RGB and LiDAR fusion based 3D Semantic Segmentation for Autonomous Driving

Khaled El Madawy[1], Hazem Rashed[1], Ahmad El Sallab[1], Omar Nasr[2], Hanan Kamel[2] and Senthil Yogamani[3]
[1]Valeo R&D Cairo, Egypt [2]Cairo University [3]Valeo Visions Systems, Ireland
{khaled.elmadawy, hazem.rashed, ahmad.el-sallab, senthil.yogamani}@valeo.com

***Abstract*— LiDAR has become a standard sensor for autonomous driving applications as they provide highly precise 3D point clouds. LiDAR is also robust for low-light scenarios at night-time or due to shadows where the performance of cameras is degraded. LiDAR perception is gradually becoming mature for algorithms including object detection and SLAM. However, semantic segmentation algorithm remains to be relatively less explored. Motivated by the fact that semantic segmentation is a mature algorithm on image data, we explore sensor fusion based 3D segmentation. To the best of our knowledge, this is the first attempt at RGB and LiDAR based 3D segmentation for autonomous driving. Our main contribution is to convert the RGB image to a polar-grid mapping representation used for LiDAR and design early and mid-level fusion architectures. Additionally, we design a hybrid fusion architecture that combines both fusion algorithms. We evaluate our algorithm on KITTI dataset which provides segmentation annotation for cars, pedestrians and cyclists. We evaluate two state-of-the-art architectures namely SqueezeSeg and PointSeg and improve the mIoU score by 10% in both cases relative to the LiDAR only baseline.**

I. INTRODUCTION

Autonomous driving is a complex task where a robotic car is expected to navigate with full autonomy in a highly dynamic environment. To accomplish such task, the autonomous vehicle has to be equipped with multiple sensors and robust algorithms that perceive the surrounding environment with high accuracy in a real-time fashion. The first step for perception pipeline is to detect objects from background. Object detection alone is not sufficient for a robot to navigate, there has to be robust classification to determine the type of each object for planning the interaction, especially for complex scenarios like parking [9]. This is a crucial task because the reaction of an autonomous vehicle to a pedestrian that showed up suddenly after occlusion will be completely different compared to a suddenly appearing vehicle for example. Moreover, the algorithms have to estimate the location of external objects within the subsequent frames to be able to take proper action.

From this perspective, 3D semantic segmentation is a critical task for autonomous driving as it simultaneously performs 3D localization and classification of objects as visualized in Fig. 1. Point cloud segmentation has been studied in [4][10][20]. Classical methods used pipelines including segmenting ground from foreground objects, clustering the objects points together and performing

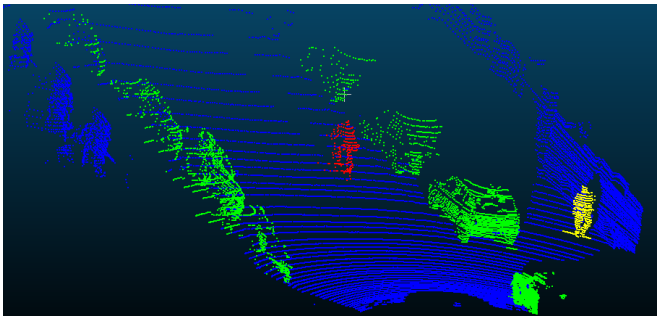


Fig. 1: Visualization of 3D semantic segmentation ground truth in Cartesian coordinates.

classification based on hand-crafted features. Such methods are prone to low performance due to accumulation of error across successive tasks. Moreover, they do not generalize well across different driving environments. Deep learning recently has gained large attention in several tasks including semantic segmentation due to its powerful automatic feature extraction, where it has the ability to find ambiguous relationships between different domains than sometimes cannot be interpreted by humans. In this paper, we adopt end-to-end convolutional neural network (CNN) which performs 3D semantic segmentation. Our approach utilizes both information from two complementary sensors, namely cameras and LiDAR, that are being deployed in recent commercial cars. Camera sensor provides color while LiDAR provides depth information. Recent work in [19][18] provided algorithms to understand semantics from LiDAR only. We build upon this work by fusing the color information with LiDAR data for 3D semantic segmentation task. We implement two fusion algorithms to study the impact of the addition of color information. The first approach is early-fusion in which we fuse the information as raw data before feature extraction. The second approach is mid-fusion in which we use CNN to extract features from two different modalities then perform the fusion feature level.

The main contributions of this paper are as follows:

1) Construction of CNN based early and mid fusion architectures to systematically study the effect of color information on 3D semantic segmentation.
2) Evaluation on two state-of-the-art algorithms, namely SqueezeSeg and PointSeg, and significant improve-

ments via fusion with camera data was achieved.
3) Design of RGB fusion representation and hybrid fusion strategy for improving performance.

The paper is organized as follows. Section II reviews related work in semantic segmentation and point cloud segmentation. Section III discusses the various proposed architectures of our algorithm. Section IV details the dataset used, the experimental setup and discusses the results obtained. Finally, Section V provides concluding remarks and future work.

II. RELATED WORK

A. Semantic Segmentation for Images

A detailed survey for image semantic segmentation for autonomous driving is presented in [17] and efficient design techniques were discussed in [2]. We briefly summarize the main methods used pixel-wise classification. In [8] patch-wise training was used for classification while in [6] the input image was fed into Laplacian pyramid to extract hierarchical features. A deep network was used in [8] to avoid further post processing. In [13] [14] [1] end-to-end methodology was adopted for semantic segmentation. In [13], the network learned heatmaps that were upsampled to generate the classification output. Multiple skip connections between the encoder and decoder were introduced to avoid losing resolution. Unlike patch-wise methods, this approach uses the full image to generate dense semantic segmentation outputs. In [1] an encoder-decoder network was deployed where the feature maps were upsampled utilizing the information of the kept indices from the corresponding encoder layer. There has been several improvements on color based segmentation but typically they don't make use of depth information.

B. Semantic Segmentation For Point Clouds

Relative to image segmentation, there is very little literature on point cloud semantic segmentation. Majority of 3D object detection literature focuses on 3D bounding box detection but it is not the best representation of objects in many scenarios. This is analogous to 2D segmentation which provides a better representation than 2D bounding box in images. The main challenge of LiDAR perception in general is the sparse nature of point cloud data. It increases the appearance complexity drastically. Thus there are many different approaches to simplify this representation including Voxels, bird-view and polar-grid map. Voxels are clustered point-cloud data and they still suffer from sparsity. Bird-view is the simplest representation which simplifies point cloud into a flat plane. However, this loses the height information and thereby important appearance features necessary for detecting objects.

Bird-View Lidar semantic segmentation was performed in [5], where the LiDAR points are projected to a grid xy-plane and a semantic classification is applied on each grid. Other approaches divided the space into voxels[21] with a predefined resolution, projected the point cloud inside these voxels, and performed voxel-level classification. However, the main problem of voxelization is the required resources in memory and processing power to represent a huge volume covered by a Lidar sensor due to considering the occupied and non-occupied voxels. On the other hand, there were other approaches that performed semantic segmentation using 3D point cloud like PointNet[15], PointNet++[15] which considered the point cloud as an un-ordered set of points. This approach provides invariance to arbitrary viewpoint but in autonomous driving the specific perspective is important. It also doesn't take into consideration the structure of LiDAR scanning. Recently, Squeezeseg [19] tackled the problem with polar-grid map which is discussed in detail in Section III A. It uses the spherical representation of Lidar Point cloud which explicitly models the scanning process and it provides a relatively dense 2D plane. This has provided the opportunity to leverage image-based semantic segmentation architectures.

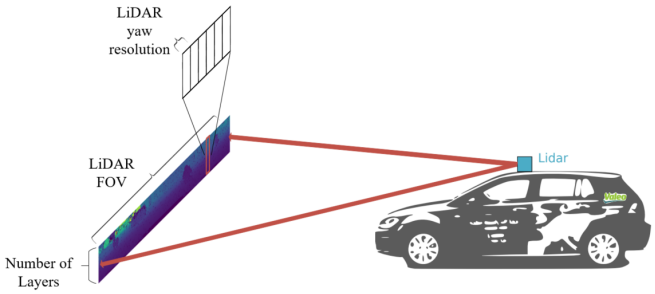


Fig. 2: Illustration of LiDAR Polar Grid Map representation.

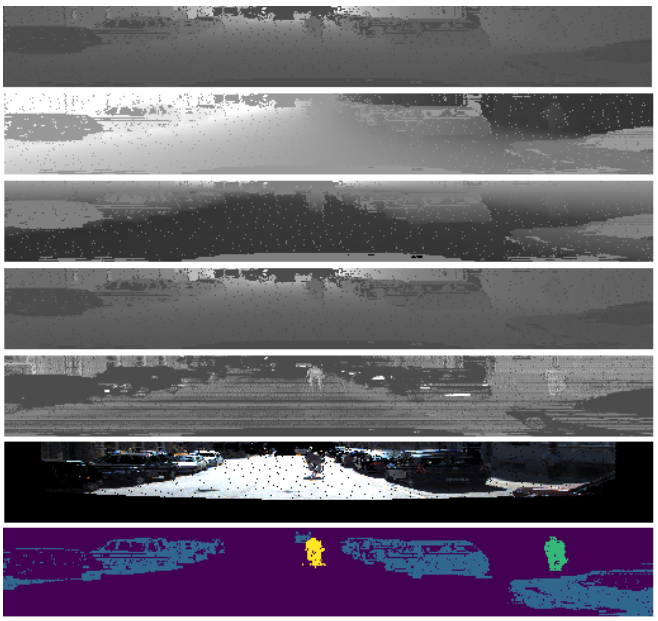
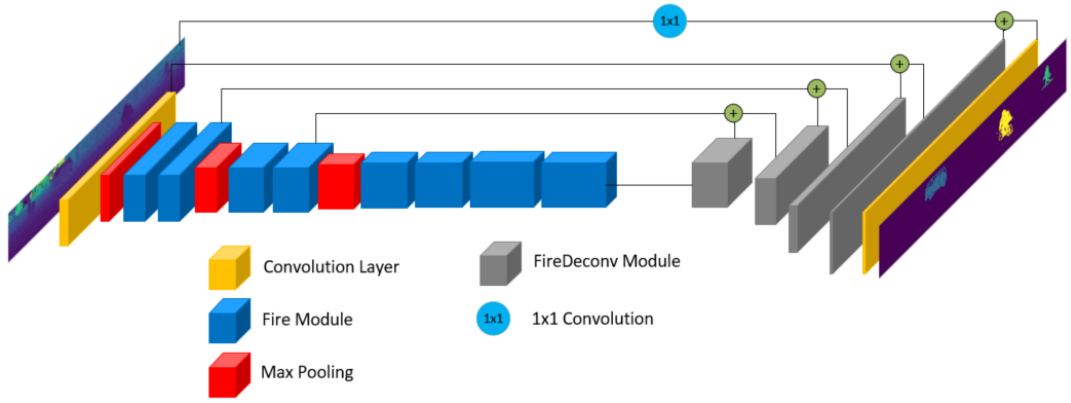


Fig. 3: Input frame and ground-truth tensor. **Top to bottom:** X, Y, Z, D, I, RGB and Ground Truth.

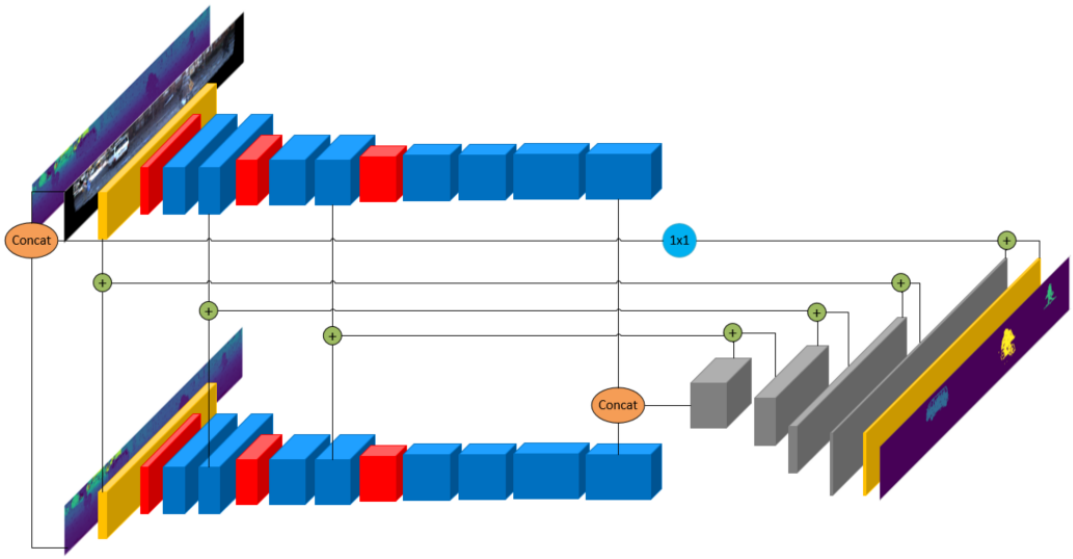
III. PROPOSED ALGORITHM

A. Polar Grid Map representation for LiDAR and cameras

LiDAR sensor can be modeled as a sorted set of points, where measurement for n number of rays is captured at an



(a) LiDAR baseline architecture based on SqueezeSeg [19].



(b) Proposed RGB+LiDAR mid-fusion architecture

Fig. 4: Semantic Segmentation network architectures used in the paper. (a) shows the baseline SqueezeSeg based unimodal baseline architecture. The architecture remains the same for early fusion except for the change in number of input planes. (b) shows the proposed mid-fusion architecture.

instant and each ray represents a scan point in a layer. This is performed at a certain yaw angle and then the LiDAR rotates to the next yaw angle to make another measurement. This is continued to obtain a full 360° perception of the scene around the vehicle. The point cloud comprises of a large number of points $N = \textit{(Number of layers)} \times \textit{(Number of points per layer)}$ where the number of points per layer $= \textit{(Yaw FOV)} \times \textit{(Yaw resolution of the firing angle)}$. This sparse 3D data can be reshaped into a 2D polar grid where the height is the number of layers and the width is the number of points detected per layer as shown in Fig. 2. Front-view is the most important part for driving and a subset of the full LiDAR scan for front-view is constructed for a field-of-view (FOV) of 90° with a yaw resolution of 0.175°, 64 layers and 5 features (X, Y, Z, I, D) as shown in Fig. 1. X, Y, Z are the Cartesian coordinates of the Lidar sensor, I is the intensity of the scan point and D is its depth. This representation is called Polar Grid Map (PGM) and the size of this tensor is 64x512x5.

Camera 2D image is obtained by projection of the light rays onto the image plane. Typically, cameras may be have a different FOV compared to LiDAR. For example, LiDAR cannot view the near-field of the vehicle. The ideal-pinhole projection model of a camera is also broken by modern lenses which is very pronounced in the case of fisheye lenses. LiDAR is typically projected onto the camera image plane [12]. However this leads to a sparse representation of LiDAR which might lead to sub-optimal models. In this paper, we explore an alternate approach of re-projecting the pixels onto the LiDAR polar-grid map plane which is dense. We project the LiDAR point cloud on the RGB image using relative calibration information. This establishes a mapping between LiDAR scan points and RGB pixels. By using this mapping, we augment three additional features (RGB) to the existing point cloud feature tensor (XYZID), creating a tensor of size 64x512x8 as shown in Fig. 3 and the corresponding

ground truth in a tensor of size 64x512x1 is shown at the bottom of Fig. 3. This representation is also scalable to map multiple cameras around the car which can cover the full 360°. However, it has the disadvantage of not utilizing color pixels which do not have a mapping to a LiDAR point.

B. LiDAR baseline architecture

Our baseline architecture is based on SqueezeSeg [19] which is a lightweight architecture that performs 3D semantic segmentation using LiDAR point cloud only. The network architecture is illustrated in Fig. 4 (a). It is built upon SqueezeNet [11] where the encoder is used until fire9 layer to extract features from the input point cloud. Fire units include 1x1 conv layer which squeezes the input tensor to one quarter of its original depth. Two parallel convolutional layers are followed by a concatenation layer to expand the squeezed tensors. The output is passed to fireDeconv modules to do the upsampling where a deconvolutional layer is inserted between the squeezing and expanding layers. Skip connections are utilized to join the deep and shallow layers in order to maintain the high resolution feature maps and avoid accuracy loss. Finally, the output probability is generated through a softmax layer after final convolutional layer. The input to this architecture is a 5-channel tensor which includes three layers for X,Y,Z that describe the spatial location of each point in the point cloud. The fourth layer is a depth map which shows the polar distance between the LiDAR sensor and the target point. Finally, the fifth layer encodes reflectance intensity of LiDAR beams. We refer to this input as XYZDI. We also implemented another baseline architecture using PointSeg [18] which improves upon SqueezeSeg.

C. Early-fusion architecture

In this architecture, we aim to fuse the input data as raw values which will be processed for joint feature extraction by the CNN. The same methodology described in the baseline network architecture is used, however the input tensor in this case consists of 8 channels which are the original XYZDI in addition to 3 RGB layers. The advantage of this architecture is that the network has the potential to learn relationships among data and combine them effectively. However, this architecture cannot leverage pre-training on large unimodal datasets like ImageNet [3]. We refer to this architecture as XYZDIRGB and we obtain improved results over the baseline architecture with negligible increase in computational complexity.

D. Mid-Fusion architecture

We construct a mid-fusion architecture where the fusion happens at the CNN encoder feature level as illustrated in Fig. 4 (b). In this case, two separate encoders are constructed for each input modality. Feature extraction is performed on each input separately, then the processed feature maps are fused together using the concatenation layer. This model is computationally more complex than early fusion model as the the number of encoder parameters are doubled. But when it is viewed from the system level, the separate encoders can be leveraged for other tasks in the respective modalities. This model typically provides better performance compared to early fusion [16]. We refer to this architecture as XYZDI + RGB. It was experimentally found that this architecture was not able to effectively fuse the modalities and there was negligible increase in accuracy. We constructed a hybrid of early-fusion and mid-fusion network where we concatenate the RGB channel to LiDAR depth and intensity channels. We obtain significant improvements over the baseline using this approach.

IV. EXPERIMENTS

In this section we provide details about the dataset used and our experimental setup.

A. Datasets

We make use of the KITTI raw [7] dataset which contains 12,919 frames, of which 10128 were chosen as training data, and 2,791 frames were used as validation data. We choose this dataset for multiple reasons. Firstly, it is based on autonomous driving scenes which is the main focus of our work. Additionally, it provides 3D bounding box annotation for multiple classes. Following [19], we divide our classes into three groups, i.e. "Car", "Pedestrian", "Cyclist" and "Background". The "Car" class includes cars, vans and trucks. We focus on these classes because they have the main collision risk for an autonomous vehicle. The points inside each 3D bounding box are labeled with the class provided by the dataset which can be used as annotation for 3D semantic segmentation. We make use of the data split provided by [19] so that it can be compared effectively.

B. Experimental Setup

We make use of PGM data representation with horizontal front FOV of 90°, creating a 3D tensor of $64 \times 512 \times nc$ where nc denotes the number of input channels depending on the experiment at hand. In the baseline experiments, nc is 5 encoding LiDAR data only, and in Early-Fusion nc is 8 encoding RGB layers concatenated to LiDAR channels. In Mid-Fusion nc is 5 in the DIRGB branch and 5 in the LiDAR branch. The output is a 64x512x1 tensor representing classification per polar gird. We used data augmentation by randomly flipping the frames in the y-axis. In all experiments, we set learning rate to 0.01 and the optimizer momentum was set to 0.9. Class-wise Intersection over Union (IoU) is used as the performance metric, and an average IoU is computed over all the classes. Our model is implemented using TensorFlow library. We ran our training and inference on a 1080-ti GPU.

C. Results

Table I (top) shows quantitative evaluation for our approach using SqueezeSeg architecture [19]. XYZDI results are obtained by training the publicly available network [19] using KITTI-raw dataset without fusion. These results serve as a baseline for our experiments for comparative purpose.

TABLE I: Quantitative evaluation on KITTI Raw dataset using SqueezeSeg and PointSeg architectures.

Network Type	Car	Pedestrian	Cyclist	mIoU	Runtime (ms)
SqueezeSeg baseline architecture					
XYZDI	62.2	16.9	21.9	33.7	8
XYZDIRGB	**65.7**	20.2	24.2	36.7	8
XYZDI + DIRGB	65.1	**22.7**	**24.4**	**37.4**	11
PointSeg baseline architecture					
XYZDI	67	18.4	19.12	34.8	9
XYZDIRGB	**68.5**	16.2	**28.8**	**37.8**	9
XYZDI + DIRGB	67.8	**18.6**	26.3	37.6	12

Results of XYZDIRGB show enhanced performance over the baseline with an absolute increase of 3% in mean IoU. XYZDI + DIRGB refers to our proposed algorithm which provides the best performance with an absolute increase of 3.7% in mean IoU. Relative increase in mean IoU is around 10%. Results using PointSeg [18] architecture are reported in Table I (bottom). Early and Mid-Fusion significantly outperform results using LiDAR data only. However, results of early fusion outperformed results of mid fusion. This result is not consistent with the previous experiments with SqueezeSeg and our prior experience on fusion architectures [16]. After careful re-experiments to cross verify the result, we hypothesize that this could be due to the atypical enlargement layer in PointSeg network which is concatenated with the regular convolutional layer features.

It is observed that no-fusion approach had difficulties in inferring classes with small volume. We believe there are three reasons for this. The first one is the unbalanced dataset especially with the proposed split provided by [19] where only 35% of Pedestrian class is used for training, and 65% for testing. In Cyclist class, 63% were used for training, and 37% for testing. On the other hand, the Car class is divided into 78% for training, and 22% for testing. The second reason is the unbalanced classes, where the Car class represents 96% of the annotated dataset, while Cyclist class represents only 1.4% of the annotated data, and pedestrian class represents also around 1.6% of the annotated data. The third reason is the small volume of the instances from the two classes compared to the Car class which minimizes the strong features specific to those classes. We believe these reasons played an important role in the detection problem. However both early or mid-fusion experiments provide enhanced performance over results with LiDAR only. In Pedestrian class we obtained 3.3% and 5.8% respectively in early and mid-fusion. In Cyclist class the mIoU was improved by 2.3% and 2.5% respectively for both fusion approaches. Using PointSeg architecture, we obtained 3% and 2.8% improvement.

Fig. 5 shows qualitative comparison between the results obtained using SqueezeSeg architecture. It is shown that our approach improved the detection of cars, pedestrians and cyclists using early and mid-fusion which are illustrated in the second and third columns. In the first and second rows in Fig. 5, the no-fusion approach classified the cyclist as a pedestrian, where early-fusion provided better accuracy with incorrect classification for the head part. Mid-fusion classified the cyclist correctly, however we notice some false positives at the edges of the cyclist, which we believe to be due to the effect of smoothing effect of convolutional filters. In the third row, Early fusion and mid fusion achieved the best classification of the car. Fig. 6 shows qualitative comparison between the results obtained using PointSeg architecture, where there is improvement in cyclist and Cars.

Due to the light-weight architecture, the performance of our algorithm is real-time taking around 10 ms per scan. Early fusion nearly takes the same execution time taken by the no-fusion approach, while the Mid-fusion costed 3 ms more in both architectures. Runtime details are tabulated in last column of Table I.

V. CONCLUSIONS

In this paper, we explored the problem of leveraging color information in addition to LiDAR point clouds for 3D semantic segmentation task for autonomous driving. We remapped RGB images to LiDAR polar-grid mapping representation and constructed early and mid-level fusion architectures. We provided experimental results on KITTI dataset and improved two state-of-the-art algorithms SqueezeSeg and PointSeg by 10% in both cases. In future work, we plan to explore more sophisticated fusion architectures using network architecture search techniques and utilize all the available color pixels.

REFERENCES

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[2] A. Briot, P. Viswanath, and S. Yogamani. Analysis of efficient cnn design techniques for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 663–672, 2018.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. Ieee, 2009.

[4] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel. On the segmentation of 3d lidar point clouds. In *2011 IEEE International Conference on Robotics and Automation*, pages 2798–2805. IEEE, 2011.

[5] K. Elmadawy, A. Al Sallab, M. Gamal, M. Abdelrazek, H. Eraqi, J. Honer, A. Valeo, and E. C. Cairo. Deep convolution long-short term memory network for lidar semantic segmentation.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2013.

[7] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[8] D. Grangier, L. Bottou, and R. Collobert. Deep convolutional networks for scene parsing. In *ICML 2009 Deep Learning Workshop*, volume 3, page 109. Citeseer, 2009.

[9] M. Heimberger, J. Horgan, C. Hughes, J. McDonald, and S. Yogamani. Computer vision in automated parking systems: Design, implementation and challenges. *Image and Vision Computing*, 68:88–101, 2017.

[10] M. Himmelsbach, A. Mueller, T. Lüttel, and H.-J. Wünsche. Lidar-based 3d object perception. In *Proceedings of 1st international workshop on cognition for technical systems*, volume 1, 2008.

[11] 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. *arXiv preprint arXiv:1602.07360*, 2016.

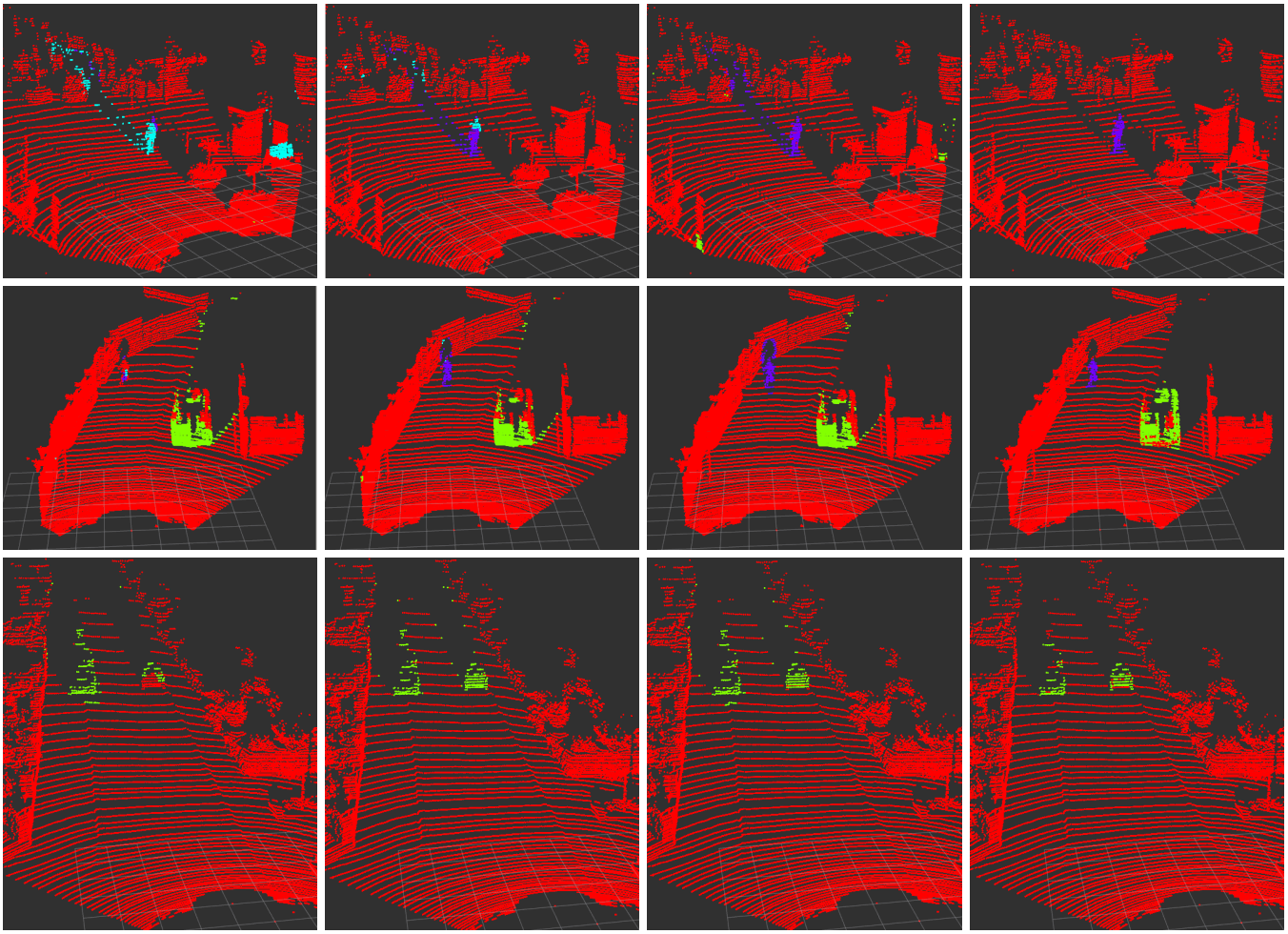


Fig. 5: Qualitative comparison of 3D semantic segmentation outputs using our approach on SqueezeSeg architecture. **Left:** No-Fusion output. **Middle-Left:** Early-Fusion output. **Middle-Right:** Mid-Fusion output. **Right:** Ground Truth.

[12] V. R. Kumar, S. Milz, C. Witt, M. Simon, K. Amende, J. Petzold, S. Yogamani, and T. Pech. Monocular fisheye camera depth estimation using sparse lidar supervision. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2853–2858. IEEE, 2018.

[13] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[14] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[15] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[16] H. Rashed, S. Yogamani, A. El-Sallab, P. Krizek, and M. El-Helw. Optical flow augmented semantic segmentation networks for automated driving. *arXiv preprint arXiv:1901.07355*, 2019.

[17] M. Siam, S. Elkerdawy, M. Jagersand, and S. Yogamani. Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8. IEEE, 2017.

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

[19] 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 *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893. IEEE, 2018.

[20] D. Zermas, I. Izzat, and N. Papanikolopoulos. Fast segmentation of 3d point clouds: A paradigm on lidar data for autonomous vehicle applications. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5067–5073. IEEE, 2017.

[21] Y. Zhou and O. Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.

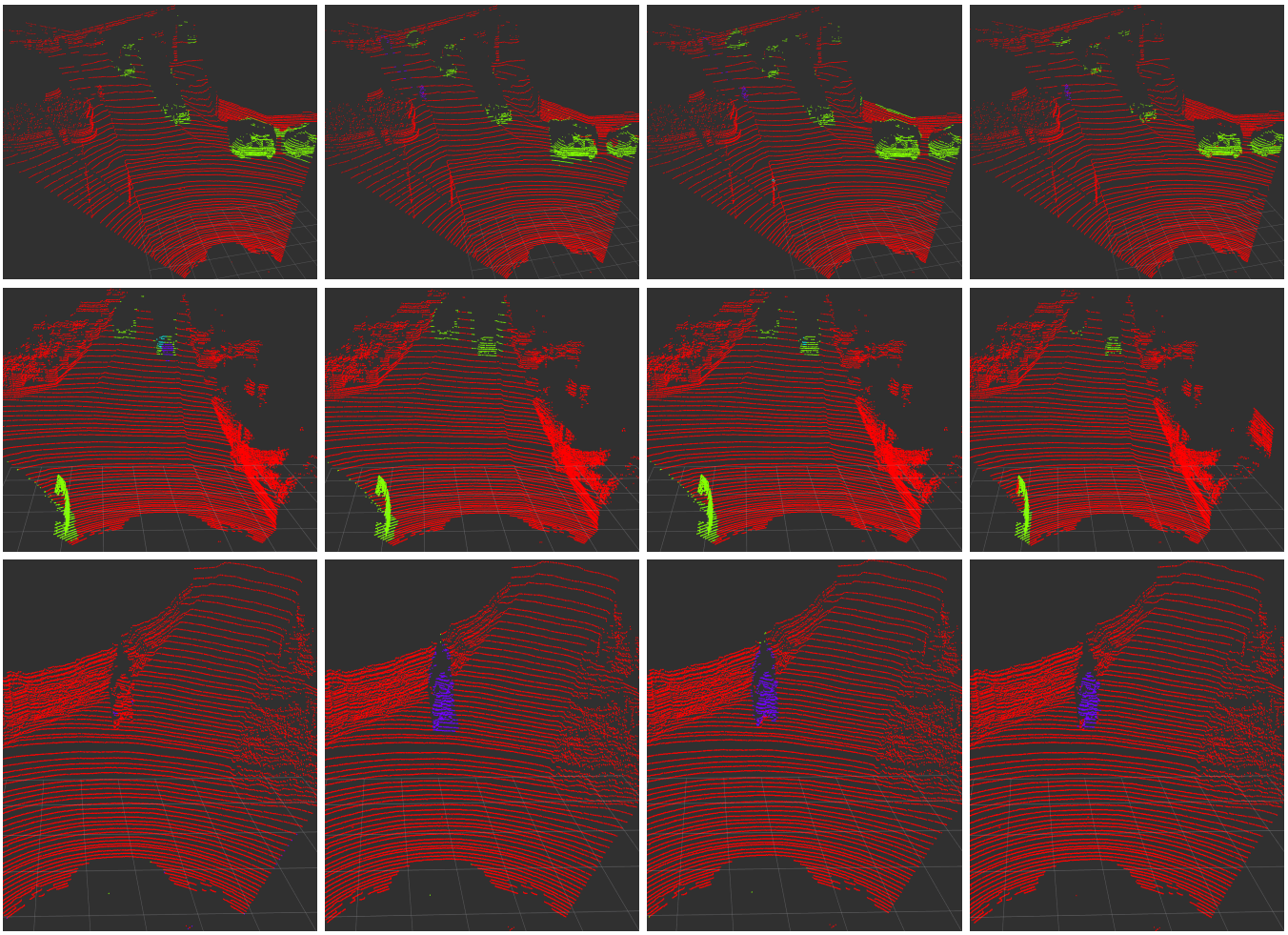


Fig. 6: Qualitative comparison of 3D semantic segmentation outputs using our approach on PointSeg architecture. **Left:** No-Fusion output. **Middle-Left:** Early-Fusion output. **Middle-Right:** Mid-Fusion output. **Right:** Ground Truth.