Robust Deep-Learning-Based Road-Prediction
for Augmented Reality Navigation Systems at Night

Matthias Limmer[1]*, Julian Forster[1]*, Dennis Baudach[1], Florian Schüle[2],
Roland Schweiger[1] and Hendrik P.A. Lensch[3]

***Abstract*— This paper proposes an approach that predicts the road course from camera sensors leveraging deep learning techniques. Road pixels are identified by training a multi-scale convolutional neural network on a large number of full-scene-labeled nighttime road images including adverse weather conditions. A framework is presented that applies the proposed approach to longer distance road course estimation, which is the basis for an augmented reality navigation application. In this framework long range sensor data (radar) and data from a map database are fused with short range sensor data (camera) to produce a precise longitudinal and lateral localization and road course estimation. The proposed approach reliably detects roads with and without lane markings and thus increases the robustness and availability of road course estimations and augmented reality navigation. Evaluations on an extensive set of high precision ground truth data taken from a differential GPS and an inertial measurement unit show that the proposed approach reaches state-of-the-art performance without the limitation of requiring existing lane markings.**

I. Introduction

Augmented reality navigation applications that support drivers navigating in unknown environments are one example of future *advanced driver assistance systems* (ADAS). Although this ADAS function is aimed mostly at urban navigation, where the difficulty lies in navigating in a complex road network, another use case is inter-urban navigation, especially for poor visibility conditions (e.g., fog, snow, night, ...). Regular navigation applications leverage a map database and a GPS sensor for coarse localization. Augmented reality applications, however, not only require a precise localization but also a precise road course estimation. Accurate lateral localization and shorter distance road course estimation is particularly important for realistic augmentations of the camera image. Common approaches exploit existing lane and road markings for this task (cf. [1], [2]). Lane and road markings, though, might not be usable or available for all inter-urban roads because of damage, soiling or simple absence.

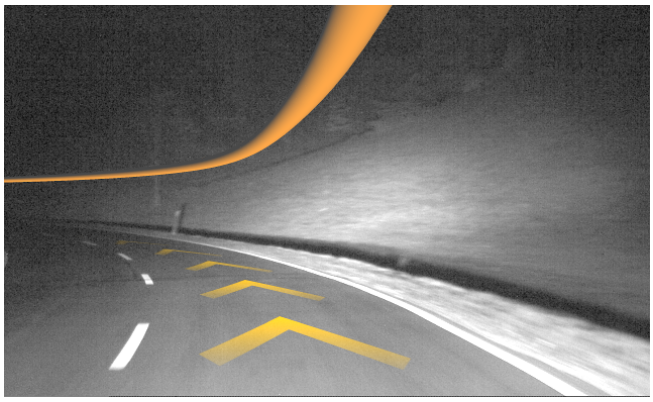

Fig. 1: An augmented reality navigation application. The orange tube displays the longer distance road course while the arrows are pinned to the road surface and augment the short distance road course. Figures are best viewed in color.

The image-based road detection approach presented in this paper classifies each pixel with a deep multi-scale *convolutional neural network* (CNN). The CNN learns feature extractors to identify road pixels in an integrated fashion. It is therefore capable of reliably classifying road pixels disregarding the presence of lane markings. Classified road pixels are homogenized by a floodfill algorithm to create a coherent road segment. A road contour is extracted from that segment and fitted into a spline-based road model, the *optical map*. This optical map is fused with processed data from a map database, the *digital map*, and a *grid map* from a radar sensor to conduct a precise localization and road course estimation. This is used in the augmented reality navigation system depicted in Fig. 1.

The approach is trained on a large number of full-scene-labeled *near infrared* (NIR) images showing nighttime road scenes including adverse weather conditions. Localization and road prediction results are evaluated in extensive experiments against ground truth trajectories measured by a high precision *inertial measurement unit* (IMU) and *differential GPS* (D-GPS). Evaluation results show state-of-the-art performance compared to a baseline approach [3], but no failures when lane markings are not available. This increases the robustness and availability of the application.

This work was partially funded by the European Commission under the ECSEL Joint Undertaking in the scope of the DESERVE project. http://www.deserve-project.eu/

[1] M. Limmer, J. Forster, D. Baudach and R. Schweiger are with Daimler AG R&D, Ulm, Germany

[2] F. Schüle is with the Institute of Measurement, Control and Microtechnology, University of Ulm, Germany

[3] H. Lensch is with the Department of Computer Graphics, Eberhard Karls Universität, Tübingen, Germany

*These authors contributed equally to this work

II. Related Work

Using digital maps as a source for road course estimation at longer distances requires accurate localization. The precision of common GPS sensors of up to 10 m for localization satisfies the needs of regular navigation systems but not those of a precise road course estimation, especially for augmented reality navigation [1]. To achieve a higher precision for longitudinal and lateral localization, road course estimation applications fuse multiple sensors with longer and shorter perception ranges. An exhaustive overview of different sensor fusion approaches is collated in [4]. In the following, a few approaches are introduced that are closer related to the scope of this paper.

Tsogas et al. [5] fuse measurements of a camera, laser scanner and a digital map based on the *clothoid* road model. A Sugeno-fuzzy system determines appropriate weightings for each of the different sensors dependent on the prediction distance from the ego-vehicle and the range of the sensor. The clothoid model, though, is only able to model cubic road curvatures. Complex curvatures, which commonly reside in arbitrary rural roads, can only be represented by joining several clothoids together. This, however, would increase the parameter space considerably and is not modeled by the aforementioned approach.

The sensor fusion system of Deusch et al. [2] is not dependent on the clothoid model and the digital map. It belongs to the category of systems that record a custom map containing landmarks and sensor data that can be used to localize the car later on. Coordinates from a D-GPS sensor are mapped to landmarks extracted from forward and backward looking cameras and the occupancy grid of a laser scanner. In a recall phase, the regular GPS-position is refined by matching concurrently extracted landmarks to those in the database. The creation of a landmark database, though, is a procedure that needs to be completed in advance. Moreover, maintenance of the database has to be performed on a regular basis to remove landmark errors because of construction works, etc.

Schüle et. al. [1] describe a framework that fuses a NIR camera sensor, a radar sensor and a digital map. It performs longitudinal localization by fusing a radar grid map and a digital map using a particle filter. Precise lateral localization is then accomplished by fusing the longitudinally mapped digital map with an optical lane recognition algorithm [3] in the camera image. In subsequent works [6], a Bayesian fusion system that performs the final road course estimation is introduced. In both systems, the road course model is not a clothoid but rather lists of connected 2D points sampling the right and left borders of the lane. This approach, as well as all aforementioned approaches, relies on lane marking detectors for the estimation of an optical map. To increase the robustness and availability of such a system, an optical road course recognition is desired that works independent of lane or road markings.

Seo et al. [7] describe a road boundary estimator based on intensity distribution thresholding from camera images. The thresholded intensity distribution is extracted from a *region of interest* (ROI) on the inverse perspective mapped camera image. Extracted road boundaries are tracked over time by a Bayes filter. Although the thresholding method is a simple and efficient approach for detecting road pixels in color images, it might fail for grayscale night vision images.

Fernández et al. [8] perform road detection by training decision trees. They use the disparity features of a stereo camera for a ground plane detection and several handcrafted color and texture features to classify superpixels segmented by a watershed transform. This approach, though, strongly relies on features not available for grayscale NIR monocular camera images.

Alvarez et al. [9] describe a road scene segmentation from single images using a convolutional neural network. The CNN is trained on publicly available annotated road scenes that are not necessarily images from the camera used in the application. To overcome this and allow adapting to immediate situations, the CNN classification is fused with the color intensity distribution from an ROI ahead of the vehicle through a Bayesian framework. Recent developments of CNN classifiers, though, show a high transferability of features [10]. If the input data between the sensors is similar enough, a CNN can be pre-trained on a big dataset (e.g. ImageNet [11] or Cityscapes [12]) and adapted to the current sensor with only fraction of the data needed to train it from scratch. Apart from that, this approach is not directly suitable for a road course extraction. Other road users that possibly occlude parts of the road are not explicitly classified, what complicates road border extractions in these cases.

Recently, many CNN topologies have been developed for image classification and subsequently image segmentation tasks [13]–[17]. Used in an image segmentation setup, the original formulations of these topologies commonly do not retain the image resolution in their output. Long et al. [10] introduced a way to recover resolution by repeatedly deconvolving the subsampled output and combining it with feature maps of higher resolution from earlier layers of the network. This method, though, requires to iteratively train each deconvolution layer after another, which is not end-to-end trainable.

To avoid training these deconvolution layers and reduce the inherent higher computational cost, Badrinarayanan et al. [18] introduce a technique that traces back max pooling activations to perform a smart upsampling of the low resolution pixel classifications. Despite its advantages, this approach shows a reduced classification performance compared to rivaling network architectures (c.f. [12]).

Another way of retaining resolution is the fragmentation scheme of Giusti et al. [19], which removes the subsampling property of pooling layers and substitutes

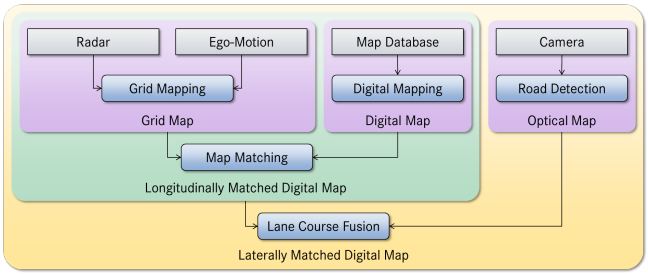

Fig. 2: Processing units of the proposed framework.

that with a spatial reordering of the pixels into fragmented feature maps. This approach was also picked up by [20] and exists in modified formulations as *Dilated Convolution* [21], *A-trous Convolution* [22] or *Strided Kernels* [23]. Thom and Gritschneder [24] proof that by using fragmentation, patch-based CNNs can be transformed into computational more efficient image-based CNNs (FCN), that produce equivalent results to patch-based networks evaluated at every possible pixel location.

The framework proposed in this paper is based on [1], but replaces the optical lane detection module from [3] with a road segmentation module based on deep multi-scale CNNs. These CNNs are structurally based on [14] and [15] and use an implementation of the fragmentation scheme [19] during inference. They are trained on a dataset of night-time images with a large variety of road and weather situations with and without lane markings. This approach therefore increases the robustness and availability of shorter distance road course estimations to situations without lane markings or adverse weather situations.

The remainder of this paper is structured as follows: Section III describes the framework, while Section IV presents the road detection module in more detail. Extensive experiments are described and discussed in Section V. Section VI summarizes the results.

III. Framework

To compute a reliable localization and road course estimation, a framework that fuses different sensor inputs is needed. This paper leverages a derivation of the framework from [1] and is depicted in Fig. 2. The modules of the framework are as follows: First, radar data in combination with a tracked ego-motion estimation produces a grid map. Second, initialized by the GPS position, the rough location in a commercially available map database is determined and map parameters for that location are transformed into a compliant digital map model. Third, the grid map and the digital map are fused to produce a longitudinally matched digital map. The fourth module performs road detection in a corresponding camera image and produces an optical map. Finally, the optical map and the matched map are fused into a laterally matched digital map.

A. Grid Mapping

A grid map is a 2D map representing the local environment quantized into equally sized cells representing occupancy (see Fig. 3). Each cell temporally integrates respective sensor measurements from a distance measuring sensor and thereby reduces the inherent noise and uncertainties of singular measurements. In the proposed framework, data from an imaging automotive radar, which returns both, the distances of reflections and their velocities, is stored in the grid map. Since the ego-vehicle is moving, its relative position on the grid map needs to be determined by estimating the ego-motion. An extended Kalman filter with a CTRV-model (constant turn rate and velocity [25]) leverages the wheel speeds and yaw rate measurements to accomplish an ego-motion estimation. The ego-motion estimation is then used to determine the correct cells where static radar objects are stored and integrated over time.

B. Digital Mapping

A commercial map database commonly stores its information in annotated discrete shape points using the UTM (Universal Transverse Mercator) coordinate system. The amount of points per road, the accuracy of such points and the meta-information per point varies greatly, since major roads are better sampled and maintained by database providers. To obtain a continuous local digital road model, shape points around the current ego-vehicle's location are interpolated by a cubic hermite spline. This creates the digital map, which serves as the base for the following fusion modules.

C. Map Matching

To estimate the orientation and longitudinal position of the ego-vehicle on the digital map, the grid map is fitted into the digital map using a particle filter. Each particle of the filter represents the position and orientation of the vehicle and is weighted by how well the digital map and the grid map fit using various features [26]. The sampling of the particles is initialized by the previous position or the GPS position if no previous position is available.

D. Road Detection

The original *Optical Lane Detection* module [3] in the framework of [1] is replaced with the lane-independent *Road Detection* module proposed in this paper. In this processing step, pixels in a camera image belonging to the currently traveled road are identified. These detected pixels are used to determine the road boundaries which are then transformed into and tracked by the optical map. Further details of this processing step are described in Section IV.

E. Lane Course Fusion

To increase the precision of the lateral localization, the optical map is fused with the digital map. Therefore,

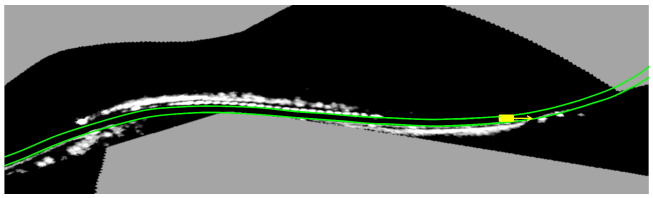

Fig. 3: An example of a grid map. The ego-vehicle and its travel direction are displayed as the orange box with the arrow. The white and gray dots show integrated values of reflections from the radar sensor. These reflections are generated primarily by grass, curbs or barriers at the side of the road. The green lines are splines of the digital map road shape points matched to the grid map.

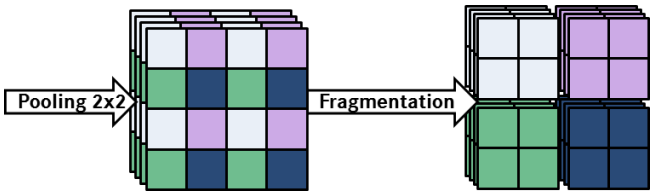

Fig. 5: Fragmentation of a 2×2 pooling. The interleaved feature map pixels of an overlapping pooling are reordered to produce 4 independent subsampled feature map arrays.

lateral coordinates in the ego-vehicle's coordinate system are sampled from both maps along the longitudinal trajectories of lane or road borders. Corresponding lateral positions are linearly interpolated by weighting each sensor according to its reliability for different distances from the ego vehicle. The optical map is very reliable for close distances while the digital map is more reliable for larger distances. The specific weighting scheme is described in [1].

IV. Road Detection

The road detection module described in the following identifies the currently traveled road in a camera image by performing a pixel classification using deep learning techniques. It then extracts and tracks the left and right road border taking into account uncertainties and border-occlusions by other road users. It then computes the optical map that can be fused with the digital map.

A. Scene Labeling

The scene labeling module proposed in this paper is a deep multi-scale CNN. It combines the approach of [15] with the multi-scale scheme of [14]. [15] introduces network topologies characterized by many convolution layers with small convolution kernels and comparatively few pooling layers. Many convolution layers increase the amount of non-linearities and thus the capability of the network to learn complex classification functions. If small convolution kernel sizes are used, the increase of convolution layers does not necessarily lead to a drastic increase of computational complexity. Multiple scales further improve the scale-invariance of the network without increasing its depth. Smaller input patches for each scale can be used than for deeper single-scale CNNs because the input patches of higher scales de facto cover a larger context of the input image. Multiple scales also increase data parallelism over sequential networks, which is often exploited by parallel hardware (e.g. GPUs) to increase its occupancy. Since real-time performance is needed for augmented reality applications, techniques from [19] and [24] are implemented for a computational efficient application of a CNN to entire images.

Multi-scale neural networks process multiple scales of the same input data concurrently. In this approach, an image pyramid of n_l levels is constructed by reducing the image resolution by 0.5 in both dimensions for each new level. This is the smallest integer downscaling factor. Each pyramid level is then normalized to zero-mean unit-variance in a local neighborhood, which enhances the texture and equalizes bright and dark areas in the image. The normalized image pyramid levels are then fed to their respective branches of the neural network. All branches of the network are built with the same structure and are finally joined in a fully-connected layer that also serves as the output layer of the network. A diagram of possible network topologies of the above defined multi-scale CNN is depicted in Fig. 4.

Though each branch is structurally identical, no weights are shared between the branches. A branch overall consists of alternating n_p pooling layers and $n_b = n_p + 1$ *convolution layer blocks*. Every convolution layer block consists of n_c convolution layers that use the ReLU function: $\mathrm{ReLU}(x) = \max(0, x)$ as activation function. The size of the filter bank n_f is identical within each convolution layer block and is doubled after each pooling layer. The kernel size of the convolution kernels k_c is the same in all convolution layers.

In a patch-based application of the proposed network, correctly sized image patches need to be extracted from the normalized image pyramid levels prior to feeding them to the branches. Applying the CNN efficiently to complete images while retaining the full image resolution requires slight changes in various layers and the introduction of several helper layers into the network (see [19] and [24]). These changes are explained in the following.

1) Overlapping Pooling: In an image-based application, pooling layers, which are normally strided ($\prod \mathrm{stride}_{k_p} > 1$) according to their kernel size k_p, need to be applied in an overlapping fashion ($\prod \mathrm{stride}_{k_p} = 1$), so that no resolution is lost.

2) Fragmentation: Fragmentation layers need to be inserted after each pooling layer. They split the oversized feature maps of the preceding layer into $\prod \mathrm{stride}_{k_p}$ feature maps of reduced resolution, which are processed individually afterwards. This ensures that the subsampling property of the pooling functions is preserved without loosing resolution. Fig. 5 depicts a 2×2 fragmentation.

3) Defragmentation: A defragmentation layer is needed after the last convolution block before all branches are joined. It reverts all performed fragmentations and transforms the fragmented feature map arrays

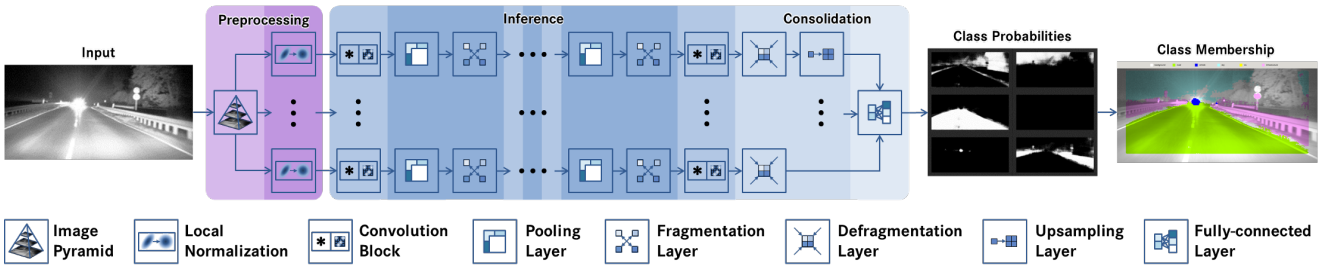

Fig. 4: Diagram of possible network architectures. A preprocessing step generates a normalized image pyramid of n_l levels. Each level performs an inference in its own CNN branch that consists of alternating convolution blocks and pooling/fragmentation layers. The fragmented feature maps of all branches are defragmented, upscaled and consolidated in a convolutional fully-connected layer that produces a *probability map* for each class. From these probability maps a final pixel classification is generated, the *class membership map*.

into one cohesive feature map array that has the same[1] resolution as the corresponding input pyramid level.

4) Upscaling: After defragmentation, the feature map arrays of lower pyramid levels need to be sampled up and eventually cropped so that they match the resolution of the lowest pyramid level. In this manner, the feature maps of all scales can be concatenated and used as an input to the fully-connected layer.

5) Convolutional Fully-Connected Layer: The fully-connected layer is applied in a convolutional fashion to emulate the patch-based functionality. To achieve this, fully-connected layers have to be transformed into convolution layers with as many input channels as incoming feature maps. This technique is the crucial step to turn a patch-based network into an image-based network and is commonly denoted as *FCN*.

B. Road Segmentation

A CNN, such as outlined above, generates a class membership map, in which every pixel is assigned to one of the trained classes. The class membership map of the preceding step needs to be segmented such that a cohesive road segment can be extracted. Fig. 6 displays a road segmentation generated by the following steps.

Assuming that the biggest connected group of classified road pixels approximates the actual connected group of road pixels, detached road pixel clusters can be neglected. Holes in the connected group of pixels are then filled leveraging a flood-fill algorithm. The algorithm is seeded at the bottom of the image, since that is supposed to be part of the road in most of the cases.

A contour is extracted from the segmented road pixels by using the snake algorithm of [27]. The left and right road border is then determined by splitting the contour in half at the highest central contour point. The road contours are ignored at all border pixels of the camera image. Contour pixels that are adjacent to pixels classified as other road-users, such as `vehicle` pixels, are ignored as well, since road users might conceal parts of the correct road border. Finally, the remaining contour pixels are transformed into the digital map's coordinate system and stored for tracking in the following frames.

[1]Valid convolutions might crop some border pixels.

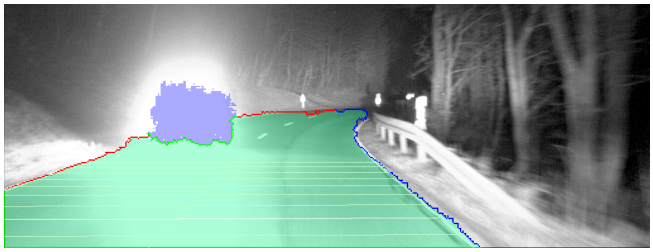

Fig. 6: The green area displays a road segmentation. The blue area depicts a detected vehicle. The red and the blue lines denote the left and right border of the detected road. The green lines (e.g. adjacent to the vehicle area) denote ignored border pixels.

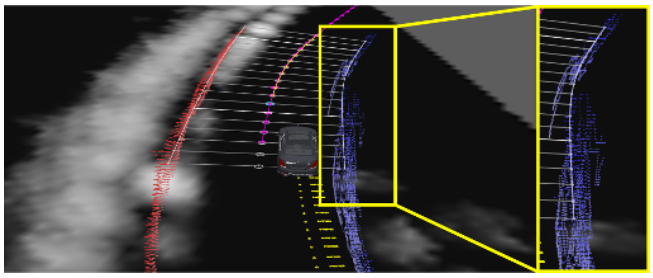

Fig. 7: Road border and center estimates projected to the grid map. The right border shows a big variance and missing values over time. So the shape points of the right border are supported by the center points and the left border.

C. Road Border Shaping

To compute the optical map needed in the following fusion step, the tracked road border estimates need to be fitted into a conclusive road model. All estimates are longitudinally binned, with each bin representing a road border shape point. The values of each bin are analyzed to compute a reliability measure of that particular road border shape point. The bin medians are used as the shape points for fitting a spline, while the interquartile range determines if a shape point is used in the spline computation. Exploiting meta-information contained in the digital map, other track splines, like lane borders, can be interpolated from the left and the right border splines. If preceding processing steps continuously fail to deliver usable measurements for one road border (e.g., in sharp curves) that road border can be extrapolated by the other road border and the other track splines. Fig. 7 displays a spline fitting for unreliable measurements.

Fig. 8: Example images for the five evaluation sequences showing various road and weather conditions.

V. Experiments

The performed experiments are twofold. First, various network topologies were redundantly trained and evaluated with respect to their classification performance (Section V-B). The dataset for training and evaluating the classifiers consists of 7095 full scene labeled images from an NIR camera of rural road sequences at night containing a large variety of weather situations, seasons and landscapes. Second, the optical map of the best performing classifiers were compared to the standard optical map from [3] using the fusion framework from [1] (Section V-C). The evaluation is performed on five nighttime sequences resulting in 13.5 km of driven distance with a ground-truth trajectory taken from a D-GPS sensor and a high accurate IMU. Fig. 8 shows examples images of these sequences.

A. Training of the CNN

Only certain combinations of parameters mentioned in section IV-A are used in the experiments. The influence of the number of pyramid levels (n_l), the deepness of the network while retaining the input patch size (n_c, k_c) and the initial number of filters of the first convolution block (n_f) were evaluated. The topologies are therefore denoted as `topo-`n_l`-`n_c`-`n_f, with a parameter range of $n_l \in [1..5]$, $(n_c, k_c) \in \{(1,7),(3,3)\}$ and $n_f \in \{16, 32\}$. Parameters n_c and k_c have to be chosen such that the input patch size stays the same, which holds for the above defined tuples. The kernel size of the max pooling layers is fixed at 2×2 pixels for all pooling layers in all topology variants. Topology `topo-4-1-32`, for example, has the following parameters: $n_l = 4$, $(n_c, k_c) = (1, 7)$, $n_f = 32$.

All scene-labeled images are split into a set of 6895 images for training and a set of 200 images for evaluation. The original resolution of 512×1024 pixels of the input images is scaled down to 256×512 and padded by 46 (the amount of border pixels lost due to the use of valid convolutions) resulting in 302×558 pixels. To train the topologies, multinomial logistic regression performs a stochastic gradient descent leveraging the backpropagation algorithm [28] with linear learn rate annealing. The target classes consist of the default class `background` and specific classes: `road`, `vehicle`, `sky`, `vru` (vulnerable road users) and `infrastructure`. Training examples are sampled patch-wise and class-balanced for an equal but random distribution of examples per class. The trainable parameters of the networks are initialized by random-sampling a Gaussian distribution. Learn rates for each topology are empirically determined by choosing the best performing learn rate in various mini-trainings. With the selected learn rate full trainings are performed. After completion, the biases of the fully connected layers are adjusted such that the multi-class extension of the *Matthews Correlation Coefficient* (MCC) [29] is optimized. To ensure that equal topologies perform similarly, each topology is trained three times. All trainings were conducted with `cuda-convnet` [13].

B. Scene Labeling Results

Table I shows the classifier performances with regard to several measures and their processing times. Those measures are the MCC [29], the overall accuracy (ACC), the intersection over union (IU) as an average over all classes ($\mathrm{IU}_{\mathrm{global}}$) and specifically for the `road` ($\mathrm{IU}_{\mathrm{road}}$) and `vehicle` class ($\mathrm{IU}_{\mathrm{veh}}$). The IU measure is defined as:

$$\mathrm{IU} = \frac{\mathrm{TP}}{\mathrm{TP} \cup \mathrm{FP} \cup \mathrm{FN}} \tag{1}$$

where TP (true positives) is the amount of correctly classified pixels and FP ∪ FN (false positives and false negatives) the amount of wrongly classified pixels regarding one specific class. The table shows the average result for one multiple trained topology of the individually evaluated classifiers. The timings have been performed on an NVIDIA GTX 970 using CUDA-7.5 and CUDNN-3 including up- and downloading of the data and preallocated memory.

Topology `topo-1-1-16` is comparable to the best performing topology of [9]. According to Table I, this topology achieves the lowest performance. Other topologies are therefore encouraged for road segmentation tasks. Topologies `topo-[3..5]-3-32` perform best regarding most of the measures. This implies that an increase of pyramid levels after level 3 has almost no effect to the best performing topology variant. Fig. 9 shows this effect in a graphical display of the MCC performances dependent on the pyramid levels. Best performing topologies `topo-[3..5]-3-32` take 75.6 to 80.61 ms per frame to be computed, which results in 13.23 to 12.41 frames per second. Switching to topologies `topo-[3..5]-3-16`, which take 28.53 to 32.11 ms, theoretical framerates of 35.05 to 31.14 frames per second can be achieved without a significant drop of classification accuracy. This shows that these topologies can be used in real-time applications.

TABLE I: Performance evaluation of the trained network topologies with respect to various quality measures and their processing time (in milliseconds). Best performing values are marked in bold font.

Name	MCC	ACC	IU_{global}	IU_{road}	IU_{veh}	ms
topo-1-1-16	0.56	0.69	0.40	0.67	0.31	**14.55**
topo-1-1-32	0.60	0.71	0.43	0.70	0.38	45.81
topo-1-3-16	0.61	0.72	0.44	0.70	0.38	19.19
topo-1-3-32	0.66	0.75	0.48	0.75	0.45	53.43
topo-2-1-16	0.66	0.76	0.48	0.78	0.42	19.56
topo-2-1-32	0.70	0.78	0.51	0.82	0.48	59.71
topo-2-3-16	0.71	0.79	0.52	0.82	0.49	25.68
topo-2-3-32	0.72	0.80	0.54	0.83	0.52	69.45
topo-3-1-16	0.71	0.79	0.52	0.84	0.48	21.73
topo-3-1-32	0.73	0.81	0.54	0.86	0.52	64.48
topo-3-3-16	0.75	**0.82**	0.56	0.86	0.55	28.53
topo-3-3-32	**0.76**	**0.83**	**0.58**	**0.87**	0.57	75.60
topo-4-1-16	0.73	0.80	0.53	0.86	0.46	23.31
topo-4-1-32	0.73	0.81	0.53	0.86	0.49	66.67
topo-4-3-16	0.75	**0.82**	0.56	0.86	0.56	30.43
topo-4-3-32	**0.77**	**0.83**	**0.59**	**0.88**	**0.58**	78.69
topo-5-1-16	0.71	0.79	0.50	0.85	0.43	24.91
topo-5-1-32	0.71	0.79	0.51	0.86	0.46	68.56
topo-5-3-16	0.75	**0.82**	0.56	**0.87**	0.56	32.11
topo-5-3-32	**0.77**	**0.83**	0.57	**0.88**	**0.59**	80.61

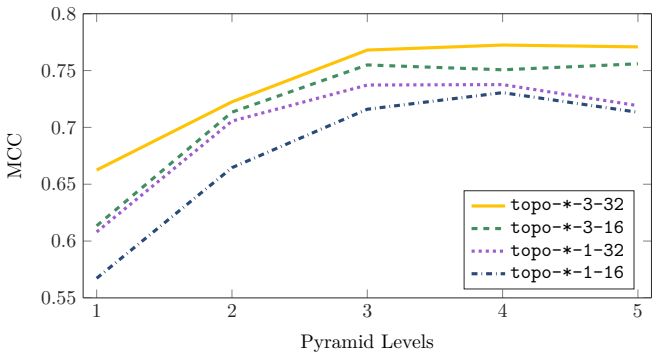

Fig. 9: The MCC of various topology variants in relation to the amount of pyramid levels.

C. Lane Course Prediction Results

In the following, the fusion system performance is evaluated using the best performing classifier for each topology variant in relation to the optical lane recognition from [3] and the system without the optical map. The performance measure is taken from [1]. It compares the deviations of the road course estimations from the ground truth trajectory for different distances to the ego-vehicle. Since [1] have shown that their fusion algorithm benefits primarily short range estimations, only the average performances of the five sequences for short range estimations (0-30 m) are displayed in Table II. The final row displays the failure rates of the lane-based recognition (percentage of frames, where no lane markings are detected). It should be noted that the lane-based recognition measure is solely computed for frames, which contain detected lane markings.

Table II displays that the CNN-based optical map approach performs slightly worse for sequences containing good lane markings ($\mathcal{A}, \mathcal{B}$), but better for sequences with bad weather ($\mathcal{C}, \mathcal{D}$) or bad lane markings ($\mathcal{E}$). Considering the better performing topologies (`topo-[3..5]-*-*`), the CNN-based optical maps show a similar range of performance values ($\sim$28 cm) for sequences $\mathcal{A}$-$\mathcal{C}$ and $\mathcal{E}$. Contrary to that, the lane-based approach shows a greater variance there, ranging from 19 cm (seq. $\mathcal{A}$) to 50 cm (seq. $\mathcal{C}$). This implies that the CNN-based approach is performing more robustly than the lane-based approach, although the peak performance of the latter might not be reached.

TABLE II: Estimation error for the five sequences $\mathcal{A}$-$\mathcal{E}$ (smaller is better). Leaving out the optical map leads to significant deviations (first row) of the error. Lane based estimation [3] performs better on scenes where the lane is clearly visible ($\mathcal{A}$, $\mathcal{B}$) but has a significant failure in all other conditions ($\mathcal{C}$-$\mathcal{E}$). Our approach generates robust estimations for all scenes.

Name	average error [m] short range (0-30 m)				
	$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$	$\mathcal{D}$	$\mathcal{E}$
w/o optical map	1.94	2.74	3.39	2.19	2.40
topo-1-1-16	0.28	0.38	0.31	0.77	0.33
topo-1-1-32	0.27	0.33	0.27	0.64	0.29
topo-1-3-16	0.28	0.33	0.35	0.74	0.35
topo-1-3-32	0.25	0.29	0.26	0.53	**0.27**
topo-2-1-16	0.25	0.30	0.35	0.73	0.31
topo-2-1-32	0.26	0.30	0.33	0.65	**0.28**
topo-2-3-16	0.26	0.31	0.33	0.64	0.31
topo-2-3-32	0.27	0.29	0.30	0.72	0.27
topo-3-1-16	0.25	0.29	0.28	0.65	**0.28**
topo-3-1-32	0.25	0.30	0.26	0.70	0.29
topo-3-3-16	0.26	0.30	0.26	0.61	0.29
topo-3-3-32	0.26	0.31	0.29	0.64	0.28
topo-4-1-16	0.25	0.29	0.34	0.70	0.32
topo-4-1-32	0.25	0.28	0.28	**0.47**	0.29
topo-4-3-16	0.27	0.30	0.32	0.67	0.33
topo-4-3-32	0.26	0.27	**0.23**	0.57	**0.27**
topo-5-1-16	0.27	0.29	0.26	0.55	0.30
topo-5-1-32	0.26	0.29	0.31	0.61	**0.28**
topo-5-3-16	0.25	0.29	0.30	0.58	0.31
topo-5-3-32	0.26	0.28	0.28	0.54	**0.28**
lane-based [3]	**0.19**	**0.25**	0.50	0.89	0.33
failure rate	6 %	8 %	65 %	93 %	23 %

Sequence $\mathcal{D}$ is an exception regarding this robustness. Its performance ranges from 50 cm to 60 cm for the CNN-based and 89 cm for the lane-based approach. This sequence contains severe snowfall and thus, poses a big challenge to sensor processing and detection algorithms. While the CNN-based approach robustly delivers road course estimations for all frames (see Fig. 10 for an example augmented image), the failure rate of the lane marking detection exceeds 90 %. This means that the lane-based approach is practically inapplicable and needs to switch to a mode without optical map. The performance of that mode, though, is always much lower than with an optical map (see first row of Table II).

VI. Conclusion

This paper presented a deep multi-scale convolutional neural network based approach for camera-based road course prediction and localization at night. Various network topologies were trained that reliably detect `road` and `vehicle` pixels, from which an optical map is extracted. Deeper topologies with a higher number of filters per convolution layer perform better, while an increase of pyramid levels after level three does not increase the performance considerably. The extracted optical maps have been successfully fused with a digital map to refine the lateral localization. Compared to a baseline lane-based algorithm, the approach proposed in this paper

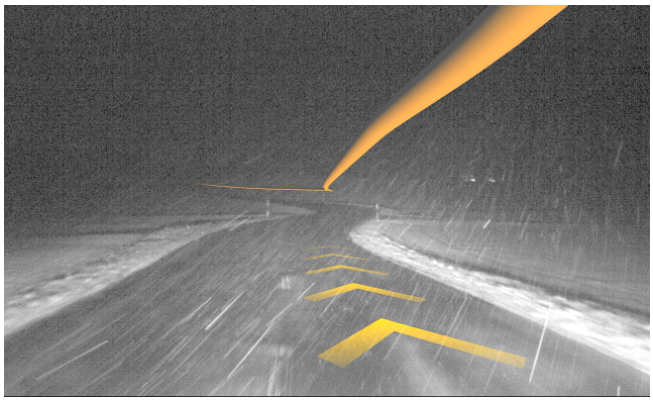

Fig. 10: Augmented reality navigation example for sequence $\mathcal{D}$. Despite the non-existent lane markings and the heavy image distortions due to weather conditions, the approach proposed in this paper is able to reasonably augment the image.

shows a slightly worse performance for optimal road and weather conditions. However, contrary to the baseline, our approach performs consistently well for various weather conditions, even if lane markings are missing. This demonstrates that state-of-the-art performance can be achieved while increasing the robustness and application scope to situations, where traditional lane marking detection is not possible.

ACKNOWLEDGMENT

The authors would like to thank Markus Thom and Oliver Hartmann for their valuable support.

References

[1] F. Schüle, R. Schweiger, and K. Dietmayer, "Augmenting night vision video images with longer distance road course information," in *IEEE Intelligent Vehicles Symposium*, 2013, pp. 1233–1238.

[2] H. Deusch, J. Wiest, S. Reuter, D. Nuss, M. Fritzsche, and K. Dietmayer, "Multi-sensor self-localization based on maximally stable extremal regions," in *IEEE Intelligent Vehicles Symposium*, 2014, pp. 555–560.

[3] R. Risack, P. Klausmann, W. Krüger, and W. Enkelmann, "Robust lane recognition embedded in a real-time driver assistance system," in *IEEE Intelligent Vehicles Symposium*, 1998, pp. 35–40.

[4] A. Bar Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: a survey," *Machine Vision and Applications*, pp. 727–745, 2012.

[5] M. Tsogas, N. Floudas, P. Lytrivis, A. Amditis, and A. Polychronopoulos, "Combined lane and road attributes extraction by fusing data from digital map, laser scanner and camera," *Information Fusion*, pp. 28–36, 2011, special Issue on Intelligent Transportation Systems.

[6] F. Schüle, C. Koch, O. Hartmann, R. Schweiger, and K. Dietmayer, "Probabilistic fusion of rural road course estimations," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, 2013, pp. 1701–1706.

[7] Y. W. Seo and R. R. Rajkumar, "Detection and tracking of boundary of unmarked roads," in *International Conference on Information Fusion*, 2014.

[8] C. Fernández, R. Izquierdo, D. F. Llorca, and M. A. Sotelo, "A comparative analysis of decision trees based classifiers for road detection in urban environments," in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, 2015, pp. 719–724.

[9] J. M. Alvarez, T. Gevers, Y. LeCun, and A. M. Lopez, "Road scene segmentation from a single image," in *Proceedings of the European Conference on Computer Vision*. Springer Berlin Heidelberg, 2012, pp. 376–389.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2015.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2009.

[12] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2016.

[13] 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.

[14] C. Farabet, C. Couprie, L. Najman, and Y. LeCun, "Learning hierarchical features for scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1915–1929, 2013.

[15] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2014, arXiv:1509.01951.

[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 2015.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," Tech. Rep., 2015, arXiv:1512.03385.

[18] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," Tech. Rep., 2015, arXiv:1511.00561.

[19] A. Giusti, D. C. Ciresan, J. Masci, L. M. Gambardella, and J. Schmidhuber, "Fast image scanning with deep max-pooling convolutional neural networks," in *Proceedings of the IEEE International Conference on Image Processing*, 2013, pp. 4034–4038.

[20] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *International Conference on Learning Representations*, 2014, arXiv:1312.6229.

[21] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations*, 2016, arXiv:1511.07122.

[22] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," in *International Conference on Learning Representations*, 2015, arXiv:1412.7062.

[23] F. Tschopp, "Efficient convolutional neural networks for pixelwise classification on heterogeneous hardware systems," Tech. Rep., 2015, arXiv:1509.03371.

[24] M. Thom and F. Gritschneder, "Rapid exact signal scanning with deep multi-scale convolutional neural networks," Tech. Rep., 2016, arXiv:1508.06904.

[25] R. Schubert, E. Richter, and G. Wanielik, "Comparison and evaluation of advanced motion models for vehicle tracking," in *International Conference on Information Fusion*, 2008.

[26] M. Szczot, M. Serfling, O. Löhlein, F. Schüle, M. Konrad, and K. Dietmayer, "Global positioning using a digital map and an imaging radar sensor," in *IEEE Intelligent Vehicles Symposium*, 2010, pp. 406–411.

[27] T. Pavlidis, *Algorithms for Graphics and Image Processing*, 1982, ch. Contour Tracing, pp. 142–148.

[28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, pp. 2278–2324, 1998.

[29] G. Jurman, S. Riccadonna, and C. Furlanello, "A comparison of mcc and cen error measures in multi-class prediction," *PLoS ONE*, 2012.