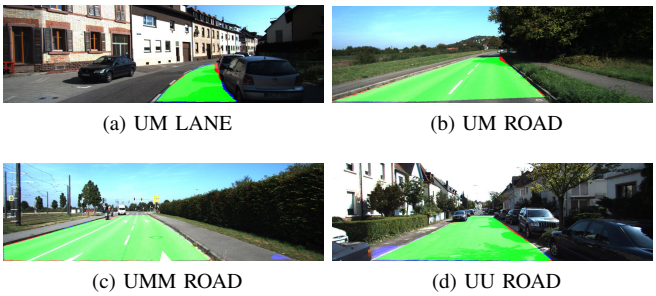
Efficient Deep Models for Monocular Road Segmentation

Gabriel L. Oliveira, Wolfram Burgard and Thomas Brox

***Abstract*— This paper addresses the problem of road scene segmentation in conventional RGB images by exploiting recent advances in semantic segmentation via convolutional neural networks (CNNs). Segmentation networks are very large and do not currently run at interactive frame rates. To make this technique applicable to robotics we propose several architecture refinements that provide the best trade-off between segmentation quality and runtime. This is achieved by a new mapping between classes and filters at the expansion side of the network. The network is trained end-to-end and yields precise road/lane predictions at the original input resolution in roughly 50ms. Compared to the state of the art, the network achieves top accuracies on the KITTI dataset for road and lane segmentation while providing a 20× speed-up. We demonstrate that the improved efficiency is not due to the road segmentation task. Also on segmentation datasets with larger scene complexity, the accuracy does not suffer from the large speed-up.**



(a) UM LANE (b) UM ROAD

(c) UMM ROAD (d) UU ROAD

Fig. 1: The four different lane/road segmentation tasks of the KITTI benchmark. The correct segmentation is shown in green, while the false negative and false positive detections are shown in red and blue, respectively.

I. INTRODUCTION

Road detection plays a crucial role in autonomous driving and intelligent transportation systems. Solutions to this problem are envisioned to reduce accidents and traffic and improve fuel efficiency. Thanks to so-called up-convolutional networks [7], [11], deep learning has become applicable also for segmentation problems. In contrast to usual classification Convolutional Neural Networks (CNNs), which contract the high-resolution input to a low-resolution output, up-convolutional networks take an abstract, low-resolution input and predict a high-resolution output, such as full-size images.

While these network architectures have dramatically increased the quality of semantic segmentation [11], [14], they are significantly slower than typical classification networks. The forward pass through these state-of-the-art networks requires between 150 and 229ms, which makes it impossible to use them at interactive frame rates in a robotics context.

In this paper, we focus on improving the efficiency of segmentation with a deep network by modifying the architecture in such a way that it is significantly faster, requires less memory, and still achieves the same accuracy as previous architectures. The key modification we propose is a new distribution of parameters at the up-convolutional part of the network. This modification saves lots of parameters, which improves the training time and leads to interactive frame rates on regular GPUs and makes this class of networks capable of running on low-power mobile GPUs.

We demonstrate the power of the proposed architecture in a visual road/lane segmentation task. Typical challenges in this task are illumination issues, such as shadows and reflections, dynamic background and non-trivial variation in appearance. Deep learning is predestined to deal with such challenges, which is why the state-of-the-art approaches are based on convolutional networks. However, they rely on patch training [4], [12]. This is suboptimal, because decisions are made on only a relatively small local patch. Moreover, a network must be run on many overlapping patches, which is inefficient. In contrast, the proposed architecture, which builds upon the work of Long *et al.* [11] and Oliveira *et al.* [14], works on the whole image and exploits all available context. Moreover, a single forward pass is necessary to segment the whole image, which makes the approach potentially more efficient than patch-based approaches.

We test our network on the KITTI-Road dataset [8]. The results show that the proposed technique achieves state-of-the-art performance with the lowest computational runtime of all top methods. We also performed tests to measure the robustness of the network to scale variations and compared our network with architectures from the same class [11], [14]. For both sets of tests we outperform the compared techniques, in terms of computational performance and accuracy.

II. RELATED WORK

Road segmentation has been attracting attention from the robotics and computer vision community for many decades. Several methods using a variety of sensors have been developed. Two seminal works are by Yu *et al.* [3] and Aly *et al.* [2]. Yu *et al.* [3] present an approach that localizes roads using the watershed algorithm. Aly *et al.* [2] use the Hough transformation to identify lane markings and to localize the road area.

All authors are with the Department of Computer Science at the University of Freiburg, 79110 Freiburg, Germany. This work has been supported by the European Commission under ERC-StG-PE7-279401-VideoLearn, ERC-AG-PE7-267686-LIFENAV, FP7-610603-EUROPA2, and by the Freiburg Graduate School of Robotics.

Monocular, vision based road segmentation is usually built upon learning methods. The first such attempts are limited to independently classify regions or pixels. These methods ignore the global properties provided by the whole image and typically misclassify regions with similar appearance. Global methods using Conditional Random Fields (CRF) [15], [19], Boosting Algorithms [18] and hierarchical approaches [13] were proposed to address this issue. Wu et al. [19] propose a CRF based approach for image road detection and an additional CRF to fuse the image segmentation with Light Detector And Range (LIDAR). Bergasa et al. [15] also employ a CRF approach with a miniaturized image calculated from superpixels. CRFs present local inference limitations because this class of methods only allow the direct influence of adjacent regions. Other methods like [13], [18] make use of hand-crafted features and hierarchical classifiers. The hierarchical road segmentation is performed by specific classifiers for each hierarchical level and uses the previous classification results as features for the next step.

Above methods dominated road and lane segmentation until recently. The unprecedented results obtained by CNNs for classification [9], [17] and segmentation [11] make CNNs interesting for almost all perception problems. Consequently, CNNs have been applied also to road segmentation [1], [4], [12]. Lopez *et al.* [1] introduce a road scene segmentation approach that learns a classifier based on hand-crafted features, creating the training samples for a CNN network. The network learns specific domain features based on the machine-generated annotations. Denzler *et al.* [4] introduce convolutional patch networks, which are CNNs designed to patch segmentation, allowing pixel-wise labeling. The technique also explicitly incorporates spatial information of the patch to the network, allowing incorporation of a spatial prior to the network. Mohan [12] presents a CNN architecture in combination with deconvolutions. He proposed a multi-patch technique that learns region-specific features, each patch region is used to train a separate network. This method currently provides the best results on the KITTI benchmark. While being less deep than our architecture, the proposed deconvolution network is computational costly and is not able to provide interactive frame rates.

In contrast to deconvolution networks, the so-called fully convolutional network (FCN) developed by Long *et al.* [11] allows training the network end-to-end for semantic segmentation tasks. This more elegant approach also led to better performance and provides the state-of-the-art performance in generic semantic segmentation. The approach replaces the fully connected layers of a deep classification network, e.g. VGG [17], by convolution layers that produce coarse score maps. Successive up-convolutional refinements allows them to increase the resolution of these score maps. There have been some recent extensions of Long et al. [11]. Chen *et al.* [5] use a fully connected CRF to refine the segmentation maps obtained from [11]. Oliveira *et al.* [14] applied a similar approach to human part segmentation and proposed several improvements with regard to over-fitting and segmentation of occluded parts in highly cluttered environments. While being much more efficient than patch based approaches, these previous works still do not achieve interactive frame rates. In this paper, we improve the efficiency of the network architecture to provide considerable speed-ups while keeping or even improving the accuracy of the results.

III. METHODOLOGY

A. Problem Definition

Road segmentation associates each pixel of an input image to one of two classes: road and non-road, i.e., it is a binary segmentation problem. The network can be easily modified to tackle multi-label problems by adding more output channels, yet in the context of road segmentation this is not necessary.

We approach the problem with a CNN that is trained end-to-end to predict a map of class labels. The output of the network are scores for each of the learned categories. We represent our architecture as one model $f(x, \gamma)$ that maps an image to the target segmentation. The model is described by the network parameters γ and is learned by minimizing its error output for an example x_i given an output ground-truth label y_i:

$$\hat{\gamma} = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{n} L\left(f\left(x_i, \gamma\right), y_i\right), \tag{1}$$

where n is the training set and L is the cross-entropy ($softmax$) loss, which converts a score a_K for class K into a posterior class probability $P_K \in [0, 1]$. When running the final network, the $softmax$ is replaced by the $argmax$ function to provide a single output class label.

B. Architecture

The architecture is based on the recently proposed fully convolutional networks [11], [14], i.e., the network has a contractive part, similar to a classification network, and a corresponding up-convolutional part that expands the representation to a high resolution segmentation.

Our modified architecture is shown in Fig. 2. The contractive network layer parameters are initialized using the VGG classification network [17]. Each refinement at the expansive layer has a corresponding layer in the contractive part with the same resolution. The output of the contractive part of the network is a low resolution feature map, from which the up-convolutional network must derive the high resolution segmentation. Each up-convolutional layer upsamples the input by a factor of 2 via bilinear interpolation. After each upsampling operation a ReLU is used to better deal with the vanishing gradient problem. The upsampled filters after passing through the ReLU serve as input to a successive convolution layer. One characteristic of the proposed expansive part is the inclusion of dropout after the first refinement layer to avoid overfitting.

The expansive network manages to produce high quality output given the coarse representation provided by the contraction side. The output of the network has the resolution of the input image. A detailed specification of the individual network layers is given in Table I.

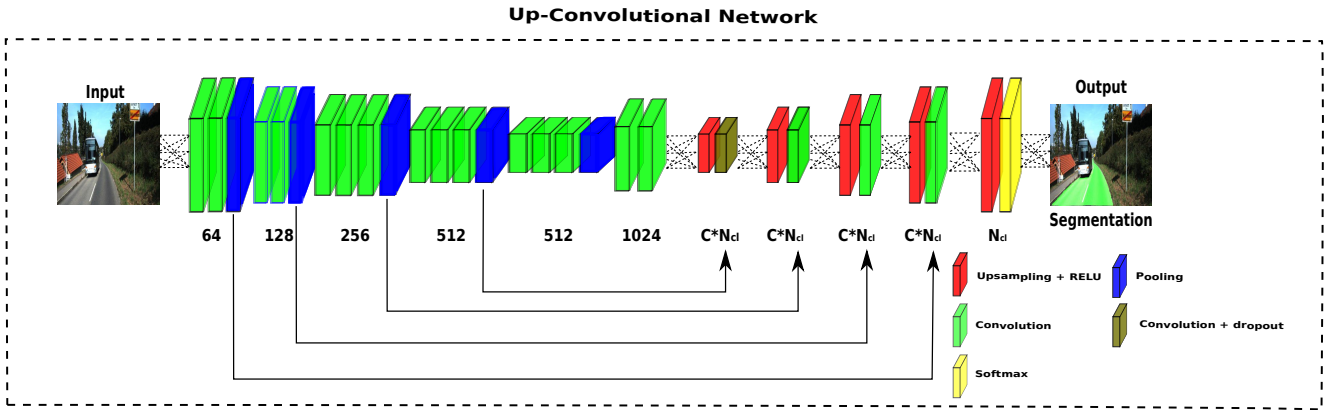


Fig. 2: Proposed architecture. Up-convolutional layers have size equal to $C * N_{cl}$, where N_{cl} stands for number of classes and C for the scalar factor of filters augmentation. We call the network part up to the first upsampling layer the contractive side of the network and the following portion the expansive network side.

name	kernel size	stride	pad	output size
data	-	-	-	$500 \times 500 \times 3$
conv1_1	3×3	1	10	$518 \times 518 \times 64$
conv1_2	3×3	1	1	$518 \times 518 \times 64$
pool1	2×2	2	0	$259 \times 259 \times 64$
conv2_1	3×3	1	1	$259 \times 259 \times 128$
conv2_2	3×3	1	1	$259 \times 259 \times 128$
pool2	2×2	2	0	$130 \times 130 \times 128$
conv3_1	3×3	1	1	$130 \times 130 \times 256$
conv3_2	3×3	1	1	$130 \times 130 \times 256$
conv3_3	3×3	1	1	$130 \times 130 \times 256$
pool3	2×2	2	0	$65 \times 65 \times 256$
conv4_1	3×3	1	1	$65 \times 65 \times 512$
conv4_2	3×3	1	1	$65 \times 65 \times 512$
conv4_3	3×3	1	1	$65 \times 65 \times 512$
pool4	2×2	2	0	$33 \times 33 \times 512$
conv5_1	3×3	1	1	$33 \times 33 \times 512$
conv5_2	3×3	1	1	$33 \times 33 \times 512$
conv5_3	3×3	1	1	$33 \times 33 \times 512$
pool5	2×2	2	0	$17 \times 17 \times 512$
FC-conv	3×3	1	0	$15 \times 15 \times 1024$
FC-conv2	1×1	1	0	$15 \times 15 \times 1024$
conv-N_{cl}	1×1	1	0	$1 \times 1 \times N_{cl}$
Up-conv1	4×4	2	0	$40 \times 40 \times CN_{cl}$
Up-conv2	4×4	2	0	$82 \times 82 \times CN_{cl}$
Up-conv3	4×4	2	0	$166 \times 166 \times CN_{cl}$
Up-conv4	4×4	2	0	$294 \times 294 \times CN_{cl}$
Up-conv5	4×4	2	0	$590 \times 590 \times N_{cl}$
output	-	-	-	$500 \times 500 \times N_{cl}$

TABLE I: Our architecture in more detail. The Up-conv layers refer to each refinement step. For brevity reasons ReLUs, dropout and some layers from the up-convolution step are omitted from the table.

C. Optimizing the Use of Parameters

The main motivation behind the proposed architecture was the need to design a network for road and lane segmentation that is efficient in terms of memory and runtime. To meet these requirements we optimized the number of network parameters.

1) Parameter reduction: The fully convolutional networks in Long et al. and Oliveira et al. [11], [14] use the VGG-16 classification network as basis for the contraction side of the network. This network has 4096 filters with $7x7$ spatial size. The large number of filter of large size is mainly responsible for the computational load.

To address this problem, we reduced the number of parameters by reducing the number of FC-conv filters from 4096 to 1024. In addition, we reduce the size of the filters from 7×7 to 3×3. The proposed reduction of network parameters makes our approach more efficient than the previously proposed dense segmentation architectures. From such a reduction, one must expect a significant drop in classification accuracy. However, we use some of the saved parameters at another part of the network to keep the high accuracy and even improve it compared to the baseline network.

2) New refinement to improve system accuracy: In order to make our network capable of producing accurate segmentation masks, after the substantial reduction in the FC-conv layer, we must strengthen other parts of the network. In particular, we increase the width of the up-convolutional side of the network. Previously our network has a $1 - to - 1$ mapping, each refinement has the same number of filters and classes (N_{cl}). Such configuration have as main drawback limiting the discriminative power of such architectures. We propose a new distribution of parameters, based on the U-nets [16], to overcome this drawback. U-nets have a variable number of filters, which are the same between the contraction and expansion side. Similar architecture was trained, however presented prohibitive time performance. In order to provide more parameters to the expansive part without hurting the computational time we proposed a similar approach that use multiple filters per class like U-net, but without the considerable increase of the network parameters. For this purpose we use a scalar C, which is empirically selected to multiply the number of filters. Such approach presented minimal time footprint and proved to make our new architecture more robust to scale.

While the new architecture has $C * N_{cl}$ filters, two parts of the network kept the same number of filters when compared to the classes. These parts are at the convolutional layer

between the contraction and expansion part of the network (conv-N_{cl}) and at the last layer of the architecture. The first layer's purpose is to maintain the network efficiency, while the last one has a goal of making the network calculate loss over only the useful classes. Figure 3 shows such changes.

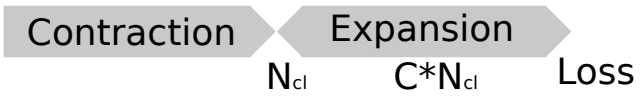


Fig. 3: Description of the new architecture weight distribution. The expansive side increases its parameter size by a factor of C. Only the convolutional layers between the contraction and expansive side and before the loss continue with N_{cl} filters.

D. Data Augmentation

Data augmentation is needed for road and lane segmentation due to the small number of training examples. To this effect, we employ a series of data transformations to the original data. In particular, we implemented:

- Scaling: scale the image by a factor between 0.7 and 1.4;
- Color: Add a value between -0.1 and 0.1 to the hue channel of the HSV representation.

For the specific application of road and lane segmentation, rotation and cropping transformations are undesirable, since the network is expected to learn spatial priors of the road. Rotation and cropping can hampers learning such priors.

E. Network Training

Training is performed in a multi-stage fashion. We initialized the contraction part of the network with the VGG architecture [17]. We also modified the network hyperparameters. We reduced the padding of the first convolutional layer from 100 to 10 pixels (slightly faster training), use Xavier initialization, higher learning rate, from $1e-10$ to $1e-9$ and lower momentum 0.90 instead of 0.99. We also changed the fixed learning rate (L_r) by a poly learning policy

$$L_r = L\left(1 - i/max_i\right)^p, \tag{2}$$

where L is the base learning rate, i is the learning step and p is the power index. The new policy converges faster than the fixed learning rate policy. On average we need half the number of iterations (300k iterations vs 600k iterations) to obtain the same results.

The network is trained by backpropagation using stochastic gradient descent (SGD) with momentum. Each minibatch consists of just one image. The training is done one refinement stage at a time and each refinement takes one day. Thus, training the whole network took 5 days on a single GPU.

IV. EXPERIMENTS

We evaluated the performance of the optimized architecture on real driving data from the KITTI benchmark dataset. We present a series of evaluations in terms of runtime, accuracy, and scale robustness. Additionally, we compare the optimized architecture to the non-optimized baseline on generic segmentation problems without the specialization on road segmentation. The implementation was based on the publicly available Caffe [10] deep learning toolbox, and all experiments were carried out with a system containing an NVIDIA Titan X GPU.

A. KITTI Road/Lane Dataset

The KITTI Visual Benchmark Suite [8] is a dataset designed to benchmark optical flow, odometry data, object detection, and road/lane detection. The road dataset consists of 600 frames of 375×1242 pixels and constitutes the main benchmark dataset for road and lane segmentation. The data was acquired in five different days.

The dataset has three different categories of road scenes: single-lane road with markings (UM), single-lane road without markings (UU), and multi-lane road with markings (UMM). In this paper, we deal with road and ego-lane detection. We do not differentiate between the road categories, but the ego-lane problem is trained separately. The dataset provides ground truth for training and online evaluation for the testing[1]. The KITTI online evaluation system allows for anonymous submission of results, thereby some of the top ranked methodologies do not have a corresponding publication. Additionally the evaluation system restricts the number of submissions, making the evaluation at multiple resolutions not feasible.

1) Road Detection: Road segmentation for KITTI dataset is divided into four benchmark outputs: UM, UU, UMM and URBAN ROAD. URBAN ROAD is the category that summarizes all three different road scene categories. Table II shows our results for road segmentation. The individual methods are ranked according to their pixel-wise maximum F-measure on the Bird's-eye view space. The other provided measurements are: Average Precision (AP), Precision (PRE), Recall (REC), False Positive Rate (FPR) and False Negative Rate (FNR).

TABLE II: Results for the road KITTI dataset.

Benchmark	MaxF	AP	PRE	REC	FPR	FNR
UM	92.20%	88.85%	92.57%	91.83%	3.36%	8.17%
UMM	95.52%	92.86%	95.37%	95.67%	5.10%	4.33%
UU	92.65%	89.20%	92.85%	92.45%	2.32%	7.55%
URBAN	93.83%	90.47%	94.00%	93.67%	3.29%	6.33%

The single-lane with markings category (UM) is formed by images taken from a marked urban two-way road and has 95 images for training and 96 images for testing. Our approach ranks second for this category, as seen in Table III. While all top results for this category report processing times of about 2 seconds, our architecture has an average runtime of 83 milliseconds. This makes it the only approach among the top performing methods that is capable of interactive frame rates.

For the single-lane road without markings (UU) we have 98 images for training and 100 images for testing. Our method ranks first and second for this set; see Table IV. The resolution

[1] http://www.cvlibs.net/datasets/kitti/eval_road.php

TABLE III: Results on UM road KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
DDN [12]	**93.65%**	88.55%	94.28%	93.03%	2.57%	6.97%	2s
Ours	92.20%	88.85%	92.57%	91.83%	3.36%	8.17%	**83ms**
CNN1	91.73%	92.08%	91.10%	92.36%	4.11%	7.64%	2s
CNN	91.22%	91.35%	91.22%	91.23%	4.00%	8.77%	2s

TABLE IV: Results on UU road KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
Ours	**92.65%**	89.20%	92.85%	92.45%	2.32%	7.55%	**83ms**
Ours-Low	**91.89%**	89.44%	92.59%	91.20%	2.38%	8.80%	**52ms**
DDN [12]	91.76%	86.84%	93.06%	90.50%	2.20%	9.50%	2s
CNN1	89.70%	90.61%	89.41%	89.99%	3.47%	10.01%	2s

TABLE V: Results on UMM road KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
Ours	**95.52%**	92.86%	95.37%	95.67%	5.10%	4.33%	**83ms**
DDN [12]	94.17%	92.70%	96.73%	91.74%	3.41%	8.26%	2s
FCN_LC	94.09%	90.26%	94.05%	94.13%	6.55%	5.87%	**30ms**
Ours-Low	93.89%	92.62%	94.57%	93.22%	5.89%	6.78%	52ms

TABLE VI: Results on URBAN_ROAD KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
Ours	**93.83%**	90.47%	94.00%	93.67%	3.29%	6.33%	**83ms**
DDN	93.43%	89.67%	95.09%	91.82%	2.61%	8.18%	2s
Ours-Low	92.39%	90.24%	93.03%	91.76%	3.79%	8.24%	**52ms**
CNN1	91.98%	92.44%	91.08%	92.89%	5.01%	7.11%	2s

TABLE VII: Results on UM lane KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
Ours	**89.88%**	87.52%	92.01%	87.84%	1.34%	12.16%	83ms
ANM	89.11%	81.11%	88.68%	89.54%	2.01%	10.46%	60ms
PCA-Lane-S	87.01%	74.16%	87.31%	86.70%	2.22%	13.30%	**30ms**
S	85.15%	76.52%	88.61%	81.95%	1.85%	18.05%	100ms

TABLE VIII: Runtime depending on the GPU.

GPU	Forward Pass Time (ms)
TK1	1440
TX1	599
GTX 680	97.4
K-40	108
GTX TITAN	96.8
GTX 970	66.4
GTX 980	**51**
GTX TITAN X	**52.2**

TABLE IX: Runtime depending on the resolution.

Resolution	Forward Pass Time (ms)
150×150	30
200×200	35.6
300×300	52.2
500×500	83

used for most of the experiments was 500×500 but we also tested with a 300×300 resolution. More tests on the impact of resolution will be discussed in Section IV-A.3.

The third setting is composed of images taken from the urban multi-lane marked road (UMM), which has 96 images for training and 94 for testing. Results are shown at Table V, where the proposed network compares favorably to all existing techniques, too. For this scenario a CNN method (FCN_LC) presents faster processing times, yet the reported accuracy is inferior in all metrics.

The final metric of the KITTI evaluation benchmark is one that combines all three experimental settings (URBAN_ROAD). Table VI presents our results and the best results available. The proposed network achieves the highest accuracy while its runtime is smallest among all top performing methods. Figure 4 illustrates some obtained segmentations between our architecture and the top two results. We consider the top URBAN_ROAD techniques for comparison, since this metric can provide a better overall measurement of how well a method behaves. The proposed architecture shows low occurrences of false positive predictions and sharp segmentation of edges, while keeping an interactive frame rate capability.

2) Lane Detection: Lane detection is a challenging task due to low inter-class variability. Without context, the ego-lane is hard to distinguish from other asphalt parts of the road. The KITTI dataset uses single-lane road marked images to segment lanes. Table VII shows our results. In contrast to road detection, there are methods with runtimes in the millisecond range. The proposed network, which is the only one among the top techniques that was not specially designed for this task, achieves the best accuracy. Figure 5 presents some qualitative results comparing with the 3 best approaches. The proposed architecture visually exhibits much better lane segmentations when compared to the next best approaches.

3) Performance Tests: Since the runtime depends much on the hardware and resolution, we present results on various GPUs and at different resolutions. We tested on six desktop GPUs and two mobile ones; see Table VIII. Even on older GPUs, such as the GTX 680, the architecture achieves more than 10 frames per second and fits into the GPU memory. The tests reveal that the network is as fast on a GTX 980 as on a GTX TITAN X.

We further extended our experiments to modern low power mobile GPUs. In our experiments we tested the proposed architecture on two mobile GPUs, Tk1 and Tx1, respectively. Our network is capable of running at speeds faster than one frame per second in the TX1 and, to the best of our knowledge, is the first up-convolutional network capable of local processing in low power mobile GPUs.

We also modified the resolution. For this experiment we used a machine with a TITAN X GPU and tested inputs ranging from 150×150 to 500×500; see Table IX. As to be expected, higher resolutions increase the runtime. However, even at a 500×500 resolution, our system is still more efficient than any top result for road detection.

B. Range Experiments

We explicitly test how the network behaves when exposed to multiple scales. The KITTI dataset does not provide any specific data for measuring range robustness. Thus, we tested our architecture on a dataset from Oliveira et al. [14] designed to measure scale in the context of human part segmentation. The same testing methodology was employed: the network was trained on the PASCAL parts dataset [6] and tested on the range data. In order to deal with the additional classes in this task, we added the corresponding number of output channels to the network.

The dataset has two persons on distances ranging from 0.8 m to 6.0 m, capturing images every 20 cm. Figure 6

(a) Ours

(b) DNN [12]

(c) CNN1

Fig. 4: Segmentation results for road segmentation extracted from the KITTI benchmark. The proposed architecture shows low occurrences of false positive predictions and sharp segmentation of edges. Green correspond to correct segmentation, red to false negative and blue to false positive detections.

Fig. 5: Lane predictions on the KITTI dataset compared with the 3 next best approaches. Top Left: Our approach, Top Right: ANN, Botton Left: PCA-Lane-S, Botton Right: S. In the images green is correct segmentation, red false negative detection and blue positive detection.

presents our results and compares it to Oliveira et al. [14]. The new architecture consistently performs better than the baseline, specially for longer distances. The smaller filters 3×3 at the last two layers of the contraction side of the network provide a smaller field of view and present a gain for longer distances. For distances beyond 4 meters, our architecture largely outperforms the current state of the art.

C. Impact of Parameter Reduction and New Refinement

In order to analyze the impact of parameter reduction and the new mapping of the expansion side we perform experiments with each of these settings. We incrementally tested each of them and also compared to other fully convolutional networks. Based on the restriction in the number of submissions to the KITTI dataset, we use the PASCAL Parts dataset [6] with 4 body parts. PASCAL parts provides a more complex multi-label scenario and allows testing the general purpose capabilities of the proposed architecture.

Table X shows our results compared to other fully convolutional networks. Although our network is much faster, we obtained state of the art results. This demonstrates that the network provides not only a very efficient solution for road and lane segmentation, but it is generally applicable for segmentation problems at high frame rates. Additionally we also quantify the impact of parameter reduction, new refinement and the complete architecture. As expected, the reduction of parameters results in a decrease of quality in the segmentation, however such reduction speed-up the network 4.8 times when compared to our base network. The addition of the new refinement parameter distribution raises our network results by 4.25 mean IOU percentage points, yet makes the network slower. The full architecture, which includes all the previous settings and the reduction of parameters between the contraction and expansive side only increases a few milliseconds the forward pass, when compared to our fastest result. The full architecture yields state of the art results and comes with a computation requirement close to the lowest tested configuration.

TABLE X: Results on the PASCAL dataset with 4 body parts.

Method	IOU Head	Torso	Arms	Legs	All	Time
FCN [11]	70.74	60.62	48.44	50.38	57.35	150ms
Up-Conv [14]	83.24	79.41	73.73	76.52	78.23	229ms
Ours - Parameter reduction	82.61	78.78	72.83	74.84	77.00	47.3ms
Ours - New refinement	84.67	82.30	77.56	79.62	81.25	53ms
Ours - Full architecture	**84.86**	**82.90**	**78.35**	**80.95**	**81.92**	48.7ms

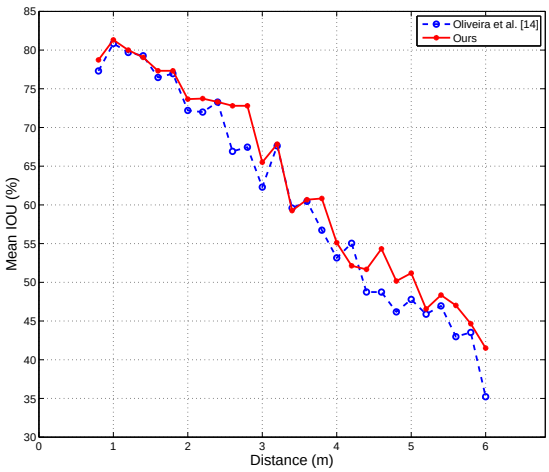


Fig. 6: Range segmentation results. The proposed approach consistently performs better than the baseline, specially for longer distances.

V. CONCLUSION AND FUTURE WORK

We presented a near real-time deep learning methodology for road and lane segmentation using up-convolutional networks. The main motivation behind this new architecture was the need to design an approach for road and lane segmentation, which is efficient in terms of memory and runtime. For that we proposed a modification of the architecture that saves many parameters in one part of the network and introduces only few new parameter in another part. This new distribution not only speeds up the network by the overall reduction of parameters but also produces better segmentations by having more filters at the expansion side of the network. The experiments showed that the proposed technique advances the state of the art for road and lane segmentation on the KITTI dataset also presenting speed gains of more than 20 times when compared to the previous top road segmentation results. We also showed the network keeps its advantage with regard to speed and segmentation accuracy when applied to a different semantic segmentation problem.

Future works will include investigating the potential gain of incorporating other sensors to the architecture and approaches to deal with network data fusion and three dimensional road segmentation. We also aim to investigate training even smaller architectures by model compression to further improve the system frame rates. Aspects related to robustness to different seasons can be explored to provide reliable road segmentation in all weather situations.

REFERENCES

[1] Jose M. Alvarez, Theo Gevers, Yann LeCun, and Antonio M. Lopez. Road scene segmentation from a single image. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VII*, ECCV'12, pages 376–389, Berlin, Heidelberg, 2012.

[2] M. Aly. Real time detection of lane markers in urban streets. In *Intelligent Vehicles Symposium, 2008 IEEE*, pages 7–12, June 2008.

[3] S. Beucher, M. Bilodeau, and X. Yu. Road segmentation by watershed algorithms. In *PROMETHEUS Workshop*, 1990.

[4] Clemens-Alexander Brust, Sven Sickert, Marcel Simon, Erik Rodner, and Joachim Denzler. Convolutional patch networks with spatial prior for road detection and urban scene understanding. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2015.

[5] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. *ICLR*, 2015.

[6] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *CVPR*, 2014.

[7] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Learning to generate chairs with convolutional neural networks. In *CVPR*, 2015.

[8] Jannik Fritsch, Tobias Kuehnl, and Andreas Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[10] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[11] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, nov 2015.

[12] Rahul Mohan. Deep deconvolutional networks for scene parsing. *CoRR*, abs/1411.4101, 2014.

[13] Daniel Munoz, J. Andrew Bagnell, and Martial Hebert. Stacked hierarchical labeling. In *Proceedings of the 11th European Conference on Computer Vision: Part VI*, ECCV'10, pages 57–70, 2010.

[14] Gabriel Oliveira, Abhinav Valada, Claas Bollen, Wolfram Burgard, and Thomas Brox. Deep learning for human part discovery in images. *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[15] M. Passani, J.J. Yebes, and L.M. Bergasa. Crf-based semantic labeling in miniaturized road scenes. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 1902–1903, 2014.

[16] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

[18] Giovani Bernardes Vitor, Alessandro C Victorino, and Janito V Ferreira. A probabilistic distribution approach for the classification of urban roads in complex environments. In *Workshop on IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[19] Liang Xiao, Bin Dai, Daxue Liu, Tingbo Hu, and Tao Wu. Crf based road detection with multi-sensor fusion. In *Intelligent Vehicles Symposium (IV), 2015 IEEE*, pages 192–198, June 2015.