

Real-Time Road Segmentation Using LiDAR Data Processing on an FPGA

Yecheng Lyu, Lin Bai, and Xinming Huang
Department of Electrical and Computer Engineering
Worcester Polytechnic Institute
Worcester, MA 01609, USA
{ylyu,lbai2,xhuang}@wpi.edu

Abstract—This paper presents the FPGA design of a convolutional neural network (CNN) based road segmentation algorithm for real-time processing of LiDAR data. For autonomous vehicles, it is important to perform road segmentation and obstacle detection such that the drivable region can be identified for path planning. Traditional road segmentation algorithms are mainly based on image data from cameras, which is subjected to the light condition as well as the quality of lane markings. LiDAR sensor can obtain the precise 3D geometry information of the vehicle surroundings. However, it is a computational challenge to process a large amount of LiDAR data at real-time. In this work, a convolutional neural network model is proposed and trained to perform semantic segmentation using the LiDAR sensor data. Furthermore, an efficient hardware design is implemented on the FPGA that can process each LiDAR scan in 16.9 ms, which is much faster than the previous works. Evaluated using KITTI road benchmarks, the proposed solution achieves high accuracy of road segmentation.

Index Terms—Autonomous vehicle, road segmentation, CNN, LiDAR, FPGA

I. INTRODUCTION

In recent years, we have witnessed a strong increase of research interests on advanced driver assistance systems (ADAS) and autonomous vehicles. While fully autonomous driving might still be years away, there are many recent research on traffic scene perception and its implementations on various platforms. The traffic scene perception task can be separated into two sub-tasks: object detection and road/lane detection. Object detection includes vehicle detection [1] [2] [3] [4] [5], pedestrian detection [6] [3] [4] and traffic light/sign detection [7] [8] [9] [10] [11], while road/lane detection includes road marking detection [12] [13] [14], lane detection [15] [16] [17] and road segmentation [18] [19] [16]. In this work, we are primarily concentrating on road segmentation, since it is a fundamental component of automated driving that provides the drivable region for the vehicle's next movement.

Many sensing modalities have been used for road segmentation. Monocular vision [16] [5] [20] and stereo vision [21] [22] are often employed because cameras are low-cost and have a similar view to human eyes. However, considering road appearance diversity, image clarity issues, poor visibility conditions [23], image-based feature describers are often difficult to generate and easy to fail. In contrast of passive sensors such as cameras, light detection and ranging (LiDAR)

actively emits laser beams and measures the distance from the reflection by time of flight (TOF). Therefore, LiDAR is robust to environmental illumination. Several recent works have studied road segmentation based on LiDAR information or the combination of LiDAR and camera data [24], [25].

For the applications of autonomous vehicles, both real-time performance and power consumption need to be considered [26]. Graphic Processing Unit (GPU) is a popular platform for parallel processing, but power consumption is usually high. FPGA suits to the condition with limited power supply, such as an autonomous vehicle. Moreover, FPGA can be developed as a customized integrated circuit that can perform massive parallel processing and data communications on-chip. Hereby, we propose to target the LiDAR based road segmentation algorithm on an FPGA as a real-time low-power embedded system.

In this paper, the problem of road segmentation is framed as a semantic segmentation task in spherical image using a deep neural network. Instead of an encoder-decoder structure often implemented in traditional neural networks, a block containing a convolutional layer and a non-linear layer is cascaded twelve times so that multiplexing can be applied on the processing blocks on-chip. The proposed solution is evaluated on KITTI benchmarks and achieve satisfactory result. The rest of paper is organized as follows. Section II introduces the related work of road perception problem. The proposed convolutional neural network (CNN) structure and its performance on KITTI benchmarks are presented in Section III. Section IV presents the FPGA design hardware architecture and implementation results. Finally Section V concludes the paper.

II. RELATED WORK

Road segmentation has been studied with different sensors and algorithms over the past decade. In the early years, researchers used manually designed feature descriptors to separate the road from others. At that time, camera was the main sensor and features were often generated based on the illumination and shapes from images [14] [16] [27], which led to low accuracy and the performance variations from different light conditions and road scenes.

Recently, two major techniques have been investigated to overcome the shortcomings of manually selected features in

images. One is to use machine learning to design a complex and robust feature descriptor, such as CNN [28] [4] [20] and conditional random field (CRF) [29]. The other is to use intensity invariant sensor or multi-sensor fusion instead of camera to obtain a more robust descriptor. A popular intensity invariant sensor is LiDAR [30]. There were also research works trying to combine those two and apply machine learning to data processing. Several results showed high accuracy, but their processing time is too long to be employed for real-time applications [25] [24]. For autonomous driving, road segmentation must be implemented on real-time embedded platforms such as FPGA, application-specific integrated circuit (ASIC), or a mobile CPU/GPU processor. A neural network was proposed in [31] to detect lane markers on the road and had the run-time of 2.5 Hz on TK1 mobile GPU platform. Similarly, research work in [32] proposed a neural network to segment multiple objects including vehicle, pedestrian and pavement and achieved 10 Hz run time at the resolution of 480-by-320 pixels on TX1 GPU platform. In [15] and [33], FPGA based solutions are proposed for lane detection and resulted 60 Hz and 550 Hz processing speed, respectively.

III. ALGORITHMS DESIGN

The goal of road segmentation is to label the drivable region, also called free space. The input data comes from different sensors such as camera, LiDAR, GPS and IMU. The output are usually presented as area on the top-view or labeled pixels on camera view. In this paper, we choose LiDAR data as input, a deep neural network as the processor, and top-view predictions as the main output to evaluate the road segmentation performance. Results on camera view are also presented for better visualization. The proposed algorithm has the following three steps: pre-processing, neural network processing and post-processing.

A. Pre-processing

During pre-processing, input data points are arranged and projected into a 3-D blob with M by N tensors and C channels so that the tensor can flow through the layers in the neural network to produce an output. A tensor refers to a specific view in the real world. There are four types of views available for the autonomous driving task: image view (also known as camera view), top view (also known as bird eye view), cylindrical view and spherical view. Image view and top view are commonly choices, because in this two views LiDAR data can be fused with camera data and those views are natural to human eyes. However, LiDAR points are sparse in those views. Statistically, LiDAR points covers only 4% of pixels in image view and 5.6% on top-view. That means the majority inputs into the neural network are zeroes, which leads to a waste of computing resource. Cylindrical view and spherical view match the LiDAR sensing scheme and data points can cover up to 91% pixels on the map. Hereby we choose spherical view as the projection scheme. The resolution of polar angle θ and azimuthal angle φ are chosen based on the LiDAR resolution. In this work, all 64 rows are included in vertical. In

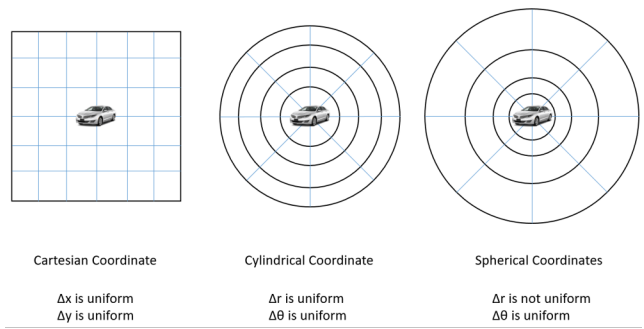


Figure 1: Grid projection to ground from different views

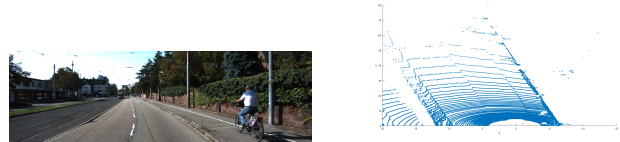


Figure 2: An illustration of camera view and the corresponding LiDAR points.

horizontal, LiDAR points are grouped by 0.4° , which doubles the designed resolution of LiDAR to minimum the number of cells without any LiDAR measurements. The input blob has 256 columns and FOV can be shifted to augment training data.

Although spherical view is chosen for data projection, we can still add additional feature channels from other views to improve the accuracy of the trained neural network. Here we select 16 channels. The first 7 channels come from the LiDAR point which has the lowest altitude in the cell, and the next 7 channels come from the LiDAR point which has the highest altitude. Each group of 7 channels includes location of measured points in Cartesian coordinate (x,y,z) , location of measured points in spherical coordinate (θ, φ, r) , and reflection intensity of measured points (H) . The remaining 2 channels are the location of cell on the 2D map (i, j) .

B. Neural network processing

In autonomous driving, traffic scene perception is often implemented on embedded systems. In consideration of limited computational resource in an embedded system, we propose a

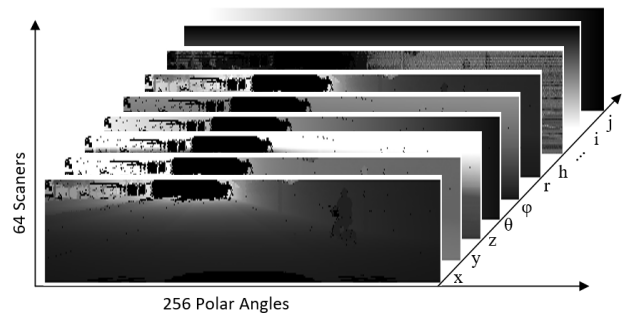


Figure 3: Input map to the neural network with 16 channels.

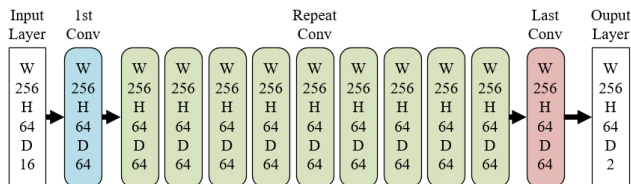


Figure 4: Architecture of the convolutional neural network for road segmentation.

new network architecture that minimize the GPU and FPGA memory by multiplexing the blob memory. The architecture is shown in Figure 4. Except for the first and last convolutional layers, those 9 layers in between are constructed using the same structure. Each repetitive structure includes a convolutional layer and an activation layer. The convolutional layer is built with 64 filters and each filter has a 5×5 kernel with stride size of 1 and padding size of 2. The stride and padding settings is chosen such that the output of the convolutional layer has the same size as the input. Rectified linear unit is chosen as the activation function for fast training. Two drop-out layers are added after 6th block and 10th block in training phase to accelerate convergence. It can be seen that there is no pooling layers and all blobs have the same size except for the input blob and score blob. Therefore all internal results can be stored in the same memory space directory without allocation or reshaping the blob. We choose 5×5 convolutional kernel size and 11 convolutional layers from our experimental results showing this settings is a good trade-off between CNN performance and resource usage.

C. Post-processing

In post-processing, results obtained from the neural network are projected back to target views, i.e. camera view and top view, for performance validation. The challenge of the post-processing is that the points in the output of neural network are non-uniformly distributed on the target view after projection. Traditional image processing methods, such as dilation, erosion, closing and opening, are not able to generate a filled area with smoothed contour. In our post-processing step, contour of the drivable area is firstly determined and then the region within the contour is marked as the segmentation results on target views. Figure 5 shows an example of the road segmentation results.

To determine the contour of the drivable area, the furthest points in each angle θ , which is corresponding to each column of the neural network output, are selected and projected onto the target view. Subsequently, a polyline is drawn along those furthest points on all angles on the target view. The polyline graph becomes a polygon if we add a straight line at the bottom. The polygon is then treated as the contour of drivable area and filled up with semantic pixel labels.

D. Training and evaluation on KITTI road benchmark

To evaluate the performance of the proposed approach, we train the network on KITTI road/lane detection dataset. As



Figure 5: Drivable area on camera view and top view projected from the neural network output

Table I: Comparison with existing results on KITTI road/lane detection dataset.

Name	F1-score	AP	run time
This work on FPGA	91.79%	84.76%	16.9ms
HybridCRF [25]	90.81%	84.79%	1500ms
LidarHisto [35]	90.67%	84.79%	100ms
MixedCRF	90.59%	84.24%	6000ms
FusedCRF [24]	88.25%	79.24%	2000ms
RES3D-Velo [36]	86.58%	78.34%	360ms

described in [34], F1-score and average precision (AP) are the key metrics to evaluate the performance of road perception algorithms. F1-score provides the insight of an algorithm's optimal performance, while AP indicates its average performance. The metrics are computed as Equation 1 to 6 in [34], where TP, TN, FP, FN are true positive, true negative, false positive, and false negative respectively. In Table I, we compare our proposed approach with several results published recently. It shows that our proposed approach has comparable performance but requires significantly less processing time. The actual processing time of the neural network implemented on the FPGA is about 16.9ms. Since most of the execution times listed in Table 1 were from various GPU platforms, we also evaluate our algorithm on a K20 GPU using MATLAB on Caffe and the total processing time is about 120ms, including pre-processing, neural network, post-processing, and visualization.

IV. HARDWARE ARCHITECTURE

As described in Section III, we organize the LiDAR data into an image map with 16 channels in the size of 256×64 . The block diagram of the convolutional layer architecture is shown in Figure 6. The same convolutional unit is used repetitively. There are 64 memories to store intermediate feature map and each memory size is 256k bits. The large 3D convolution can be broken into 64 parallel 2D convolutions, each with 2 filters, followed by an adder tree to generate the feature map.

A. Zero padding generation

Zero padding helps to control the size of feature maps and to reserve the boundary information of the input images in

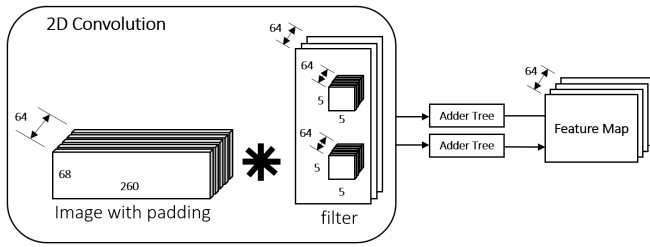


Figure 6: Hardware architecture of the implementation of convolution layer.

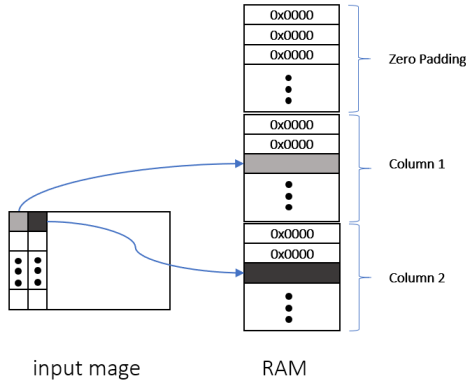


Figure 7: An illustration of the zero-padding in the RAM

convolution operation. Since the input images are transmitted without padding, a special dual-port RAM is designed for convolutions. As shown in Figure 7, each slot of RAM represents one column of the input image. The padded zeros are stored in the block RAM in advance. Control logic is used to store each pixel into proper memory location. On the other side of the memory, a scanning circuit reads out data from this RAM pixel by pixel.

B. 2D convolution

2D convolution is implemented in conjunction with a line buffer which consists of 4 lines and 5 additional registers. As demonstrated in Figure 8, it outputs 5×5 pixel window in parallel for the multiplication with the weight matrix using 25 multipliers. A highly pipelined adder tree follows the multiplication to compute the sum.

C. Control logic

Because of the large RAM consumption for images with zero padding and feature maps, a loop-based control is proposed. Each 2D convolution generates 2 feature maps. One finite state machine is used to generate 64 feature maps in 32 loops, reusing the block RAM for images with padding. To achieve 11 layers of convolution, another finite state machine is implemented for loop control. Therefore, completing the 11 fully convolutional layers with each depth of 64 requires to perform 352 times of the 2D convolutions.

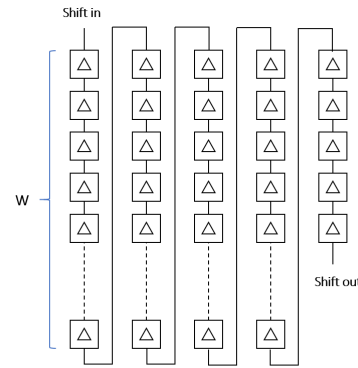


Figure 8: Line buffer for 2D convolution.

Table II: Resource usage of the neural network implementation on an FPGA

	Used	Available	Utilization
Slice Registers	43726	1326720	3.30%
Slice LUTs	18684	663360	2.82%
Block RAM Tile	1513	2688	70.05%
DSPs	4480	5520	81.16%

D. Implementation results

We implement this fully convolutional network on Xilinx UltraScale XCKU115 FPGA. The targeted operating frequency is set to 350MHz. Each 2D convolution takes about 18,000 clock cycles. It takes about 16.9ms to complete all 11 convolutional layers, each with filter depth of 64 (except for depth of 16 in the first layer and the depth of 2 in the last layer). Since LiDAR normally scans at 10Hz, this FPGA implementation fulfills the requirement of real-time processing. When tested on the Intel Xeon CPU E5-2687W v3, the processing time is about 500ms. Therefore, the FPGA implementation gains the speedup factor of 30 over CPU. The resource usage of the FPGA implementation is listed in Table II.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a neural network based approach for road segmentation using LiDAR data. The neural network is trained with KITTI road/lane detection dataset and evaluated on its test benchmark. Moreover, the proposed fully connected neural network is implemented on an FPGA for real-time low-power processing, which results the processing time of only 16.9ms for each LiDAR scan. The implementation consumes a large amount of FPGA on-chip memory. For future work, we are considering using the external DDR4 SDRAM to store feature maps. We also notice during testing that sidewalk and railway with same altitude as road pavement contributes to the majority of false positives. Fusion of LiDAR and camera data is needed to further improve the accuracy.

VI. ACKNOWLEDGEMENT

This work was supported in part by the U.S. National Science Foundation under Grant 1626236 and by the Mathworks.

REFERENCES

- [1] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," in *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*. IEEE, 2017, pp. 924–933.
- [2] E. Dagan, O. Mano, G. P. Stein, and A. Shashua, "Forward collision warning with a single camera," in *Intelligent Vehicles Symposium, 2004 IEEE*. IEEE, 2004, pp. 37–42.
- [3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.
- [4] A. González, G. Villalonga, J. Xu, D. Vázquez, J. Amores, and A. M. López, "Multiview random forest of local experts combining rgb and lidar data for pedestrian detection," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 356–361.
- [5] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "Multinet: Real-time joint semantic reasoning for autonomous driving," *arXiv preprint arXiv:1612.07695*, 2016.
- [6] D. M. Gavrilu and S. Munder, "Multi-cue pedestrian detection and tracking from a moving vehicle," *International journal of computer vision*, vol. 73, no. 1, pp. 41–59, 2007.
- [7] Z. Chen and X. Huang, "Accurate and reliable detection of traffic lights using multiclass learning and multiobject tracking," *IEEE Intelligent Transportation Systems Magazine*, vol. 8, no. 4, pp. 28–42, 2016.
- [8] Z. Chen, X. Huang, Z. Ni, and H. He, "A gpu-based real-time traffic sign detection and recognition system," in *Computational Intelligence in Vehicles and Transportation Systems (CIVTS), 2014 IEEE Symposium on*. IEEE, 2014, pp. 1–5.
- [9] J. Zhao, X. Huang, and Y. Massoud, "An efficient real-time fpga implementation for object detection," in *New Circuits and Systems Conference (NEWCAS), 2014 IEEE 12th International*. IEEE, 2014, pp. 313–316.
- [10] R. N. Rajaram, E. Ohn-Bar, and M. M. Trivedi, "Refinenet: Iterative refinement for accurate object localization," in *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*. IEEE, 2016, pp. 1528–1533.
- [11] A. De La Escalera, L. E. Moreno, M. A. Salichs, and J. M. Armingol, "Road traffic sign detection and classification," *IEEE transactions on industrial electronics*, vol. 44, no. 6, pp. 848–859, 1997.
- [12] T. Chen, Z. Chen, Q. Shi, and X. Huang, "Road marking detection and classification using machine learning algorithms," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 617–621.
- [13] T. Wu and A. Ranganathan, "A practical system for road marking detection and recognition," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*. IEEE, 2012, pp. 25–30.
- [14] M. Aly, "Real time detection of lane markers in urban streets," in *Intelligent Vehicles Symposium, 2008 IEEE*. IEEE, 2008, pp. 7–12.
- [15] J. Zhao, B. Xie, and X. Huang, "Real-time lane departure and front collision warning system on an fpga," in *High Performance Extreme Computing Conference (HPEC), 2014 IEEE*. IEEE, 2014, pp. 1–5.
- [16] M. Beyeler, F. Mirus, and A. Verl, "Vision-based robust road lane detection in urban environments," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4920–4925.
- [17] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017, pp. 1856–1860.
- [18] L. Caltagirone, S. Scheidegger, L. Svensson, and M. Wahde, "Fast lidar-based road detection using convolutional neural networks," *IEEE Intelligent Vehicles Symposium 2017*, 2017.
- [19] A. Laddha, M. K. Kocamaz, L. E. Navarro-Serment, and M. Hebert, "Map-supervised road detection," in *Intelligent Vehicles Symposium (IV), 2016 IEEE*. IEEE, 2016, pp. 118–123.
- [20] G. L. Oliveira, C. Bollen, W. Burgard, and T. Brox, "Efficient and robust deep networks for semantic segmentation," *The International Journal of Robotics Research*, p. 0278364917710542, 2017.
- [21] G. B. Vitor, A. C. Victorino, and J. V. Ferreira, "A probabilistic distribution approach for the classification of urban roads in complex environments," in *IEEE Workshop on International Conference on Robotics and Automation*, 2014.
- [22] N. Einecke and J. Eggert, "Block-matching stereo with relaxed fronto-parallel assumption," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*. IEEE, 2014, pp. 700–705.
- [23] A. B. Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: a survey," *Machine vision and applications*, vol. 25, no. 3, pp. 727–745, 2014.
- [24] L. Xiao, B. Dai, D. Liu, T. Hu, and T. Wu, "Crf based road detection with multi-sensor fusion," in *Intelligent Vehicles Symposium (IV), 2015 IEEE*. IEEE, 2015, pp. 192–198.
- [25] L. Xiao, R. Wang, B. Dai, Y. Fang, D. Liu, and T. Wu, "Hybrid conditional random field based camera-lidar fusion for road detection," *Information Sciences*, 2017.
- [26] R. Okuda, Y. Kajiwara, and K. Terashima, "A survey of technical trend of adas and autonomous driving," in *VLSI Technology, Systems and Application (VLSI-TSA), Proceedings of Technical Program-2014 International Symposium on*. IEEE, 2014, pp. 1–4.
- [27] H. Kong, J.-Y. Audibert, and J. Ponce, "General road detection from a single image," *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2211–2220, 2010.
- [28] C. C. T. Mendes, V. Frémont, and D. F. Wolf, "Exploiting fully convolutional neural networks for fast road detection," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3174–3179.
- [29] M. Passani, J. J. Yebe, and L. M. Bergasa, "Crf-based semantic labeling in miniaturized road scenes," in *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*. IEEE, 2014, pp. 1902–1903.
- [30] A. S. Huang, D. Moore, M. Antone, E. Olson, and S. Teller, "Finding multiple lanes in urban road networks with vision and lidar," *Autonomous Robots*, vol. 26, no. 2, pp. 103–122, 2009.
- [31] B. Huval, T. Wang, S. Tandon, J. Kiske, W. Song, J. Pazhayampallil, M. Andriluka, P. Rajpurkar, T. Migimatsu, R. Cheng-Yue *et al.*, "An empirical evaluation of deep learning on highway driving," *arXiv preprint arXiv:1504.01716*, 2015.
- [32] E. Romera, J. M. Alvarez, L. M. Bergasa, and R. Arroyo, "Efficient convnet for real-time semantic segmentation," in *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017, pp. 1789–1794.
- [33] W. Wang and X. Huang, "An fpga co-processor for adaptive lane departure warning system," in *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 1380–1383.
- [34] J. Fritsch, T. Kuehnl, and A. Geiger, "A new performance measure and evaluation benchmark for road detection algorithms," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.
- [35] L. Chen, J. Yang, and H. Kong, "Lidar-histogram for fast road and obstacle detection," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1343–1348.
- [36] P. Y. Shinzato, D. F. Wolf, and C. Stiller, "Road terrain detection: Avoiding common obstacle detection assumptions using sensor fusion," in *Intelligent Vehicles Symposium Proceedings, 2014 IEEE*. IEEE, 2014, pp. 687–692.