation with a discriminative loss function," in *Proc. IEEE Conf. Comput. Vis. Pattern Regnit. Workshops*, 2017, pp. 478–480.

[42] L. McInnes and J. Healy, "Accelerated hierarchical density based clustering," in *Proc. IEEE Int. Conf. Data Mining Workshops*, 2017, pp. 33–42.