Deep Lidar-Camera Fusion for Road Detection

Abhinav Grover
<abhinav.grover@mail.utoronto.ca>
University of Toronto Institute for Aerospace Studies

I. Introduction

Autonomous driving has been a long-standing goal for humans in the 21th century. There are hundreds of companies, both large corporations as well as startups, that are working tirelessly to solve this highly complex problem. To solve such a problem, perceiving the environment of the autonomous agent is a fundamental task. Given the recent success of deep learning in solving perception tasks, the above companies are investing large number of resources in collecting enormous driving datasets, which are ubiquitous to the deep learning process. Moreover, many of these companies have now publicly released their labelled datasets, and setup challenges for the research community in order to accelerate development. Similarly, perception benchmarks setup by various institutions have also accelerated the development of new perception techniques, out of which the KITTI benchmarking suite [1] is the most widely accepted.

The KITTI benchmark is targeted towards autonomous driving, hence one of the benchmarking tasks performed on the KITTI dataset is the road detection/lane estimation problem. The goal of this report is to investigate the performance and transferability of a high-ranking road detection technique (ranked high on the KITTI benchmark) on a different driving dataset. This will help deduce the general usefulness of such high-ranking perception approaches and provide an estimate of the expected performance when adapting an approach. The contribution of this work is two-fold:

1. Adapt as well as evaluate an FCN-based technique to perform road-detection on the Audi A2D2 dataset [2].
2. Evaluate the transferability of a well-known road-detection technique from the KITTI dataset to the Audi A2D2 dataset.

II. Background and Motivation

A. Autonomous Driving

The complex task of Autonomous driving is generally divided into four subtasks: perception, localization, planning and vehicle control. In order to follow the traffic rules properly, it is important for an autonomous vehicle to perceive objects and road markings. These include visual cues such as lane markings, traffic flow symbols, road boundaries and objects like vehicles, cyclists, and pedestrians. Figure 1 gives a pictorial illustration of the perception needs. In order to perform these perception tasks, autonomous vehicles are equipped with a large suite of sensors; these include Lidars, Radars, Cameras, GPS (Global positioning system), IMU (initial measurement units), and even street maps. These sensors are able to provide data that is essential to understand the vehicle's environment, while onboard computers process the data in real-time.

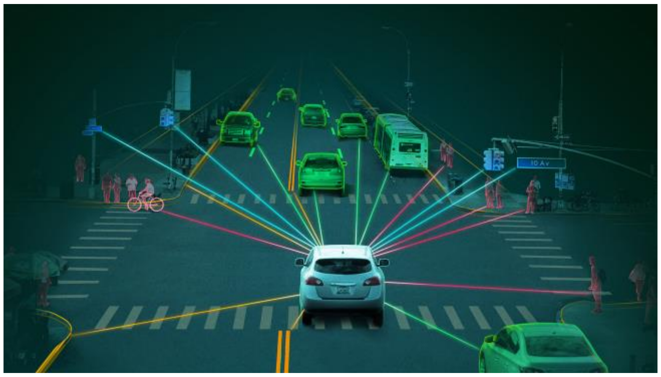

Figure 1: Image by Nvidia Inc. illustrating the perception requirements of an autonomous vehicle [4]

In some cases, detection of road signs and marking is performed with the aid of GPS and high definition navigation maps [5]. Yet the dependence of the autonomous pipeline on these non-native sensors prevents it from working in the driving environments such as country roads, tunnels etc. where either the GPS or the navigation maps are not reliable. This reinforces the need to perform the necessary detections using native sensors such as cameras, lidars, and radars. This work concentrates on the subproblem of road detection with an emphasis on fusion of lidar and camera data to achieve the task.

B. Road Detection

In the last decade, vision-based road detection has seen abundant innovation, majorly due to the popularity of the widely accepted KITTI vision benchmarks [1]. The KITTI lane and road estimation benchmark evaluates both lane and road detection with an emphasis on per pixel classification. To evaluate a road detection algorithm, metrics such as MaxF, Average Precision, Recall, False Positive Rate, False Negative rate, etc. are used [3]. This report will use these metrics as well in order to evaluate the road detection methods illustrated in this report.

If we comb through the road detection techniques benchmarked using the KITTI suite, there will be an overwhelming trend of using images to perform the task. Compared to other sensors like lidars and radars, camera data is much highly dense and feature rich. As a result, it is the most widely used sensor for the task of road detection. Yet, when it comes to 3D scene understanding, the camera data lacks essential pixel depth information. This information is usually derived from stereo images by performing feature matching disparity calculations, which is considered to be an expensive computational step. As a replacement to stereo

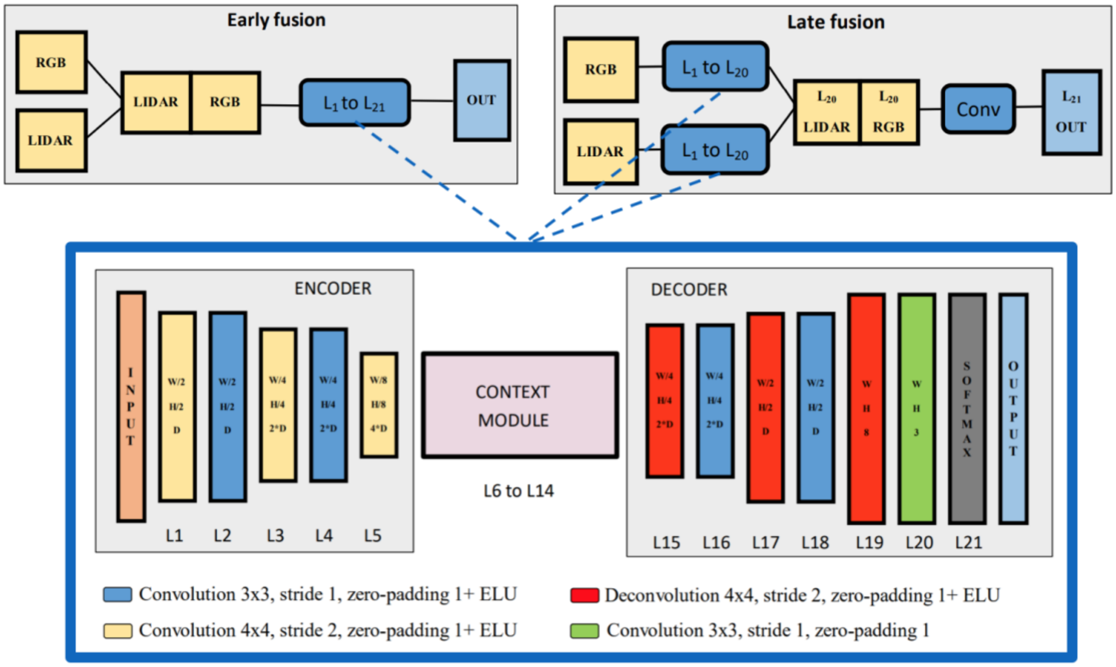

Figure 2: The proposed FCN architectures for early and late fusion.

images, lidar data provides accurate 3D information for each point in its point cloud, enabling the users to exploit the 3D knowledge of the scene. In order to use the dense camera image information and the high dimensional Lidar information, there has been a recent resurgence of data fusion techniques for vision based tasks [6][7][8]. These fusion approaches for vision tasks like object detection or scene understanding, have superseded the performance of purely vision-based approaches, which implies that fusing the lidar data with the images leads to a gain in information. In the context of road detection, this claim is evident from the top position of lidar-camera fusion based approaches on the KITTI benchmark. We would like to validate this trend for the A2D2 dataset as well.

C. Data Fusion using FCN

The work by Caltagirone et al. [6] serves as a foundation for the lidar-camera fusion based road detection techniques mentioned in this report. In their paper, the authors propose a technique that uses fully convolutional neural networks (FCN) to fuse the lidar and camera data, and perform per pixel semantic classification. This technique involves two modular steps: lidar data projection along with data up-sampling, followed by a road segmentation FCN. The method performs extremely well on the KITTI road detection benchmark and is the second-best publication in the rankings.

1) Lidar data processing

The goal of processing the lidar data, before inputting it into the FCN, is to transform it into an FCN compatible format. First, the lidar points are projected into the coordinates space of the camera using the sensor calibration information. Next, the lidar points clouds are concatenated to create lidar images that correspond to the field of view of the camera images. Since the lidar data is much more sparse compared to the camera images, an image up-sampling approach is employed [9]. The final result is three up-sampled lidar images corresponding to the X, Y and Z coordinate of each pixel feature.

2) Segmentation Backbone Architecture

FCNs are widely used to perform semantic image segmentation [10]. Per-pixel image segmentation using an FCN usually involves two overall steps: feature extraction using an encoder and context module, followed by an up-sampling or deconvolution segment to get back the original image size. The paper describes a basic encoder decoder FCN structure with a context module in the middle, as shown in Figure 2. The encoder creates a latent space representation of the image while the context module performs majority of the deep feature extraction. The decoder module is responsible for decoding the extracted features as well as performing per pixel classification. The specifics of this architecture have been shown in Figure 2. The exact structure of the context module can be found in the original paper [6]. The work in this report utilizes the same FCN architecture as the backbone of the semantic segmentation pipeline.

3) Fusion using FCN

In order to fuse the lidar and the image data for a given time step, Caltagirone et al. utilize the lidar XYZ images similar to RGB camera images in an FCN framework. The authors introduce three fusion strategies: early fusion, late fusion and cross fusion. But we will only talk about early and late fusion here. Early fusion implies concatenating the RGB camera images and the XYZ lidar images as the inputs to the base FCN, while late fusion implies using separate encoders and decoders for the two data sources and using them as combined input for the softmax layer. Figure 2 illustrates the idea behind these fusion strategies.

D. Datasets

In the current age there are numerous open source driving datasets available for research and development. The most popular one is the KITTI benchmarking dataset, because of its high acceptance across the research community, as well as its label versatility. Surprisingly, many of the well-known datasets that are available have been released by commercial entities. Waymo, a subsidiary of Alphabet inc., has released a large high-quality dataset targeted towards the task of 3D object detection & tracking [11]. Similarly, Lyft [12] and Nuscenes [13] also released driving datasets with a whole list of accompanying competitions and challenges. The automotive industry has also caught up, with Audi releasing their driving dataset “A2D2” [2] featuring German roads. This is one of the only datasets with semantic pixel-level labelling for each image, much like the KITTI dataset. BDD100k by the BAIR lab in Berkeley is a large community backed dataset with highly versatile labels and abundant datapoints [14], yet it is scares on per pixel semantic labels.

III. Methodology

The work in this report is highly based on the work of Caltagirone et al. [6] and can a considered as an adaptation of their technique. As mentioned before, this report outlines the methodology of adopting their work towards a different dataset and evaluating the performance.

A. Dataset of choice

For this work, the minimum pre-requisites for a dataset were the availability of front camera images segmentation labels and the corresponding lidar data with calibration information. Many of the above-mentioned dataset fulfilled these pre-requisites. A promising one was the widely used Nuscenes dataset which provided a large number of 20 second long driving clips consisting of camera and lidar data. These clips were also provided with a corresponding bird’s eye view (BEV) semantic map and each GPS waypoint, which made it possible to superimpose the semantic mask onto the camera images, as shown in Figure 3. Unfortunately, after visual inspection of overlaid semantic images, it was concluded that the average accuracy of the semantic mask was not enough for my purpose. The Lyft dataset was also rejected due to similar reasons, along with the idea to project semantic maps onto the camera images for ground-truth labels.

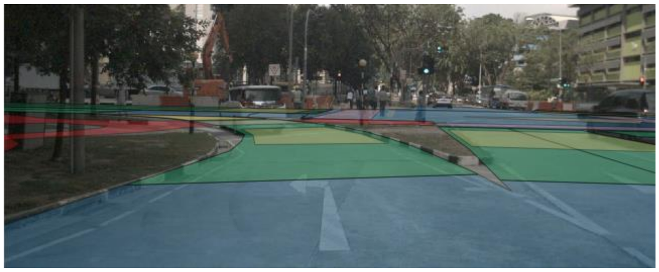

Figure 3: Image from the rejected Nuscenes driving dataset with semantic map overlaid on the image.

The A2D2 Audi dataset was the most promising one, due to its highly accurate semantic labelling and the availability of lidar data. Moreover, the lidar data provided was already cleaned and transformed onto the camera image space, as well as concatenated according to the field of view of the image. Figure 4Figure 4 shows an example of camera images with lidar data overlaid onto it. The figure also shows the corresponding semantic map containing 38 different semantic classes.

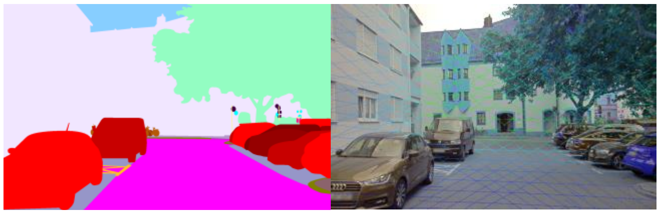

Figure 4: Semantic label (left) and the camera image (right) with the overlaid lidar point-cloud from the A2D2 dataset.

B. Lidar Data Pre-processing

In [6], the authors use a lidar up-sampling technique to create dense lidar images from the spare point cloud [9]. This technique, when applied to the a2d2 dataset, did not produce good results. Figure 5 shows the up-sampling result when the dense images are created using the above-mentioned technique. Visual inspection of these images clearly shows unwanted patterns and artifacts like diamond-shaped patterns which have the potential to serve as invalid features for the neural network.

In order to produce a dense lidar image, void of unwanted patterns and artifacts, a lidar depth completion algorithm called “IP basic” by Ku et. al. [15] is employed that produces much better dense lidar images. Since the a2d2 lidar point cloud is spare and moreover the IP basic algorithm is tuned for the dense lidar data of the KITTI dataset, the depth completion hyperparameters were slightly modified in order to make the depth completion work appropriately, for example the dilation kernel was increased in size. Figure 5 shows the result of applying the IP basic depth completion on the A2D2 lidar data.

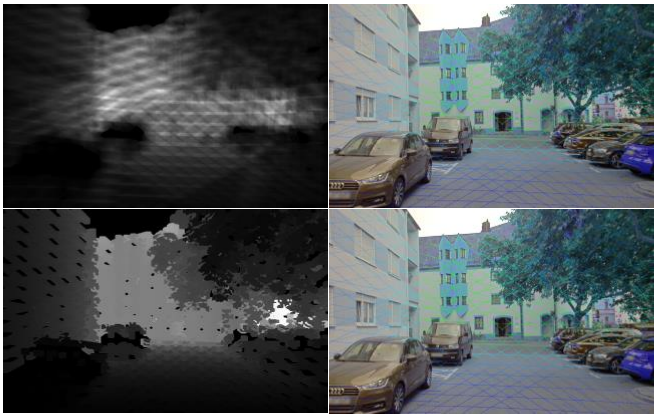

Figure 5: Up-sampled lidar images (left) along with its lidar-camera images (right) from the A2D2 dataset. The top-left up-sampled image was generated using the algorithm illustrated in [9] while the bottom left was generated using IP basic.

There is a significant difference in terms of computation time for the two lidar up-sampling algorithms. IP basic takes an average of 102 milliseconds per image on a 12-core intel i7 CPU, whereas the former algorithm takes over 1.04 seconds per image on the same compute configuration. This is due to the fact that IP basic performs linear kernel functions using the highly optimized OpenCV library, whereas the other method performs pixelwise non-linear operations and has not been optimized in any way. The reported times include the time it takes to save the up-sampled images, which implies that the compute-time ratio is expected to be higher than 10:1.

C. FCN and Fusion

For this work, The FCN architecture as well as the fusion techniques are similar to the work in [6], as described in section II.C, but with slight differences in the layer activations and the loss functions. Since we are trying to achieve binary classification, the activation of the last layer is modified to be logistic regression while the loss function is a linear combination of binary cross entropy (BCE) loss [16] and IOU loss [17]. The relative weight of the two losses is a hyperparameter which was kept at 0.5.

It is worth noting that the images in the A2D2 dataset have a resolution of 1920x1280 and were resized to a resolution of 480x320 for faster training on the limited GPU memory of Nvidia RTX 2070 Max-Q.

IV. IMPLEMENTATION

The paper by Caltagirone et al. [6] did not have any accompanying code, so I used an alternate source of an implementation of the described FCN architecture, also called ‘lidcamnet’[1]. Additional implementation details such as hyperparameter values are available in my code[2].

V. *RESULTS AND DISCUSSION*

Three types of FCN structures were trained on the A2D2 dataset: late fusion, early fusion, and no fusion (no lidar images). The networks were all trained for 100 epochs with identical training images. The lidar training images were creates using the IP basic method mentioned in section II.C.1), and both the lidar and the camera images were resized to a resolution of 480x320. For network training, the ADAM optimizer [18] and a polynomial learning rate were used along with batch normalization and 25% dropout. The training images consisted of 2800 image sets (a set consists of 1 RGB camera image and XYZ lidar images), and were limited to front facing camera images. Figure 6 compares the IOU metric for different fusion strategies at each epoch. Early fusion and fusionless FCN have a very close curve after 20 epochs, though the fusionless method has a slightly better final IOU value. Late fusion on the other hand performs very poorly with high fluctuations throughout the learning process.

As a final comparison, the 3 trained FCNs are compared using the same metrics as the KITTI benchmark in Table 1. As expected, the architecture devoid of fusion performs the best, while the early fusion is a close second. Late fusion, on the other hand, performs poorly, which is contrary to the results in source paper [6].

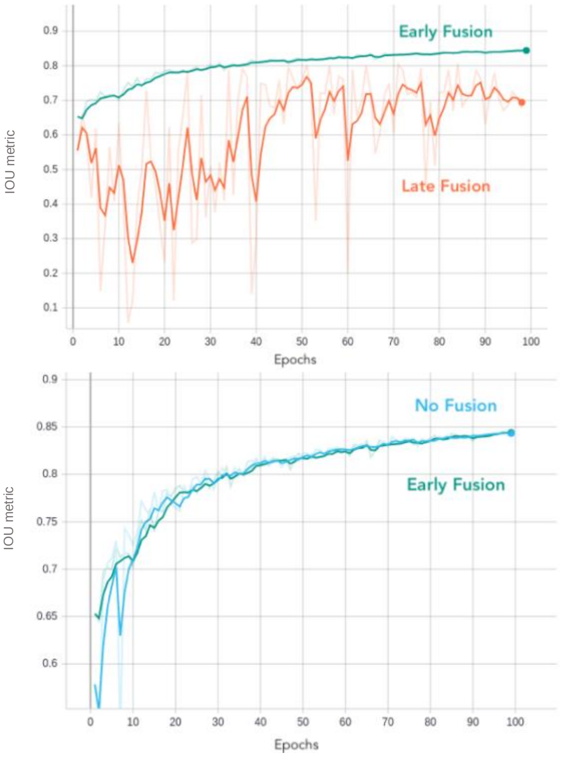

Figure 6: Intersection-over-union loss vs epoch plots for late vs early (top) and early vs fusionless (bottom) approaches.

The output masks generated by the three FCNs for a single test image are compared in Figure 7. Late fusion FCN appears to be learning the correct behavior but creates a highly conservative mask. The poor performance of the late fusion case can be attributed to the lack of hyperparameter tuning for both the lidar pre-processing and the two FCN branches, where the hyperparameters for both branches were kept identical due to time constraints on the project.

Table 1: A metric-based comparison of late, early, and no fusion FCN, trained for 100 epochs and tested on the same image

Fusion Method	MAXF	AVG. PREC.	PREC.	RECALL	FPR	FNR
Late	76.2%	72.5%	76.8%	75.7%	4.63%	24.28%
Early	87.2%	89.3%	86.9%	87.4%	2.67%	12.57%
None	**88.0%**	**89.8%**	**87.5%**	**88.5%**	**2.56%**	**11.46%**

The images of early and fusionless technique are very similar, with the former being slightly more conservative compared to the later. Another shocking revelation is the adversarial effect of having the lidar XYZ images along with the camera images. The naïve conclusion would be to deem the lidar data in the A2D2 dataset to be useless for road segmentation, but it is possible that independent hyperparameter tuning for each case could output different comparative results.

[1] Code by r3krut: https://github.com/r3krut/KITTI_ROAD_SEGMENTATION

[2] Link: https://github.com/abhitoronto/a2d2_road_segmentation

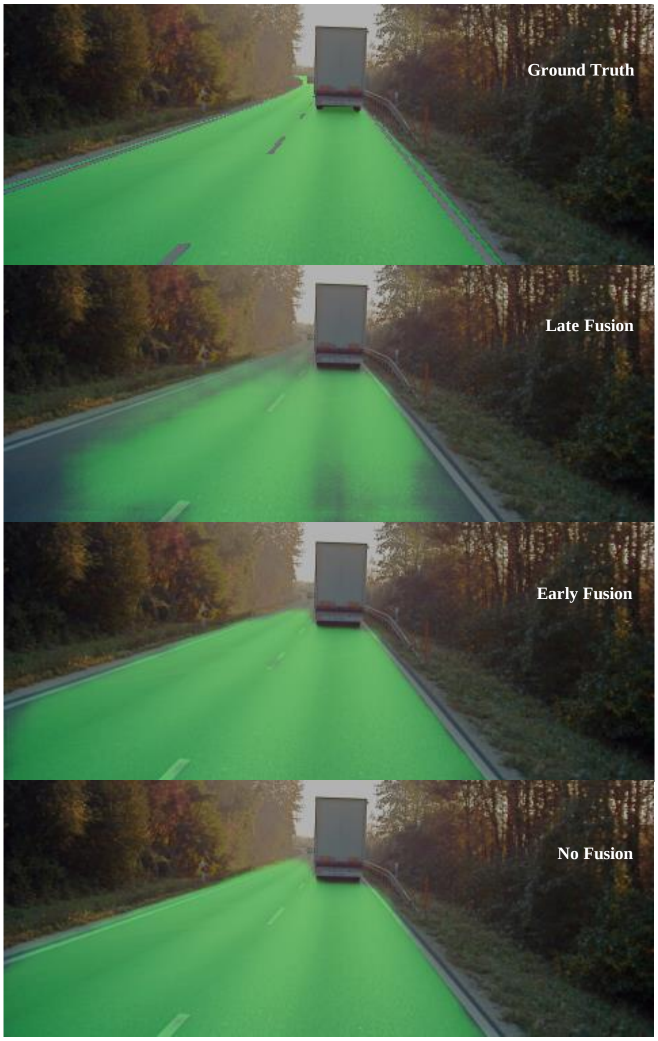

Figure 7: Resultant test images of late, early, and no fusion FCN structures, with the overlaid output pixel mask.

Finally, the transferability of the fusion approach by Caltagirone et al. is evaluated. Though it is unclear whether the fusion process is beneficial when compared to a purely camera-based approach, the road-segmentation performance of the FCN is considerable, given the fact that it lacks hyper-parameter tuning or any general performance enhancent. Table 2 compares the performance of the original fusion FCN trained and evaluated on the KITTI dataset with the performance of the above-mentioned FCN on the A2D2 dataset. This comparison is not a fair one yet it emphasizes the abnormally higher false negative rate (FNR) of the later network. This implies that the network is possibly on the conservative side of the spectrum.

Table 2: Performance comparison of the original fusion FCN (on KITTI) and the FCN in this report (on A2D2).

	MAXF	AVG. PREC.	PREC.	RECALL	FPR	FNR
KITTI	**95.6%**	**93.5%**	**95.7%**	**95.4%**	**1.92%**	**4.52%**
A2D2	88.0%	89.8%	87.5%	88.5%	2.56%	11.46%

VI. Conclusion and Future Work

This report outlines the procedure to adapt a well-known road segmentation approach, that ranks high on the KITTI benchmark tests, for Audi's driving dataset called the A2D2 dataset. The process includes a lidar data processing step followed by a Fully Convolutional Network (FCN) that performs fusion on lidar point clound and camera images. Due to the sparsity of the lidar data in the A2D2 dataset, a lidar processing procedure that is different from the original lidar camera fusion paper is used. With slight modifications to the backbone FCN, early, late, and fusionless networks are trained and evaluated. With minimal hyperparameter tuning, the late fusion network performs poorly while early and fusionless networks have a very similar performance. It is unclear whether the lidar data fusion can provide a performance boost, but the current results indicate otherwise. The base FCN with or without the lidar data clearly converges well on the A2D2 dataset, although more work is necessary to make a conclusive claim on whether a fusion approach works better or worse.

Acknowledgment

I would like to thank Qiyang Li, from the AUToronto student team, for his help with the literature review as well as his consistent feedback on the project. I would also like to thank Dr. Steven Waslander for guiding me towards his "IP basic" publication as a potential lidar data up-sampling algorithm.

References

[1] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite."

[2] "Driving Dataset > Audi Electronics Venture." [Online]. Available: https://www.audi-electronics-venture.de/aev/web/en/driving-dataset.html. [Accessed: 15-Dec-2019].

[3] J. Fritsch, T. Kuhnl, and A. Geiger, "A new performance measure and evaluation benchmark for road detection algorithms," in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2013, pp. 1693–1700.

[4] "Startup Aims to Make Autonomous Driving Safer | NVIDIA Blog." [Online]. Available: https://blogs.nvidia.com/blog/2017/11/23/safer-autonomous-driving/. [Accessed: 15-Dec-2019].

[5] R. P. D. Vivacqua, M. Bertozzi, P. Cerri, F. N. Martins, and R. F. Vassallo, "Self-Localization Based on Visual Lane Marking Maps: An Accurate Low-Cost Approach for Autonomous Driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 2, pp. 582–597, Feb. 2018.

[6] L. Caltagirone, M. Bellone, L. Svensson, and M. Wahde, "LIDAR-Camera Fusion for Road Detection Using Fully Convolutional Neural Networks."

[7] Z. Chen, J. Zhang, and D. Tao, “Progressive LiDAR adaptation for road detection,” *IEEE/CAA J. Autom. Sin.*, vol. 6, no. 3, pp. 693–702, May 2019.

[8] D. Xu, D. Anguelov, and A. Jain, “PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation,” Nov. 2017.

[9] C. Premebida, J. Carreira, J. Batista, and U. Nunes, “Pedestrian detection combining RGB and dense LIDAR data,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, no. Iros, pp. 4112–4117.

[10] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation.”

[11] “Data – Waymo.” [Online]. Available: https://waymo.com/open/data/. [Accessed: 15-Dec-2019].

[12] “Dataset | Lyft Level 5.” [Online]. Available: https://level5.lyft.com/dataset/#download. [Accessed: 15-Dec-2019].

[13] H. Caesar *et al.*, “nuScenes: A multimodal dataset for autonomous driving,” Mar. 2019.

[14] F. Yu *et al.*, “BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling,” May 2018.

[15] J. Ku, A. Harakeh, and S. L. Waslander, “In Defense of Classical Image Processing: Fast Depth Completion on the CPU,” Jan. 2018.

[16] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. 2016.

[17] P. Jaccard and E. Zurich, “Article in Bulletin de la Societe Vaudoise des Sciences Naturelles,” *Bull. la Société Vaudoise des Sci. Nat.*, vol. 37, pp. 547–579, 1901.

[18] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.