RangeSeg: Range-Aware Real Time Segmentation of 3D LiDAR Point Clouds

Tzu-Hsuan Chen and Tian Sheuan Chang, *Senior Member, IEEE*

***Abstract*—Semantic outdoor scene understanding based on 3D LiDAR point clouds is a challenging task for autonomous driving due to the sparse and irregular data structure. This paper takes advantages of the uneven range distribution of different LiDAR laser beams to propose a range aware instance segmentation network, RangeSeg. RangeSeg uses a shared encoder backbone with two range dependent decoders. A heavy decoder only computes top of a range image where the far and small objects locate to improve small object detection accuracy, and a light decoder computes whole range image for low computational cost. The results are further clustered by the DBSCAN method with a resolution weighted distance function to get instance-level segmentation results. Experiments on the KITTI dataset show that RangeSeg outperforms the state-of-the-art semantic segmentation methods with enormous speedup and improves the instance-level segmentation performance on small and far objects. The whole RangeSeg pipeline meets the real time requirement on NVIDIA JETSON AGX Xavier with 19 frames per second in average.**

***Index Terms*—LiDAR point clouds, semantic segmentation, instance segmentation, deep learning.**

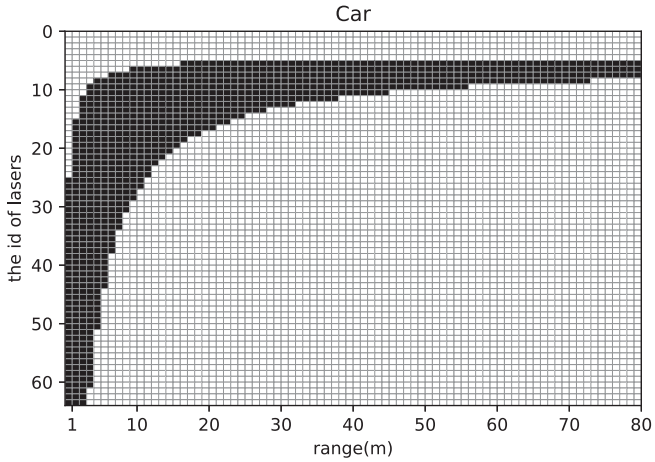


Fig. 1. Range distributions of the detected cars (denoted by black) for different ID of lasers in LiDAR. The simulation setting is based on Velodyne HDL-64E S3.

I. Introduction

The semantic scene understanding from 3D LiDAR point clouds is one of the fundamental blocks to provide robust and real time 3D object detectors for the autonomous driving. LiDAR sensors provide range measurements by sampling a specific location with spanning vertical and horizontal angular resolutions, which is different from the 3D point clouds sampled densely and uniformly on all sides used in indoor scenes. In addition, LiDAR sensors are robust under almost all light conditions or the foggy weather. As a result, 3D LiDAR point clouds attract the significant research attention recently.

The major difficulty in processing LiDAR data is that the sensors provide non-Euclidean data in the form of point clouds with about 100 k points around 360°, which poses a great challenge for object detection and segmentation tasks and thus needs the high computational cost. For the 3D object detection and the 3D semantic segmentation, the segmentation task gives the dense predictions of the scene understanding. The previous work such as SqueezeSeg [1] proposed a light weight convolutional neural network (CNN) backbone with the conditional random field (CRF) [2] to get the real time performance, but it still leaves room for accuracy improvements. Higher accuracy demands higher computational cost, which hurts real time performance.

To achieve high accuracy within a real time constraint, this paper proposes a instance-level segmentation, denoted as RangeSeg. RangeSeg exploits the uneven range distribution of 3D LiDAR point clouds as shown in Fig. 1 for the autonomous driving. It has far and small objects at the top of the range image, and large and near objects at the bottom of the range image. This algorithm achieves high accuracy for these far and small objects by adopting a heavy decoder only on the top of the image, and meets the real time demand by adopting a light decoder on the whole image with a shared backbone encoder. The semantic segmentation results are further clustered as instances by the density-based spatial clustering and applications with noise (DBSCAN) [3] method based on a resolution weighted distance. The result shows that the proposed method can improve the detection on far and small objects with the real time execution performance on NVIDIA JETSON AGX Xavier.

The rest of the paper is organized as follows. Section II first introduces the related works. Section III presents our approach. Section IV shows the experimental results and comparisons with other methods. Finally, we conclude this paper in Section V.

Manuscript received June 17, 2020; revised October 20, 2020, January 25, 2021, and April 18, 2021; accepted May 29, 2021. Date of publication June 2, 2021; date of current version April 21, 2022. This work was supported by the Ministry of Science and Technology, Taiwan, under Grants 109-2634-F-009-022 and 109-2639-E-009-001. *(Corresponding author: Tian Sheuan Chang.)*

The authors are with Department Electronics Engineering, National Chiao Tung University, Hsinchu 30010, Taiwan (e-mail: jesse.ee06g@nctu.edu.tw; tschang@g2.nctu.edu.tw).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIV.2021.3085827.

Digital Object Identifier 10.1109/TIV.2021.3085827

II. Related Works

A. Data Representation of 3D LiDAR Point Clouds

3D LiDAR point clouds have an unstructured data format. To tackle such unstructured data for 3D outdoor scene understandings, one approach is to transform point clouds into a structured format to utilize standard convolutional operations. The other approach is to define a new operation directly on unstructured data. Current data transformations are mainly divided into two types: 3D voxel grids or 2D projections. The 3D voxel grid method transforms the data into a regular space of 3D grid such that the following 3D convolution operations can be applied to extract the high order of feature representations. However, the 3D voxel grids are sparse since the point clouds are inherently sparse, which wastes lots of computations on unnecessary grids. The 2D projections such as the birds' eye view (BEV) and the range image encoding are much more compact. BEV is sparse while preserves the size of objects. The range image encoding is dense but distorts objects. The novel graph-based neural networks can directly apply on point clouds, but the recent approaches only apply on the 3D indoor scene understanding, in which the point clouds are uniform sampled on surface with about 1 k points.

B. 3D Object Detection of Point Clouds

VoxelNet [4] encodes point clouds into hand-crafted 3D voxel grids by a voxel feature encoding layer to extract high order of features. They use 3D CNN layers to aggregate the voxel-wise features with expanded receptive fields. However, the 3D CNN has high computational cost even for such sparse representation. Its real time performance is limited to four frames per second (fps). PIXOR [5] uses a 2D birds' eye view representation to build a real time pipeline, but fails to deliver a good performance on small objects. PointRCNN [6] is the first two-stage 3D detector that only uses 3D LiDAR point clouds, which uses PointNet++ [7] on the unstructured data to get the preliminary bounding boxes and applies a simple multi-layer neural network to get a final prediction. This approach gets a good performance on small objects such as pedestrians and cyclists, but the two-stage pipeline makes it unsuitable for real time applications.

C. 3D Semantic Segmentation of Point Clouds

SqueezeSeg [1], PointSeg [8] and SqueezeSegV2 [9] all use a light weight SqueezeNet [10] as their backbone with range images as input for the semantic segmentation task. They use different post-processing methods to improve the accuracy. SqueezeSeg and SqueezeSegV2 use a recurrent CRF module [2] to reduce the blurry boundaries. SqueezeSegV2 additional tackles the inherent problems of missing points in range images with a context aggregation module. PointSeg uses a squeeze re-weighting layer [11] and an enlargement layer [2] to achieve a better performance. RIU-Net [12] directly uses U-Net [13] on range images with focal loss [14]. These works get real time performance due to a light weight backbone but has a low accuracy. Moreover, none has discussed the instance-level segmentation.

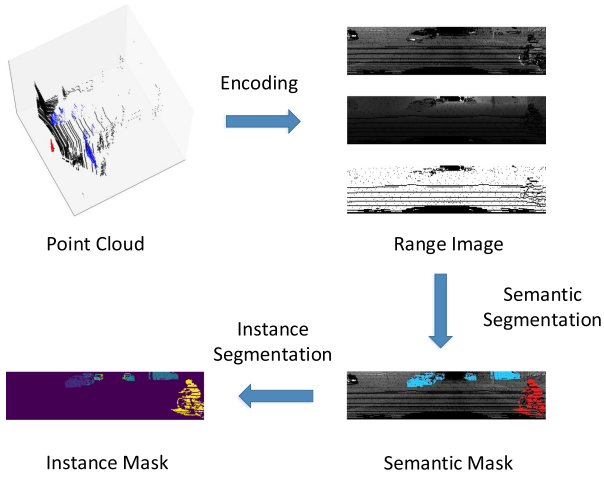


Fig. 2. The proposed pipeline of RangeSeg.

D. Graph Neural Networks of Point Clouds

Pointnet [15] proposed to use an end-to-end pipeline to learn point-wise features directly from point clouds. The follow-up work improves the performance by extracting local features [7]. Furthermore, DGCNN [16] and PointCNN [17] define a new convolution operation on point clouds. They succeed in the 3D indoor scene understanding ($\sim$1 k points). However, the outdoor scene contains about 100 k points, which will make the above network training demand high requirements of the memory and the computation.

E. 3D Instance Segmentation of Point Clouds

A novel paper [18] proposed a pipeline with a feature learning network and a stacked hourglass network for the instance segmentation in the outdoor LiDAR point clouds, which could help localize the small and far-away objects. However, the high complexity of the model makes it hard to run in real time, where the elapsed time in TESLA V100 GPU was 300 ms.

III. RangeSeg Framework

In this paper, we propose a real time pipeline, RangeSeg, that gets accurate instance-level segmentation results by exploiting 2D range image representations of LiDAR point clouds. Our range-aware framework attains a fast and accurate semantic segmentation by a complex heavy decoder to predict far and small objects and a light decoder to reduce complexity of general predictions. Next, we use DBSCAN with the proposed weighted distance function to get instance-level segmentation results. An overview of the whole pipeline is shown in Fig. 2. In the following subsections, we will introduce our input representation, range-aware network architecture and how to use DBSCAN as a post-processing to get instance-level segmentation results.

A. Input Representation

3D point clouds are unstructured data while the standard neural networks perform discrete convolution operations on grids. Thus, several methods have been proposed to encode point

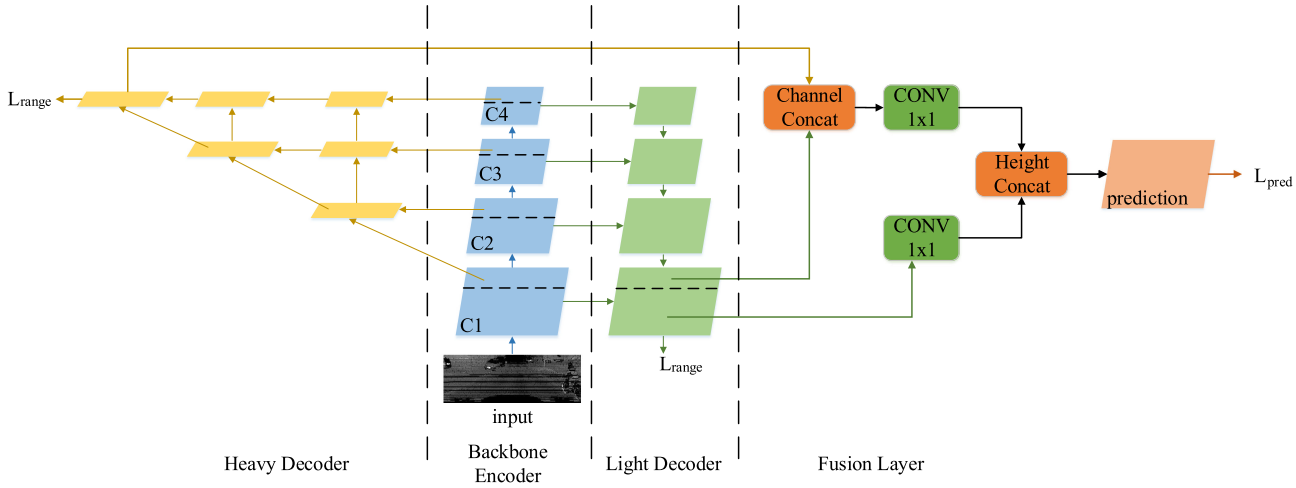


Fig. 3. The architecture of the proposed range-aware network.

clouds into a suitable format. In which, the 3D voxel grids are one possible solution. However, 3D convolution operations are computational intensive, and the sparse voxel grids will lead to lots of unnecessary computations. The 2D birds' eye view is another solution, but leads to information loss. Instead, we use range images to represent the point clouds. The range image is a 2D dense image-like data format without the information loss, and does not need hand-crafted parameters during the format conversion.

The point clouds are converted to range images as following. First, project the points onto a spherical coordinate system with the grid-based representation as

$$\theta = \arctan\left(\frac{z}{\sqrt{x^2+y^2}}\right), \hat{\theta} = \lfloor \frac{\theta}{\Delta\theta} \rfloor$$
$$\phi = \arctan\left(\frac{y}{x}\right) \qquad , \hat{\phi} = \lfloor \frac{\phi}{\Delta\phi} \rfloor \quad (1)$$

where θ and ϕ are an azimuth angle and an elevation angle respectively. $\Delta\theta$ and $\Delta\phi$ are resolutions for the discretization and $(\hat{\theta}, \hat{\phi})$ denotes the position of 2D spherical grids. Applying (1) on each point, we can get a 3D $H \times W \times K$ tensor. In this paper, we use the KITTI dataset [19], [20] collected by Velodyne HDL-64E S3, which has 64 laser beams, $(H = 64)$. Also, the horizontal angular resolution is 0.1728° and the annotations are only available in the 90° front view, $(W = 512)$. K is the number of features, which is 3 in this paper, $(K = 3)$, including intensity, range measurements and occupancy. The occupancy channel indicates whether the grids contain points. The visualization of a range image representation is shown in Fig. 2.

B. Range Distribution on Range Images

In order to get 360° view of scenes, the LiDAR sensors are mostly placed on the top of vehicles. As a consequence, the vertical view is asymmetric, where only few laser beams are emitted to the upper part of a scene. The lower position lasers can only sample the objects in a much shorter range. Fig. 1 shows the laser ID that can detect "car" at different ranges. As shown in the figure, only the top part of lasers can detect objects in the whole range, especially if they are far away. Besides, the far objects only occupy few pixels on range images due to the lower density of points associated to objects at larger distances. Based on the observation, we propose the range-aware framework that uses different decoders for different parts of the range image to get higher accuracy and speedup as well.

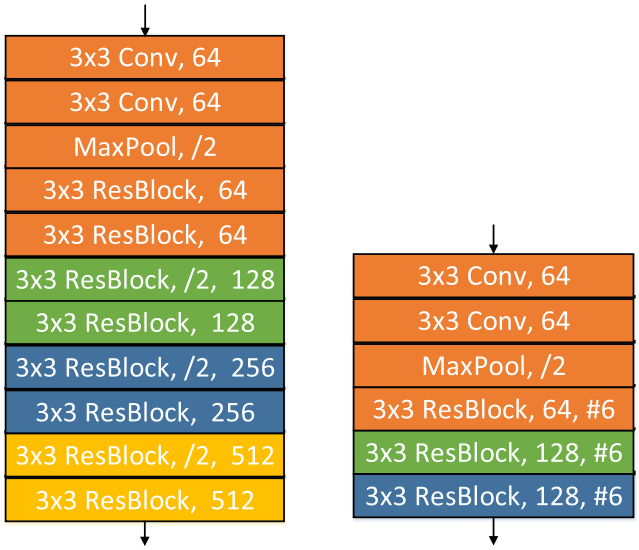
C. Network Architecture

RangeSeg as shown in Fig. 3 is a fully-convolutional network with one backbone encoder, two decoders, a fusion layer and a simple post-processing for final instance-level segmentation prediction. The outputs of network are the same size as inputs to get higher accuracy segmentation results.

For the encoder backbone, this paper uses two different backbones for testing. The first one is the modified version of ResNet-18 [21]. The original ResNet-18 performs two down-sampling steps at the beginning of the network, which are removed in this paper to enable computations on original resolution feature maps for better accuracy, as shown in Fig. 4(a). The other one is based on LaserNet [22] that performs 3D object detection on range images. In which, the residual blocks are also used to get better performances with deeper layers and fewer channels. Although the LaserNet backbone is much deeper, its parameters of kernels are fewer due to the fewer channels as shown in Fig. 4(b).

For the target KITTI dataset, the input representation is $64 \times 512 \times 3$. Thus, we only perform the stride and downsample operations on the vertical dimension to minimize information loss.

The range-aware decoders use the same feature maps from the backbone network for the heavy and light decoders to exploit the different range distribution of range images. The *heavy decoder* only predicts the results for top of images, where the small and far objects locates and needs the deeper network aggregation for accurate predictions. The heavy decoder uses the 'deep' skip connections inspired by DLA [23], where the high level feature maps will be upsampled and aggregated with lower level ones repeatedly. In this paper, instead of using the tree-structured



(a) ResNet18 (b) LaserNet

Fig. 4. The architecture of the modified ResNet18 and LaserNet, where # denotes the number of blocks.

DLA, a much dense concatenation is used as shown in Fig. 5(b) since this decoder only processes the top rows of the range maps. The *light decoder* predicts the results of whole images with low computational cost. The light decoder uses the 'shallow' skip connections like U-Net [13] that only concatenates the feature maps once as shown in Fig. 5(a). It has low computational cost while preserves the information of different range features.

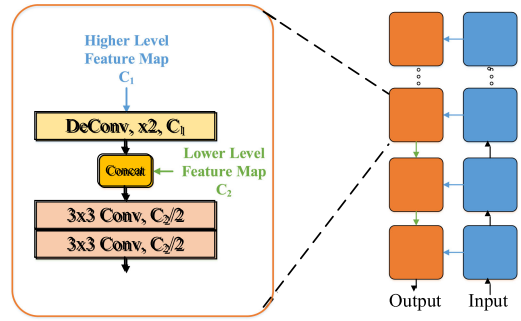
The predictions from the light and heavy decoders are fused together for final results. However, these two predictions have different spatial sizes, which prohibits a direct fusion. For a smooth fusion, the whole output of the heavy decoder is concatenated with the top same size output from the light decoder. Then the concatenated result changes its channel numbers by 1x1 convolutions to match the channel numbers of the bottom part from the light decoder, as shown in Fig. 3. Finally, these two parts are concatenated along the height dimension for the final results. In order to get the predictions on the same resolution, a transpose convolution layer is applied after the fusion layer. In our experiments, we apply the heavy decoder on the top 16 rows of range images based on the observations of the KITTI dataset.

D. Training Strategy

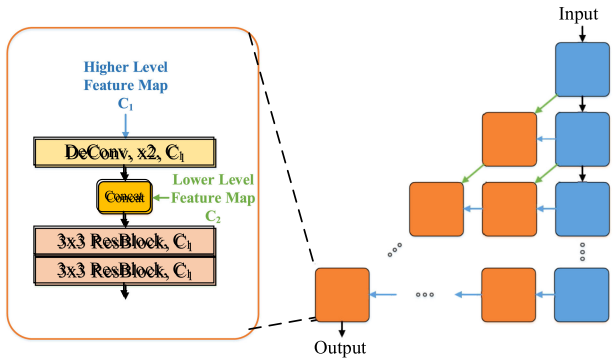
RangeSeg uses the common multi-class cross entropy loss and Lovász-softmax loss [24] to train the network. Cross-entropy loss is used for the pixel-wise classification loss on the classification output p with the target y. If k denotes the channels of the input images, cross-entropy loss is defined as

$$Loss_{xent}(\boldsymbol{p}, \boldsymbol{y}) = \frac{1}{HWK} \sum_{i=0}^{H} \sum_{j=0}^{W} \sum_{k=0}^{K} -y_{i,j,k} \log p_{i,j,k} \quad (2)$$

However, cross-entropy loss is not directly related to intersection over union (IoU). Therefore, Lovász-softmax loss, which is a Lovász extension of the Jaccard index, is applied to regularize



(a) The light decoder



(b) The heavy decoder

Fig. 5. The architecture of the heavy and light decoders. The blue block is the feature extractor and the orange block is the up-sample module. The black arrow denotes the normal path of the feature extractor. The blue one denotes the input of the higher level feature maps. The green one denotes the input of the lower level feature maps. The details of up-sample module are shown in the left part.

the network. If c denotes the class, Lovász-softmax loss is defined as

$$m_{i,j}(\boldsymbol{p}, \boldsymbol{y}, c) = \begin{cases} 1 - p_{i,j}(c), & \text{if } c = y_{i,j} \\ p_{i,j}(c), & \text{otherwise} \end{cases} \quad (3)$$

$$L_{Lovász}(\boldsymbol{p}, \boldsymbol{y}) = \frac{1}{C} \sum_{c=0}^{C} \overline{\Delta_{J_c}}\left(\boldsymbol{m}\left(\boldsymbol{p}, \boldsymbol{y}, c\right)\right) \quad (4)$$

where $\overline{\Delta_{J_c}}$ is the surrogate function of Jaccard loss (Δ_{J_c}). Lovász-softmax loss is IoU-aware and helps solve the data imbalance. Therefore, the loss function, $L(\boldsymbol{p}, \boldsymbol{y})$, is defined as

$$L(\boldsymbol{p}, \boldsymbol{y}) = L_{xent}(\boldsymbol{p}, \boldsymbol{y}) + \lambda_{Lovász} L_{Lovász}(\boldsymbol{p}, \boldsymbol{y}) \quad (5)$$

with the Lovász weighted term $\lambda_{lovász}$. Therefore, the loss of the prediction part from the fusion layer as in Fig. 3 is defined as

$$L_{pred} = L(\boldsymbol{p}_{pred}, \boldsymbol{y}_{pred}) \quad (6)$$

The range-aware decoder loss combines the results from heavy and light decoders directly as a regularization term instead of computing loss on results of the fusion layer alone, which is

defined as

$$L_{range} = L(\boldsymbol{p}_{light}, \boldsymbol{y}_{light}) + L(\boldsymbol{p}_{heavy}, \boldsymbol{y}_{heavy}) \quad (7)$$

The total network loss is defined as

$$L_{total} = L_{pred} + \lambda_{range} L_{range} \quad (8)$$

with a range-aware weighted term λ_{range}. In our experiments, we set both $\lambda_{lovász}$ and λ_{range} as 1.

We use the super convergence strategy [25] as our learning rate scheduler, which has higher and dynamic learning rate for fast convergence.

E. Resolution Weighted Distance Function for Instance Segmentation

To the best of authors' knowledge, none works have used 3D information to segment instances. Unlike objects in the 2D RGB images, the 3D objects will not be overlapped in the 3D space domain, which will make it much easier to segment different objects. Therefore, this paper uses the density-based spatial clustering applications with noise (DBSCAN) [3] for instance clustering, which does not require a pre-defined number of clustering. The clusters are defined by their density. In this paper, DBSCAN takes the points labeled as objects after semantic segmentation as input. Then, the clustering process is applied once for all the objects to save computations for background points and multiple iterations. For the DBSCAN distance function, instead of directly using vanilla distance function, we propose a weighted distance function to deal with the resolution differences of the LiDAR data. The resolution of the vertical dimension is twice fewer than the horizontal one, which leads to sparser vertical values compared with the horizontal ones. Therefore, a resolution weighted distance function is defined as

$$Dis. = \sqrt{2\left(x_2 - x_1\right)^2 + 2\left(y_2 - y_1\right)^2 + \frac{\left(z_2 - z_1\right)^2}{2}} \quad (9)$$

This distance function scales up the coordinates X and Y as well instead of scaling down the coordinate Z alone since scaling down the coordinate z alone cannot help segmentation if the distance is dominated by the horizontal one.

IV. Experimental Results

We evaluate our model on the KITTI dataset and empirically showcase the strengths and weaknesses of the proposed approach. First, we compare the vanilla segmentation frameworks with our range-aware framework using different backbones on the KITTI 3D object detection benchmark [19]. We show that our RangeSeg outperforms on accuracy and inference speed. Second, the comparison of different distance functions on DBSCAN shows that the proposed distance function helps improve the accuracy. Third, we compare RangeSeg with related works on the KITTI raw data, where the accuracy gets lots of improvement with the same inference speed. Fourth, we implement RangeSeg on NVIIDA JETSON AGX Xavier, which gets real time performance even on such small embedded system. Finally, the experiment results on synthetic foggy KITTI data [26] show that our approach is still robust in the foggy weather. The evaluation metric used in this paper is mean class IoU (mIoU).

A. KITTI 3D Object Detection Benchmark

a) Implementation Details: We encode the front 90° into a $64 \times 512 \times 3$ tensor with 3 features: intensity, range and occupancy. The value range of intensity is already within [0,1], and occupancy is either 0 or 1. Therefore, we normalize the range features to be within [0, 1]. In addition, the random horizontal flip with probability 0.5 is applied as the augmentation. For ground truth, we treat the points in the bounding boxes as the objects while the remains are background due to the limitations of the KITTI annotations.

b) Range-Aware Framework: Table I summarizes the comparisons between the vanilla segmentation frameworks and range-aware framework. For ResNet based backbone, the range-aware ResNet18 leads the ResNet18-UNet by 3.9%. The improvements are more significant on small objects such as pedestrians and cyclists by 3.2% and 6.5%, respectively. The range-aware ResNet18 even outperforms ResNet34-UNet by 0.3%. For LaserNet based backbone, the range-aware LaserNet leads LaserNet-DLA by 1.9%. Moreover, the results on top 16 rows of range images show that RangeSeg helps improve detection of the far and small objects since the far objects only lie in the top of images.

c) Inference Speed: Table I shows the inference speed on the Nvidia TITAN Xp. The range-aware ResNet18 improves 3.9% than ResNet18-UNet with only 27% fps loss, while improves 0.3% than ResNet34-UNet with extra 7% speedup. Range-aware LaserNet improves 1.9% of mIoU than LaserNet-DLA with extra 42% speedup since the heavy decoder helps improvements on mIoU with computation overhead, but the low complexity light decoder helps overcome the problems.

d) Ablation Study: Table II shows the ablation study on several design parameters. For loss function, the combination of cross-entropy loss (xent) and Lovász-softmax loss gets the best results because the Lovász-softmax loss is unstable alone even with its direct optimization on mIoU. In addition, both the data normalization and augmentation help improve the accuracy by 3.4%. Further regularized with the range-aware loss helps optimize the networks to higher accuracy by 3.9%.

e) Impact of λ_{range} and $\lambda_{lovász}$: This subsection shows the ablation study of λ_{range} and $\lambda_{lovász}$. For λ_{range}, we use the range-aware LaserNet with the same training strategy to see how λ_{range} improves the accuracy. We train the model with different values of λ_{range} with as $\lambda_{lovász}$ set to 1.0. We can find that the accuracy of the large objects is not affected by the range loss as shown in Table III. However, the range loss can definitely improve the accuracy of the small objects compared with baseline. The accuracy of the cyclist can be even improved by 8% by setting λ_{range} to 0.1 when compared with no range loss. Similarly, Table IV shows the tuning result of $\lambda_{lovász}$ along with λ_{range} set to 1.0. The accuracy of the cyclist has been improved by 8.2% by setting $\lambda_{lovász}$ to 0.01 when compared with baseline in Table IV.

TABLE I
RESULT COMPARISON ON THE KITTI 3D OBJECT DETECTION BENCHMARK. RESNET-UNET USES RESNET AS THE ENCODER BACKBONE WITH THE UP-SAMPLE LAYERS AS UNET. LASERNET-DLA USE LASERNET AS THE ENCODER BACKBONE WITH THE FULLY DLA-LIKE FEATURE AGGREGATION

Architecture	IoU(%)						FPS
	Car	Pedestrian	Cyclist	Overall Mean	Top 16 Rows Mean	Lower 48 Rows Mean	
UNet	75.9	66.7	47.4	63.3	62.9	63.6	249.8
ResNet18-UNet	74.8	65.3	46.0	62.0	62.4	61.3	**122.9**
ResNet34-UNet	76.0	67.2	**53.5**	65.6	65.2	**65.6**	76.6
Ours(ResNet18)	**76.7**	**68.5**	52.5	**65.9**	**67.3**	64.1	89.2
LaserNet-DLA	77.3	69.1	51.7	66.0	67.6	64.0	77.0
Ours(LaserNet)	**77.8**	**70.2**	**56.2**	**67.9**	**69.4**	**66.0**	**109.1**

TABLE II
ABLATION STUDY ON THE LOSS FUNCTION, DATA PRE-PROCESSING AND RANGE-AWARE LOSS

Loss	Norm.	Aug.	Range	mIoU(%)
xent	-	-	-	45.8
focal	-	-	-	20.6
lovász	-	-	-	39.6
focal+lovász	-	-	-	56.7
xent+lovász	-	-	-	**59.1**
xent+lovász	+	-	-	59.8
xent+lovász	+	+	-	**63.2**
xent+lovász	+	+	no $\text{loss}_{\text{range}}$	62.0
xent+lovász	+	+	$\text{loss}_{\text{range}}$	**65.9**

TABLE III
RESULT COMPARISON OF DIFFERENT VALUES FOR λ_{range}

λ_{range}	IoU(%)			
	Car	Ped.	Cyclist	Mean
0.00 (baseline)	74.7	64.2	47.2	62.0
0.01	78.2	70.0	45.7	64.6
0.05	78.1	69.3	50.2	65.9
0.10	77.5	70.3	55.6	67.8
0.50	77.5	69.8	55.0	67.4
1.00	77.8	70.7	52.3	66.9

TABLE IV
RESULT COMPARISON OF DIFFERENT VALUES FOR $\lambda_{lovász}$

$\lambda_{lovász}$	IoU(%)			
	Car	Ped.	Cyclist	Mean
0.00 (baseline)	80.1	62.4	50.8	64.5
0.01	79.8	70.7	59.0	69.8
0.05	79.6	70.9	57.8	69.4
0.10	79.0	71.0	58.1	69.4
0.50	78.5	71.1	55.2	68.3
1.00	77.6	70.8	52.6	67.0

B. Instance-Level Segmentation

a) Implementation Details: For DBSCAN parameters, we choose $minPts$ value as twice the minimum number of points. Also, we choose ϵ value as the half of the average object size. After analyzing the data, we can see that the minimum number of points is about 3.5 in Fig. 1. Thus,we choose 7 for $minPts$ and 0.7 for ϵ. We have tested different distance functions: the original Euclidean distance function, (A), coordinate Z only scaling, (B), and the proposed function as (9), (C).

TABLE V
RESULT COMPARISON OF INSTANCE SEGMENTATION

Distance	IoU(%)			
	Car	Ped.	Cyclist	Mean
Semantic	*77.8*	*70.2*	*53.5*	*67.2*
(A)	75.1	53.7	50.5	59.8
(B)	**75.2**	54.0	49.2	59.5
Ours (C)	74.6	**58.1**	**52.7**	**61.8**

TABLE VI
RESULT COMPARISON WITH THE RELATED WORK

Architecture	IoU(%)				FPS
	Car	Ped.	Cyc.	Mean	
SqueezeSeg [1]	64.6	21.8	25.1	37.2	122.2
PointSeg [8]	67.4	19.2	32.7	39.8	106.7
SqueezeSegV2 [9]	73.2	27.8	33.6	44.9	79.7
RIU-Net [12]	62.5	22.5	36.8	40.6	**249.8**
Ours(LaserNet)	**75.6**	**49.5**	**50.1**	**58.4**	109.1
Ours(ResNet18)	75.5	44.1	49.5	**56.3**	89.2

b) Evaluation results: Table V shows the instance-level segmentation evaluation results. The Z-only scaling distance function (B) is insufficient to segment instance objects, whose performance is almost the same as the original function (A). In contrast, our proposed weighted distance function (C) gets 2% of mIoU improvement. This post-processing step consumes only 16.48 ms with standard deviation 24.13 ms on i9-7900X. The visualization results of RangeSeg is shown in Fig. 6. The proposed method can accurately predict the small and far objects when compared with the previous RIU-Net.

C. Result Comparison on the KITTI Raw Data

Table VI shows the result comparison between RangeSeg and other related works on the KITTI raw data. We follow [1] to split the KITTI raw data [20]. We choose the $\lambda_{lovász}$ as 0.05 and λ_{range} as 0.5 which is the best setting in the KITTI raw data. RangeSeg outperforms other state-of-the-art methods. For more detailed comparisons, RangeSeg gets significant improvements on small objects. Also, RangeSeg gets 18.6% improvements of mIoU compared to PointSeg with almost the same inference speed.

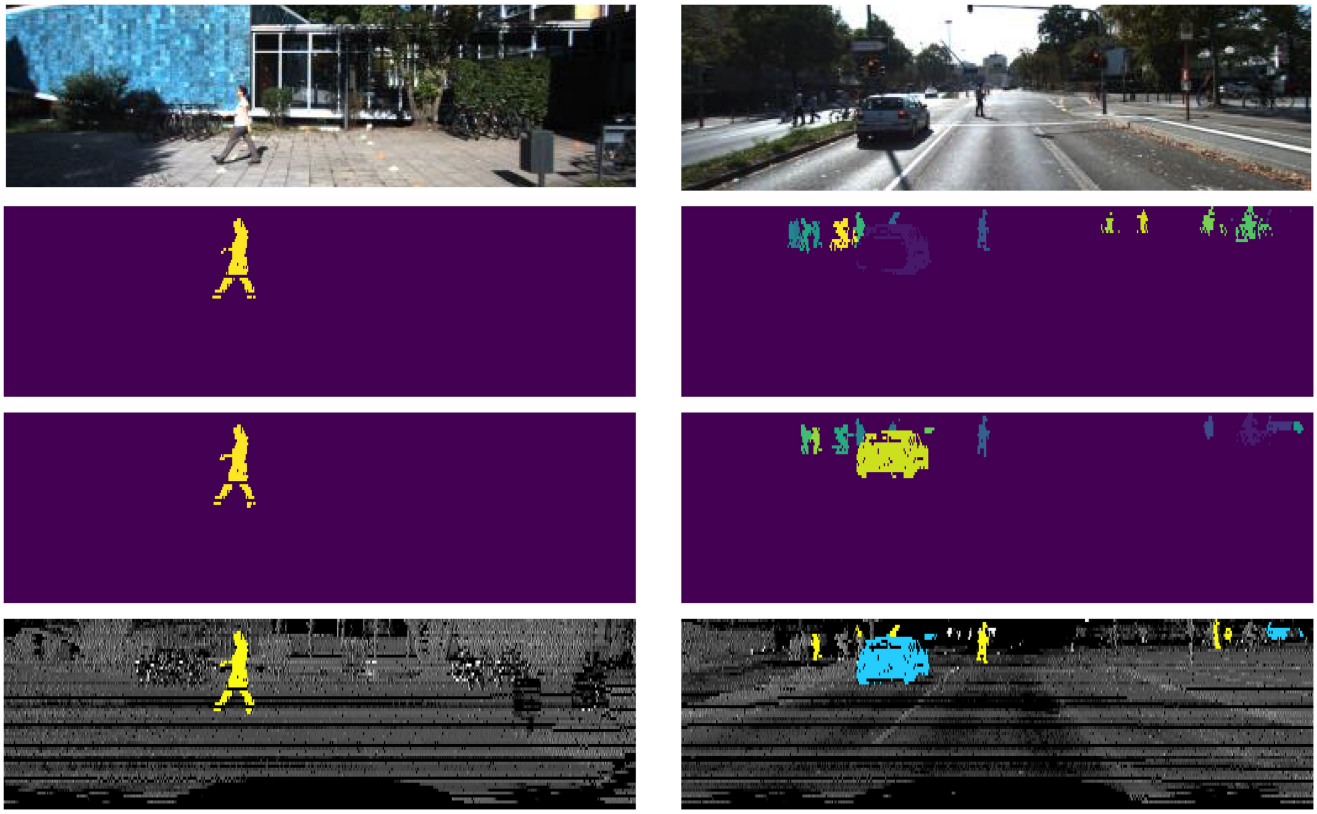


Fig. 6. The visualization results of instance segmentation for different scenes. For each scene, the images from top to bottom are raw images, ground truth, prediction by RangeSeg, prediction by RIU-Net. Though RIU-Net has good good real time performance and good accuracy for simple scenes, its false negative results become obvious for small or far objects when compared to the proposed approach.

TABLE VII
PROCESS TIME OF EACH OPERATION ON NVIDIA JETSON AGX XAVIER

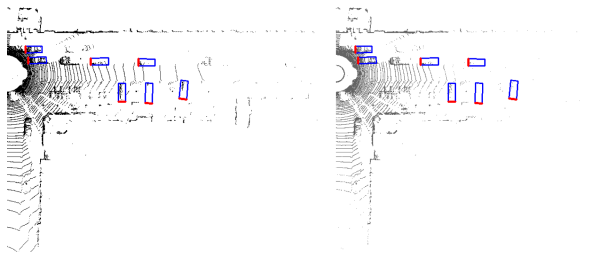
Process	Time Avg.(ms)	Time Std.(ms)
Encoding	4.75	0.97
Model	13.75	0.26
DBSCAN	38.22	50.92

D. Real Time Implementation on Nvidia AGX Xavier

The range-aware LaserNet is implemented on NVIDIA JETSON AGX Xavier for embedded system applications. In our experiment, we use TensorRT FP16 to optimize our framework. The processing time of data encoding, models and post-processing is summarized in Table VII. It shows that the fps is about 19 Hz with TensorRT FP16 optimization, which is much higher than the 10 Hz capture frequency of LiDAR sensors in the KITTI dataset.

E. Synthetic Foggy KITTI Dataset

a) Foggy Dataset Details: An autonomous driving application should be robust in every environment. In this experiment, we use our range-aware LaserNet on the synthetic foggy KITTI dataset [26] with different visibilities. The visibility is defined as the maximum range that the objects can be visually seen by human. However, the range of objects can be detected by LiDAR is half of visibility since the point clouds are detected by reflection pulses. The experiments use visibility at 70 m and 40 m, which is much more extreme than the worst visibility at 150 m in SFSU [27]. The foggy weather will result in false alarm measurements for range within 2 meters mostly, and low intensity reflections for all points. Fig 7 shows the birds' eye view of the LiDAR point clouds in different weathers. The figure shows that there is a blind zone due to car roof mounted LiDAR. The points with wrong range measurements lead to an inner circle as in Fig 7(b). Therefore, a simple defog method is applied that removes the points shorter than the 2 meters range.



(a) Clear weather. (b) Visibility at 70 meters.

Fig. 7. The birds' eye views of different weathers. False measurements will result in an inner circle in the foggy LiDAR point clouds.

TABLE VIII
RESULTS OF THE RANGE-AWARE LASERNET TESTED ON A SYNTHETIC FOGGY DATA

Model	Aug.	Test Data	IoU(%)			
			Car	Ped.	Cyc.	Mean
3 channels	-	clear	77.2	70.2	56.2	67.9
3 channels	-	70m	52.7	62.3	11.2	32.0
+defog(A)	-	70m	60.3	37.9	22.2	40.1
2 channels	-	clear	76.4	65.3	45.5	62.4
+defog(B)	-	70m	52.6	62.7	43.2	52.9
(A)+(B)	-	70m	58.2	59.7	44.5	54.1
2 channels	70m	clear	72.2	66.4	47.6	62.1
		70m	67.7	66.7	47.5	60.6
3 channels	70m	clear	73.2	69.3	49.2	63.9
		70m	68.9	68.6	49.3	62.3
		40m	62.7	64.2	44.8	57.2
3 channels	40m	clear	72.3	68.8	47.2	62.8
		40m	66.8	68.0	47.4	60.7
		70m	68.3	67.2	44.0	59.8

b) Evaluation Results: Table VIII shows the evaluation results. When directly testing the model on the foggy data, the accuracy is significantly degraded to 32.0% of mIoU while the simple defogging (A) only get 8.1% improvements. The reason is that the distribution of the intensity channel in the foggy weather is different from that of the clear weather. Therefore, we train a new model that only contains 2 channels without intensity (denoted as 2 channels). After defogging on the 2 channels model (B), the accuracy is 52.9%. Combining (A) and (B) gets a robust accuracy at 54.1% of mIoU.

Next, we take foggy data as data augmentation. The 3 channels model gets higher mIoU with augmentation. The mIoU is more than 60% when augmented or tested at 70 m or 40 m, respectively. Also, the model trained on visibility at 70 m gets 57.2% of mIoU on visibility at 40 m while the model trained on visibility at 40 m gets 59.8% of mIoU on visibility at 70 m. This indicates the LiDAR sensors are robust even in different weather conditions.

V. CONCLUSION

By exploiting the LiDAR data distribution in the autonomous driving application, this paper proposes a range aware instance segmentation network that can achieve high accuracy with a heavy decoder and high speed with a light decoder. The heavy decoder is applied to the top of the range image where the far and small objects lie in for accurate detection. The light decoder is applied to the whole range image for low complexity computation. The proposed weighted distance metric helps segment instances with a simple post-processing. Compared with previous works, our range-aware framework is simple, efficient, fast and has great applications on autonomous driving in different weathers. While we have only conducted the experiments on two backbones, further experiments on state-of-the-art models are a potential area for improvements. Applying this range aware framework to other LiDAR tasks is another interesting future research.

REFERENCES

[1] 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.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[3] M. Ester *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," *Kdd*, vol. 96, no. 34, pp. 226–231, 1996.

[4] 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.*, 2018, pp. 4490–4499.

[5] B. Yang, W. Luo, and R. Urtasun, "Pixor: Real-time 3D object detection from point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7652–7660.

[6] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 770–779.

[7] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[8] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu, "Pointseg: Real-time semantic segmentation based on 3D lidar point cloud," 2018, *arXiv:1807.06288*.

[9] 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. Int. Conf. Robot. Automat.*, 2019, pp. 4376–4382.

[10] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size," 2016, *arXiv:1602.07360*.

[11] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[12] P. Biasutti, A. Bugeau, J.-F. Aujol, and M. Brédif, "Riu-net: Embarrassingly simple semantic segmentation of 3D lidar point cloud," 2019, *arXiv:1905.08748*.

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Interv.*, 2015, pp. 234–241.

[14] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2980–2988.

[15] 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.

[16] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.

[17] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on x-transformed points," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 820–830.

[18] F. Zhang *et al.*, "Instance segmentation of lidar point clouds," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 9448–9455.

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

[20] 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.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[22] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "LaserNet: An efficient probabilistic 3D object detector for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 12 677–12 686.

[23] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2403–2412.

[24] M. Berman, A. R. Triki, and M. B. Blaschko, "The Lovász-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.

[25] L. N. Smith and N. Topin, "Super-convergence: Very fast training of neural networks using large learning rates," *Artif. Intell. Mach. Learn. Multi-Domain Operations Appl.*, vol. 11006, 2019, Art. no. 1100612.
[26] T. H. Sang, S. Tsai, and T. Yu, "Mitigating effects of uniform fog on SPAD lidars," *IEEE Sens. Lett.*, vol. 4, no. 9, Sep. 2020, Art. no. 3501404.
[27] C. Sakaridis, D. Dai, and L. Van Gool, "Semantic foggy scene understanding with synthetic data," *Int. J. Comput. Vis.*, vol. 126, no. 9, pp. 973–992, Sep. 2018.

Tzu-Hsuan Chen received the B.S. and M.S. degrees in electric engineering and electronic engineering from National Chiao Tung University, Hsinchu, Taiwan, in 2017 and 2019, respectively. His current research interests include VLSI signal processing and deep learning.

Tian Sheuan Chang (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronic engineering from National Chiao Tung University (NCTU), Hsinchu, Taiwan, in 1993, 1995, and 1999, respectively.

From 2000 to 2004, he was a Deputy Manager with Global Unichip Corporation, Hsinchu, Taiwan. In 2004, he joined the Department of Electronics Engineering, NCTU, where he is currently a Professor. In 2009, he was a Visiting Scholar with IMEC, Belgium. His current research interests include system-on-a-chip design, VLSI signal processing, and computer architecture.

He was the recipient of the Excellent Young Electrical Engineer from the Chinese Institute of Electrical Engineering in 2007 and the Outstanding Young Scholar from Taiwan IC Design Society in 2010. He is actively involved in many international conferences as an organizing committee or technical program committee member.