Deep Fully Convolutional Networks with Random Data Augmentation for Enhanced Generalization in Road Detection

Jesús Muñoz-Bulnes, Carlos Fernandez, Ignacio Parra, David Fernández-Llorca, Miguel A. Sotelo

Abstract—**In this paper, a Deep Learning system for accurate road detection is proposed using the ResNet-101 network with a fully convolutional architecture and multiple upscaling steps for image interpolation. It is demonstrated that significant generalization gains in the learning process are attained by randomly generating augmented training data using several geometric transformations and pixelwise changes, such as affine and perspective transformations, mirroring, image cropping, distortions, blur, noise, and color changes. In addition, this paper shows that the use of a 4-step upscaling strategy provides optimal learning results as compared to other similar techniques that perform data upscaling based on shallow layers with scarce representation of the scene data. The complete system is trained and tested on data from the KITTI benchmark and besides it is also tested on images recorded on the Campus of the University of Alcala (Spain). The improvement attained after performing data augmentation and conducting a number of training variants is really encouraging, showing the path to follow for enhanced learning generalization of road detection systems with a view to real deployment in self-driving cars.**

Index Terms—**CNN, Deep Learning, Road Detection, Random Data Augmentation, Multistep Up-sampling**

I. Introduction & Related Work

Autonomous vehicles require a precise and robust perception of the environment. It is a crucial point in the development of autonomous vehicles because the perception layer is the base for higher level systems, such as control algorithms or path planning. One of the main issues is the road detection. It has traditionally been an exhaustive topic of research in the fields of Advanced Driver Assistance Systems (ADAS) and autonomous driving. The advent of Deep Learning techniques, namely, Convolutional Neural Networks (CNNs) has signified a breakthrough in the field of Artificial Intelligence, with strong implications in a large variety of application domains. Thus, research on self-driving cars is experiencing a significant thrust due to the enhanced perception capabilities that the deployment of CNNs are making possible today. Powerful CNN models, such as AlexNet or ResNet, are endowing self-driving cars with advanced capabilities to robustly and accurately interpret road scenes, even in complex urban scenarios with a great deal of clutter.

It is well known that CNNs can achieve state-of-the-art results on image classification [1]–[4], and they have also been successfully applied to object detection [5], [6] as well as to monocular color image segmentation. There are several approaches to obtain a pixelwise classification of an input image. The widely-adopted fully convolutional networks (FCN) [7] adapt classifier networks, such as AlexNet [1] and VGG [2], to the segmentation task by replacing fully-connected layers with convolutional ones and using a progressive interpolation approach. Others, like [8], follow this trend using other base networks, such as the ResNet [3]. In [9] they introduce the use of dilated convolutions to reduce the downsampling performed by the net and remove the necessity of the progressive interpolation. That kind of dilated convolutions are further explored in [10] along with another upsampling method called dense upsampling convolution. A more complex approach such as DeconvNet [11] learns a deep deconvolutional network on top of the convolutional one. SegNet [12] uses an encoder-decoder architecture, PSPNet [13] exploits pyramidal pooling to introduce global contextual priors in a dilated fully convolutional network, and FRRN [14] presents a novel architecture that keeps a stream with full-resolution features. Finally, there are specialized methods for road detection. One example is [15], where the goal is to optimize the models to speed-up inference and make them capable of being used in a real road detection scenario. In [16], MultiNet system is presented, which performs simultaneous street classification, vehicle detection and road segmentation, all with the same CNN encoder and three different decoders.

II. System Description

Given the generalized use of CNNs also on the road detection problem, this paper develops and evaluates a road detector based on the ResNet network model [3] and the fully convolutional architecture [7]. Initially, a ResNet-50 model has been used. This model was already trained on the ImageNet dataset, which consists of 1000 labels at image level. In contrast, our detector is evaluated on the KITTI road detection dataset [17], which only defines 2 labels (road/non-road) at pixel level. It requires to transform the original ResNet-50 architecture into a fully convolutional network in order to admit an input of an arbitrary size and to produce an output of the same size with pixelwise classification. This is addressed by replacing the last inner-product fully-connected classifier layer (1000 outputs) with a new convolutional layer (two outputs) that will be learned from scratch. In addition, upsampling will be needed, since the ResNet network downsamples the input in some layers, resulting in an overall downsampling factor of 32. The

The authors are with the Computer Engineering Department, University of Alcalá, Madrid, e-mail: {carlos.fernandezl, ignacio.parra, david.fernandezl, miguel.sotelo}@uah.es e-mail: jesus.munozb@edu.uah.es

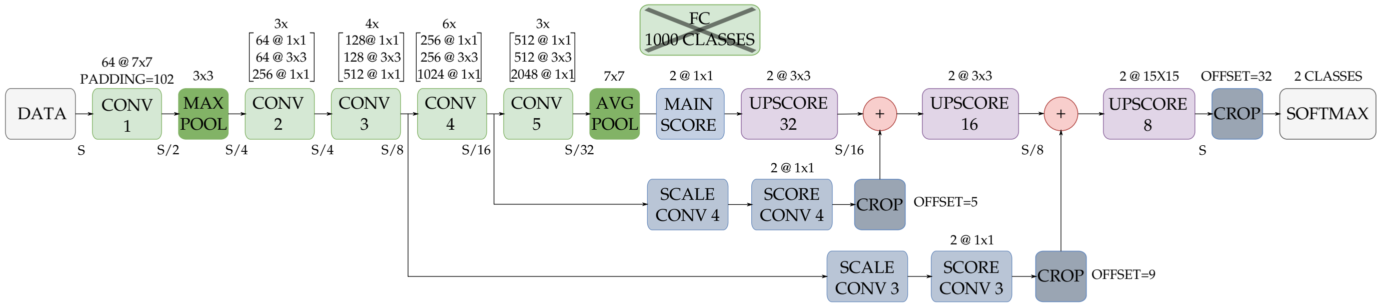


Fig. 1. High-level schematic of the CNN-based road detector, implementing the FCN-8s architecture [7] on a ResNet-50

final output consists of two channels that are the probability maps for background and road respectively, obtained from the final SOFTMAX layer.

As described in Figure 1, the upsampling is performed in three interpolation stages: the first stage (UPSCORE 32) upsamples the main output scores by a factor of two, and then the output from a previous block (CONV 4) is added, since both scores have the same accumulated downsampling factor (16). The result is upsampled again by a factor of two, an the output from another previous block (CONV 3, with a downsampling factor of 8) is added. Finally, the result is upsampled by a factor of eight to recover the scores in the original input size. This process allows to recover pixelwise scores smoothly, with a high level of detail: the final output combines the coarser global features (main score) with some finer local features (SCORE CONV 4 and SCORE CONV 3). Upsampling layers are initialized with bilinear interpolation kernels, that do not need to be trained. The scores from the shallower layers are obtained with a two-output convolutional layer in the same manner as the main score. Also, a learnable scaling layer is placed before each one to help the network to adapt the different features to their addition. A large padding is added on the first stage (CONV 1) to compensate for the width and length reduction that pooling layers and convolutions combined with downsamplings can cause. Then, some croppings have to be performed to align the score maps and match dimensions, with an offset which is calculated automatically during the architecture definition.

III. Experimental setup

After these transformations, the network is ready to be trained on the KITTI dataset, which is composed of 289 images manually labeled with two classes (road/non-road). 50% of the images are used for training the net, and the remaining 50% are kept aside for validation.

More specifically, the ResNet-50 model previously trained on ImageNet is used for weight initialization, and then the full net is fine-tuned on the road detection task. The training is run for 24K iterations, with validation checkpoints every 4K iterations.

The Caffe framework [18] has been used for the network prototype definition and the control of training and testing processes. The ImageNet per-channel pixel mean is subtracted, and the label images are converted into a 1 × height × width integer array of label indices to be compatible with the loss function. Instead of passing the original image to the network, some data augmentation operations are applied to extend the training set, prevent overfitting and make the net more robust to image changes. The data augmentation layer runs on CPU, and the rest of the processing can be done on GPU. It takes between two and three hours to complete the standard training on a single Titan X GPU.

A. *Data augmentation*

Data augmentation techniques can be very useful to extend the training set. An on-line augmentation approach is adopted. This way, the network never sees the same augmented image twice, as the modifications are performed at random each time. Besides, this virtually infinite dataset does not require extra storage space on disk. Moreover, data augmentation plays an important role in making the net more robust against usual changes that appear in road images, such as illumination, color or texture changes, or variations in the orientation of the cameras. One of the main weaknesses of CNNs is their dependence on the previous training data. With data augmentation a better generalization can be achieved and different road conditions can be simulated.

1) Geometric transformations: These transformations must be applied to both the image and the ground truth mask.

- Random affine transformations: Translations, rotations, scalings and shearings are performed in order to change the positions of three reference points, while keeping lines parallel. Although these transformations could be applied independently, better results are obtained with combined affine transformations due to the high variability.
- Mirroring.
- Cropping the image and scaling it to the original size: Crops are defined by a random top left corner and also random size, within image limits.
- Distortion: Random distortion parameters are applied to the image.
- Perspective transformations: The original positions of four reference points are selected empirically on usual road limits. Their final positions are calculated adding Gaussian noise to the original ones with two restrictions: the shift of top points is the opposite of that of the bottom

ones, and top points should not cross each other to prevent reflected images.

2) Pixel value changes: These transformations are only applied to the image, since they produce changes only on pixel values.

- Noise: Addition of Gaussian, speckle, salt & pepper noise, generation of an image with signal-dependent Poisson noise.
- Blur.
- Color changes: Three types of transformations are applied. The first one is casting, which consists in adding a random constant to each RGB channel, with the effect of altering the color components of the image [19]. The second one is an additive jitter, which is generated at random by means of exponentiation, multiplication and addition of random values to the saturation and value channels, or simply drawing a constant from a uniform distribution in the case of hue channel. This jitter is then added to the original HSV image. Uniform distribution limits have been tuned empirically for this dataset in order to keep those transformations realistic. The last one is a PCA-based shift, which is a method presented in [1] for performing slight alterations in RGB space. It is based on a previous PCA analysis of RGB values throughout the training subset. It consists in adding to each pixel a linear combination of the found three principal components (eigenvectors of the covariance matrix) with magnitudes proportional to their corresponding eigenvalue times a random gaussian variable (standard deviation of 0.01). This way, instead of changing RGB values independently, the shift is performed in the principal components' space.

B. Network components and training variants

Regarding fully convolutional networks, there are several elements that can be optimized, such as the initialization of the score layers (with zeros, noise, etc.), and the initialization and training of the upsampling layers. We can also use more complex activation functions rather than the simple ReLU, such as parametric ReLUs (PReLUs), which are recommended in combination with MSRA initialization [20]. Note that it is not possible to change the original ResNet-50 structure since we would lose the previous learning, but we can add PReLUs to the new score layers. Training alternatives involve trying different learning rates (lower or higher) and learning rate policies, such as decreasing the learning rate when the training stalls in previous trials, or doing a *warmup* stage [3] at a reduced learning rate until error goes under (20%). Other common suggestion is to have a higher learning rate for score layers, which are learned from scratch, and a lower rate for inherited layers. Moreover, in [7], several training schemes are defined: the standard accumulated learning (batch size of 20 and standard momentum of 0.9) or the *heavy learning* scheme, which uses a single image per gradient actualization and a high momentum of 0.99, that simulates the gradient accumulation effect of the batch size. In [10] they use a variant of the accumulated learning (batch size of 12) with a polynomially decreasing learning rate which we try in the form $2.5 \cdot 10^{-4} \times (1 - iter/max\_iter)^{0.9}$.

C. Training in Bird's Eye View

The traditional procedure of training starts using images in perspective view and obtaining detections in this space. However, since KITTI benchmark evaluates its results with the F1-measure in Bird's Eye View (BEV) [21], our proposal trains the model directly in BEV. In this case, a less aggressive data augmentation strategy is used since geometric transformations in BEV create important distortions.

D. Deeper models

A ResNet-101 [3] model has been adapted in the same manner as the ResNet-50, to test a deeper model in this problem. On the one hand, this model has an increased learning capacity but on the other hand, the risk of overfitting becomes more relevant.

E. Upsampling variants

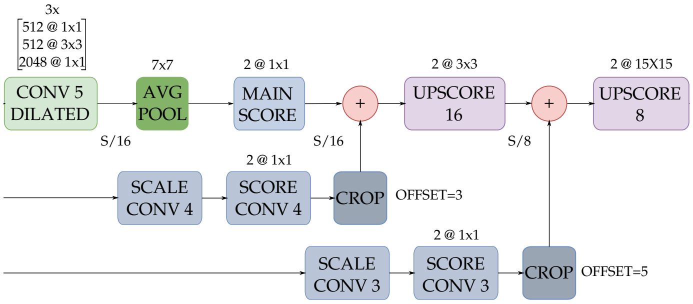
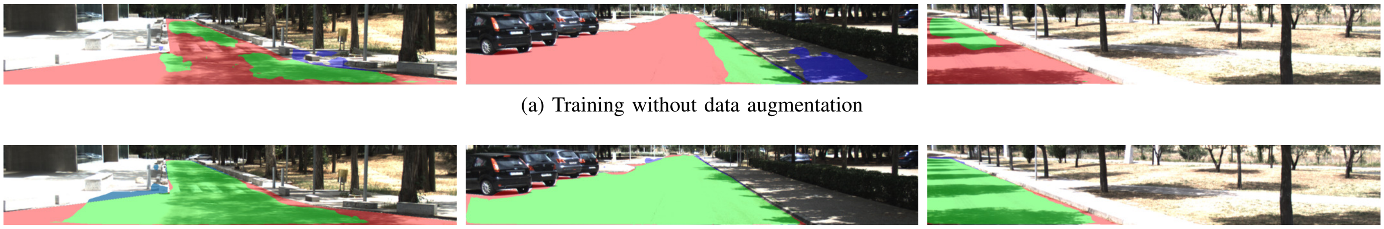


Fig. 2. Detail of the scoring stage with dilated convolution in CONV 5. It removes the necessity of using the UPSCORE 32 layer.

Apart from the schematics presented in Figure 1, the number of connections from shallower layers is modified. In order to obtain a more fine grained classification, both the full step-by-step upsampling (with additional connections from CONV 2 and CONV 1) and the four-step one (only additional connection from CONV 2) have been tested. Likewise, a two-step approach has also been evaluated to cover all possible cases, as well as the basic approach with no skip connections and just one large interpolation step.

There are other methods to increase the field of view of the deeper layers without downsampling the input features. The dilated convolution [9] and its improved version [10], which is claimed to avoid grid effects are evaluated. This approach replaces the downsampling performed in one or more blocks with dilated convolutions in all of the subsequent layers. However, downsampling not only is necessary to enlarge the field of view, it also plays an important role reducing the size of the input features to reduce the GPU memory consumption. If downsampling is completely removed, the model will not fit in memory. For this reason, our tested method combines a dilated convolution in the deeper block of the ResNet-50, with two upsampling steps to achieve a tradeoff, see Figure 2.



(a) Training without data augmentation

(b) Training with data augmentation

Fig. 3. The qualitative results of the proposed model are coloured as follows: TP in green, FP in blue and FN in red. The scenarios with strong illumination changes and challenging road textures are better detected in the model trained with data augmentation.

IV. RESULTS

In the following sections, the results obtained from the previously mentioned experiments are discussed. As proposed in [21], quantitative results are calculated in terms of F1-measure, computed over the validation subset on Bird's Eye View (converting from perspective view when the road detector is trained in this space). Namely, the "MaxF" is computed using the working point (confidence threshold) in the precision-recall curve that maximizes F1-measure.

A. Data augmentation

The use of data augmentation prevents the network overfitting, since the gap between training and validation losses disappears: training losses rise slightly whereas validation losses decrease. Transforming the full image with a single random operation each time, a higher variability is obtained. Furthermore, geometric transformations introduce higher variability than pixel value changes, obtaining better results when both are working together. The improvement is approximately 1% in F-measure training in perspective space (from 94.59% to 95.76%), and 2% training in Bird's Eye View. Moreover, the trained model was tested on some sequences at the campus of the University of Alcala, Madrid (Spain), to test the network in a different environment from that used in the training. Figure 3 demonstrates that data augmentation makes the model more robust against illumination, texture, perspective and orientation changes.

B. Network components and training variants

The upscore described in Figure 1 is composed of fixed bilinear kernels and score layers are initialized using the MSRA method because it is considered robust against symmetries in gradient propagation. PReLU activation functions are not used. Regarding the learning rate, three different rates are compared ($1 \cdot 10^{-6}$, $5 \cdot 10^{-5}$, $1 \cdot 10^{-4}$). The slower one ($1 \cdot 10^{-6}$) does not converge even with 40K iterations, the faster one ($1 \cdot 10^{-4}$) adds instability to the process and the best results are obtained with the trade off between both approaches ($5 \cdot 10^{-5}$), a fixed learning rate, same for the whole net, and *heavy learning*.

C. Training in Bird's Eye View

In general, the model is able to learn better (less training losses) and also to generalize better (smaller gap with validation losses) during the training in perspective view because perspective images have more information about the scene, and more aggressive data augmentation recipes can be applied while maintaining the meaning of the image. Thus, without data augmentation, the model trained in BEV (94.08%) is worse than the one in perspective view (94.59%).

Data augmentation can significantly reduce the gap between training and validation losses and makes it worthwhile to train in BEV. Although the BEV approach with data augmentation is still worse at learning than the perspective one, the fact of learning in the same space as the evaluation obtains a better performance (96.06%). Analysing in detail the performance, the model trained in BEV performs similarly at near and further pixels, whereas the perspective model has more problems with further pixels. Some problems of the BEV approach is that in some cases, buildings at the end of the road or incoming tunnels can be confused with a continuation of the road.

D. Deeper models

The tests over deeper models trained in perspective space establish that the ResNet-101 achieves slightly better results over ResNet-50, which are obtained consistently with fewer iterations. As a drawback, the training takes slightly more time to complete than with ResNet-50 and more GPU memory is needed, see Table I.

TABLE I
COMPARATIVE RESULTS OF DEEPER MODELS PERFORMANCE.

Model	F-measure	Training Time	Iterations	Memory
ResNet-50	95.76	2h00	24K	7GB
ResNet-101	95.88	2h30	20K	10GB

In the experiments over BEV-trained models, overfitting is observed in the learning curve (training losses decrease while validation ones do not) because the deeper model has more learning capacity and needs a larger training set to generalize. Therefore, the training is stopped at 20K iterations to avoid the problem and the obtained F-measure is better (96.13%). In conclusion, ResNet-101 offers a small but consistent improvement in detection performance, at the expense of needing more computing resources and time.

E. Upsampling variants

Different upsampling variants are evaluated in a ResNet-50 trained in perspective view. As expected, the detections with the full step-by-step upsampling scheme have the highest resolution, but they are noisier and the F-measure is worse (95.49%), probably because the extracted features come from too shallow layers with little knowledge of the full scene. In the four-step case, the resolution is higher than in the original setup and the F-measure is slightly upraised (95.80%). The four-step approach has also been tried with a ResNet-50 trained in BEV, and a ResNet-101 trained in perspective and in BEV spaces. Whereas the ResNet-50 gives similar results (95.97%), the ResNet-101 yields the best detections so far, with a F-measure of 96.09% and 96.31% in perspective and BEV spaces respectively. The dilated convolution approaches yield similar results. In particular, the method from [9] combined with two upsampling steps seems to be as good as the four-step approach in a ResNet-50 and less (20K) iterations, but it does not improve the results with the ResNet-101.

F. Final Results

Table II summarizes the quantitative results in F-measure over our KITTI validation subset for the most interesting network variants. The baseline algorithm is the ResNet-50 model with the fully convolutional architecture and three-step upsampling shown in Figure 1.

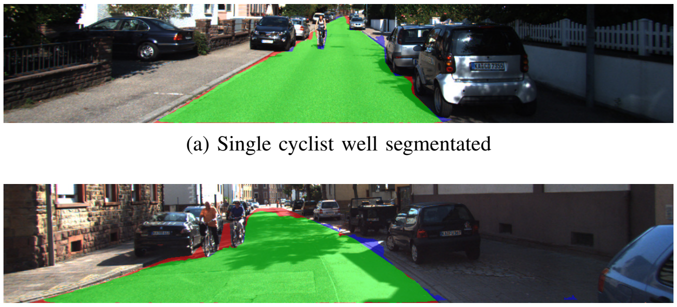
TABLE II
QUANTITATIVE RESULTS IN F-MEASURE ON KITTI DATASET.

Data aug.	BEV train	ResNet-101	4-step up.	F-measure
	✓			94.08%
				94.59%
✓				95.76%
✓			✓	95.80%
✓		✓		95.88%
✓	✓			96.06%
✓		✓	✓	96.09%
✓	✓	✓		96.13%
✓	✓	✓	✓	**96.31%**

The best-performing method, namely the ResNet-101 with data augmentation and four-step interpolation, is further analyzed, trained in perspective and in BEV. Small obstacles such as pedestrians, cyclists or cars are well differentiated from the road areas (Figure 4a), although two cyclists riding together are considered as a single obstacle (Figure 4b) since the space between them is not well segmented. This problem is also present when training in BEV and may be solved with higher resolution approaches.

Both models may leave FN gaps (Figure 5a and Figure 6a on top-right corner), as well as FP patches outside road limits (Figure 5b) that could be filtered with post-processing methods. However, the BEV-trained model seems to be better delimiting road limits in the same image (Figure 6b) because in this representation they are straighter.

It can be seen that the BEV-trained model is better at detecting irregular road limits (Figures 6c and 6d) than the perspective-trained one (Figures 5c and 5d). The main problem of the BEV approach is that in some cases the resulting image is so distorted that the net confuses buildings with the continuation of the road (Figure 6e). This would be unlikely to happen in a perspective-space analysis. (Figure 5e). Finally, this system has been trained on the full KITTI labelled set in BEV, achieving state-of-the-art performance on their benchmark reserved testing set.



(a) Single cyclist well segmentated

(b) Group of cyclists considered a single obstacle

Fig. 4. Results from the perspective-trained model in a scene with cyclists.

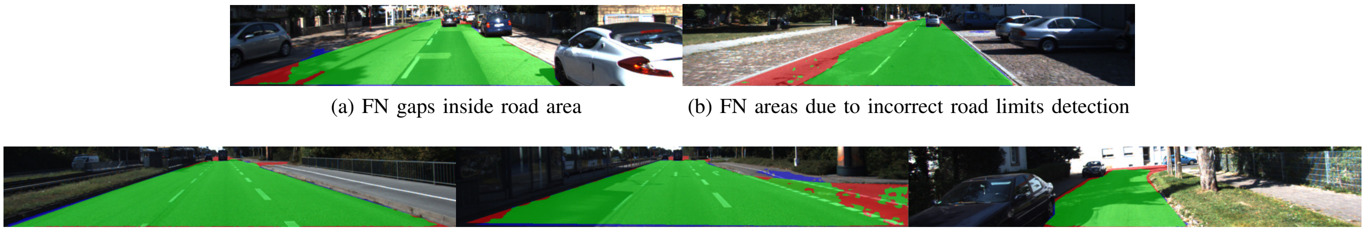
V. CONCLUSIONS & FUTURE WORK

An in-depth analysis of a deep learning-based road detection system is presented. It starts with the ResNet-50 network model with a fully convolutional architecture and three interpolation steps, finetuned in perspective KITTI images. Several variations are introduced to improve the training: data augmentation, training in BEV space, tuning training parameters, using deeper models and other upsampling architectures. Data augmentation offers a significant improvement between 1% and 2% in F-measure, and thus it is included in the rest of variations. These can lead to an additional consistent improvement over 0.5% if they are properly combined. Finally, the use of a ResNet-101 model with a four-step upsampling scheme, trained directly in BEV with data augmentation improves our results up to 96.31% on the validation subset, and achieves state-of-the-art results on the testing set. Nevertheless, the appropriate configuration depends on the final application and the tradeoff between performance and computing capacity. For future work, a post-processing layer will be added into the system to obtain smoother results and also high level information provided by digital navigation maps will be included to solve some of the problems described before.

ACKNOWLEDGMENT

This work was supported by Research Grants DPI2014-59276-R (Spanish Ministry of Economy), SEGVAUTO-TRIES-CM S2013/MIT-2713 (Community of Madrid) and BRAVE Project, H2020, Contract number 723021.

This project has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No 737469. This Joint Undertaking receives support from the European Union's Horizon 2020 research and innovation programme and Germany, Austria, Spain, Italy, Latvia, Belgium, Netherlands, Sweden, Finland, Lithuania, Czech Republic, Romania, Norway.



(a) FN gaps inside road area

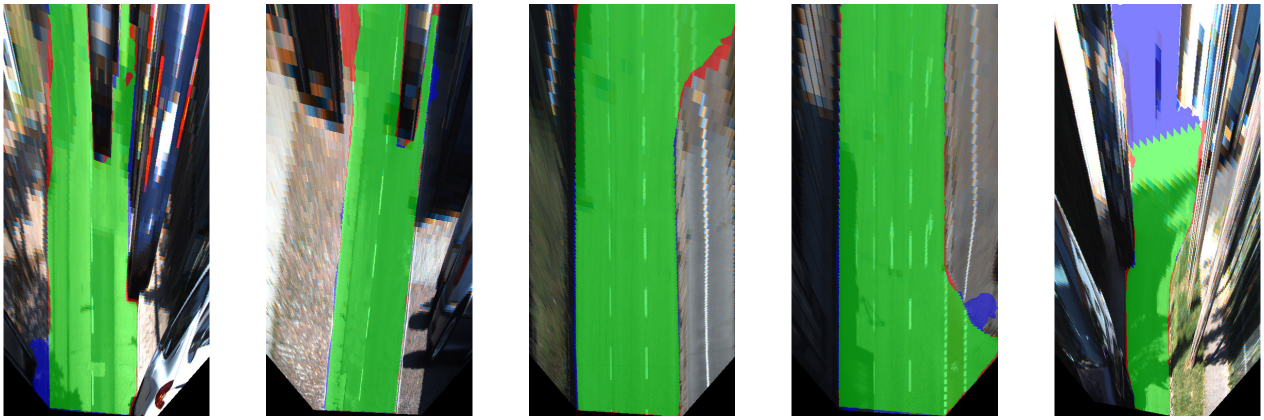
(b) FN areas due to incorrect road limits detection

(c) Missdetection of a distant intersection

(d) FN of an incoming lane

(e) Correct detection of the building

Fig. 5. Results from the perspective-trained model



(a) FP in parking areas in the closer meters

(b) Improved road detection

(c) Improved far intersection detection

(d) Improved near intersection detection

(e) FP detection of a building

Fig. 6. Results from the BEV-trained model

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv:1409.1556 [cs]*, Sept. 2014, arXiv: 1409.1556.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[4] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," *arXiv:1602.07261 [cs]*, Feb. 2016, arXiv: 1602.07261.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *arXiv:1506.01497 [cs]*, June 2015, arXiv: 1506.01497.

[6] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[7] E. Shelhamer, J. Long, and T. Darrell, "Fully Convolutional Networks for Semantic Segmentation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, Apr. 2017.

[8] Z. Wu, C. Shen, and A. v. d. Hengel, "High-performance Semantic Segmentation Using Very Deep Fully Convolutional Networks," *arXiv:1604.04339 [cs]*, Apr. 2016, arXiv: 1604.04339.

[9] 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," *arXiv:1606.00915 [cs]*, June 2016, arXiv: 1606.00915.

[10] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, "Understanding Convolution for Semantic Segmentation," *arXiv:1702.08502 [cs]*, Feb. 2017, arXiv: 1702.08502.

[11] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proc. IEEE Int. Conf. on Computer Vision*, 2015, pp. 1520–1528.

[12] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *arXiv:1511.00561 [cs]*, Nov. 2015, arXiv: 1511.00561.

[13] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid Scene Parsing Network," *arXiv:1612.01105 [cs]*, Dec. 2016, arXiv: 1612.01105.

[14] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, "Full-Resolution Residual Networks for Semantic Segmentation in Street Scenes," *arXiv:1611.08323 [cs]*, Nov. 2016, arXiv: 1611.08323.

[15] G. L. Oliveira, W. Burgard, and T. Brox, "Efficient deep models for monocular road segmentation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4885–4891.

[16] 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.

[17] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[18] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. of the 22nd ACM Int. Conf. on Multimedia*. ACM, 2014, pp. 675–678.

[19] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun, "Deep Image: Scaling up Image Recognition," *arXiv:1501.02876 [cs]*, Jan. 2015, arXiv: 1501.02876.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE Int. Conf. on Computer Vision (ICCV)*, 2015, pp. 1026–1034.

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