# High Level Interpretation of Urban Road Maps Fusing Deep Learning-based Pixelwise Scene Segmentation and Digital Navigation Maps

Carlos Fernández[1], Jesús Muñoz-Bulnes[2], David Fernández-Llorca[3], Ignacio Parra[3], Iván García-Daza[3], Rubén Izquierdo[3], and Miguel. Á. Sotelo[3]

[1]Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany. e-mail: carlos.fernandez@kit.edu
[2]ALTRAN Innovation, Madrid, Spain.
[3]Computer Engineering Department, University of Alcalá, Madrid, Spain. e-mail: david.fernandezl@uah.es

***Abstract*—This paper addresses the problem of high level road modeling for urban environments. Current approaches are based on geometric models that fit well to the road shape for narrow roads. However, urban environments are more complex and those models are not suitable for inner city intersections or other urban situations. The approach presented in this paper generates a model based on the information provided by a digital navigation map and a vision-based sensing module. On the one hand, the digital maps include data about the road type (residential, highway, intersection, etc.), road shape, number of lanes and other context information such as vegetation areas, parking slots, railways, etc. On the other hand, the sensing module provides a pixelwise segmentation of the road using a ResNet-101 network with random data augmentation, as well as other hand-crafted features such as curbs, road markings and vegetation. The high level interpretation module is designed to learn the best set of parameters of a function that maps all the available features to the actual parametric model of the urban road, using a weighted F-score as a cost function to be optimized. We show that the presented approach makes the maintenance of digital maps using crowd-sourcing easy, due to the small number of data to send, and adds important context information to traditional road detection systems.**

***Index Terms*—Autonomous Vehicles, Enhanced Digital Maps, Deep Learning, Random Data Augmentation, Urban Road Detection.**

## I. Introduction

The high number of casualties on the road can be explained by many factors. As reported in [1], more than 12.000 lives can be saved per year on European roads if everybody fasten their seat belt, respect speed limits and do not drive under the influence of alcohol. Distraction is another factor since drivers need to keep their attention focused on surrounding traffic continuously, not just for their own safety but for the sake of their passengers and other road users too. Apart from driver distractions, all road elements (vehicles, drivers, infrastructure, etc.) play an important role in the probability of crash or the final outcome. Future scenarios aim to increase the efficiency in several aspects and autonomous driving can help to reduce the number of accidents and also the $CO_2$ emissions.

Self-driving cars require a precise and robust perception of the urban 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 path planning and control algorithms. It has been an exhaustive topic of research in the fields of Advanced Driver Assistance Systems (ADAS) and Autonomous Driving (AD). On the one hand, ADAS have mainly focused on increasing the safety of drivers and road users by warning and assisting the drivers. On the other hand, AD has become a high priority research issue. Most of the major car makers aim at producing fully autonomous vehicles by 2020.

Different levels of autonomy have been demonstrated in highways, urban scenarios and cooperative environments [2]–[5]. However, in all cases a high definition 2D, or even 3D, map is required. Enhanced maps integrate precise information of the environment such as road shape, lane markings, curbs, intersections, buildings, etc. The main drawbacks of this type of maps are their size ($\sim 2GB/km$), the complexity of measurements integration and their maintenance.

On the other hand, the advent of Deep Learning techniques, namely Convolutional Neural Networks (CNNs), has involved 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 architectures, such as AlexNet, VGG-16 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 and complex road shapes. On top of that, the sensor costs of these road detection approaches is considerably low since it only involves the use of digital cameras. Other approaches based on high cost LIDARs ($\sim 75K\$$) are not affordable for the car industry.

In this paper we present a high level interpretation approach capable of estimating a parametric model of the road representing the real scenario appearing in front of the vehicle even in complex and cluttered environments. Based on our previous works [6], a hybrid vision-map method is proposed. However, instead of using the best estimate of the road shape from an enhanced digital map as a feature of a hand-crafted road segmentation classifier [6], we propose a Deep Learning framework using ResNet-101 network with a fully convolutional architecture and multiple upscaling steps for image interpolation, to obtain an accurate estimation of the road, outperforming our previous results. We demonstrate 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, image cropping, mirroring, blur, noise, distortions, and color changes. In addition, we show 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. This pixelwise segmentation process is combined with previously designed hand-crafted features extraction methods ( [6]–[9]) to provide a multi-class segmentation of the road and nearby regions. Finally, a high level interpretation module is trained to learn the best set of parameters of a mapping function that is capable of transforming the multi-class segmentation output to the actual parametric model of the urban road. The presented approach makes the maintenance of digital maps using crowd-sourcing easy, due to the small number of data to send, and adds important context information to traditional road detection systems.

## II. Related Work

### A. Road segmentation

Different vision-based methodologies to detect the road can be distinguished. Some of them are based on road appearance learning, where the main hand-crafted features are texture and color information. The second approach focuses on road limits detection, assuming that the space between limits is the road surface. Finally, model-based approaches try to extract a compact high level representation of the road. In order to have a better overview of the different sensing technologies, road appearance and limits modeling techniques, geometrical models and features integration we refer to [6].

On the other hand, it has been proved that CNNs can improve state-of-the-art results on image classification [10]–[13], and they have also been successfully applied to object detection [14]–[16] as well as to monocular color image segmentation. There are several approaches to obtain a CNN-based pixelwise classification of an input image. First, the widely-adopted fully convolutional networks (FCN) [17] adapt classifier networks, such as AlexNet [10] or VGG [11], to the segmentation task by replacing fully-connected layers with convolutional ones and using a progressive interpolation approach. Other approaches follow this trend using other base networks, such as the ResNet [12]. In [18] dilated convolutions are introduced to reduce the downsampling performed by the convolutional layers, removing the need of the progressive interpolation stages. These kind of dilated convolutions are further explored in [19] along with another upsampling method called *dense upsampling convolution*.

A more complex approach such as DeconvNet [20] learns a deep deconvolutional network on top of the convolutional one. SegNet [21] uses an encoder-decoder architecture. PSPNet [22] exploits pyramidal pooling to introduce global contextual priors in a dilated fully convolutional network. FRRN [23] is based on a novel architecture that keeps a stream with full-resolution features.

Other models do not directly produce a full image classification but work patchwise instead. In [24] the inputs to the classifier layers are enhanced with spatial information of the patch, and in [25] they use a deconvolution scheme. However, the current trend is to use fully convolutional, end-to-end models, from images to pixelwise classification masks [16].

In addition, there are specialized methods for road detection. One example is [26], 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 [27], MultiNet system is presented, which performs simultaneous street classification, vehicle detection and road segmentation, all with the same CNN encoder and three different decoders.

In order to learn the huge number of parameters of CNNs architectures, thousands of images labeled with the categories to learn are required. There are different datasets available for autonomous vehicles. One of the most popular is KITTI benchmark [28]. This dataset has labeled 289 images with road and non road labels. However, CityScapes dataset [29] is increasing the number of submissions due to the number of labeled categories (30) and the number of images (25K). The results of CNNs in the field of road detection have outperformed all the algorithms that use other approaches.

### B. Map road model fitting

The simplest geometric model used for road boundaries are straight lines. Due to the pin hole camera model, straight parallel lines converge in a vanishing point. This principle is exploited in the state of the art to model the road using an edge descriptor extracted from the image [30]. More complex models are used to model curved roads, such as parabolic curves [31], clothoids [32], B-splines [33] or active contours (snakes) [34]. These parametric models improve the noisy bottom-up detections due to their constraints of width and curvature. However, urban environments are more difficult to model due to the presence of intersections and the variety of curvatures and width changes.

Non parametric models are less common because they demand only that the line should be continuous. It provokes the model to be less robust than parametric models but more flexible to adapt to the irregular shapes present in urban environments [35] or rural paths [36].

Map-based models 1 are dominated by high definition maps. They have demonstrated to be a robust way to navigate [2]–[4] and they are usually built integrating several measurements of a multi beam LIDAR [37], [38] or multiple single beam LIDARs [39]. The drawback is their size ($\sim 2GB/km$), which is difficult to manage in a long trip or in a city. The ultimate goal is to drive everywhere with full functionality, but there are two main points of view on how to get to it. The first approach tries to drive in some places with full functionality using 3D detailed map ($\sim GB/km$) and low resolution sensing. On the other side, the goal is to drive everywhere with partial functionality using low resolution maps and high resolution sensing. The problem of the first approach is the scalability of the map and the updates. The problem of the second approach is to get stronger artificial intelligence. The ultimate goal is to get cognitive perception as humans do. However without that level of artificial intelligence, a higher resolution map is required to compensate that weakness.

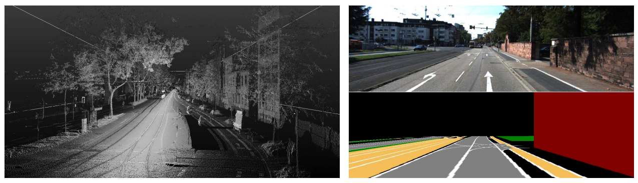

(a) High definition scene reconstruction after the integration of several measurements of a multi beam LIDAR. (b) High level output provided by our approach.

Fig. 1. Map-based models: comparison between heavy 3D cloud-based models and light high-level interpretation.

## III. CNN-Based Road Classifier

On the one hand, and given the generalized use of CNNs on the road detection problem, this paper proposes a road detector based on the ResNet network model [12] adapting its last stages to the fully convolutional architecture [17]. First, a ResNet-50 model already trained on the ImageNet dataset (1000 labels at image level) has been used. In contrast, our system is designed on the KITTI road detection dataset [28], which only defines 2 labels (road/non-road) at pixel level. Accordingly, the original ResNet-50 architecture has to be transformed into a fully convolutional network to admit an input of an arbitrary size and to produce an output of the same size with pixelwise segmentation. In order to do that, the last inner-product fully-connected layer (1000 outputs) is replaced with a new convolutional layer (two outputs) that will be learned from scratch. In addition, due to the fact that ResNet has an overall downsampling factor of 32, some upsampling stages are needed. As can be observed in Figure 2 the upsampling is performed in three interpolation stages:

1) UPSCORE 32: the main output scores are upsampled by a factor of 2. The output from the previous block CONV 4 is added, since both scores have the same accumulated downsampling factor (16).
2) UPSCORE 16: the previous result is upsampled by a factor of 2. The output from the previous block CONV 3 (downsampling factor of 8) is then added.
3) UPSCORE 8: the result is upsampled by a factor of 8 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. Finally, 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. The final output of our architecture consists of two channels that represent the probability maps for background and road respectively, obtained from the final SOFTMAX layer.

### A. Data Augmentation

One of the main weaknesses of CNNs is their dependence on the training data. With data augmentation a better generalization can be achieved and different road conditions can be simulated, increasing the robustness of the network against illumination, color or texture changes, or variations in the orientation of the cameras. We adopt an on-line augmentation approach where modifications are performed at random each time. This way, the CNN never sees the same augmented image twice and this virtually infinite dataset does not require extra storage space on disk.

It can be distinguished between geometric transformations and pixel-value modifications. It is also possible to apply several augmentations on the same image, or to apply different augmentations for each label (road or background) or to patches.

*1) Geometric Transformations:* These transformations are applied to both the image and the ground truth mask. Zero padding is added when needed to keep the original image size and the padding pixels are assigned to the ignore label" of the classifier. The applied transformations include:

- Random affine transformations: translations, rotations, scalings and shearings are performed in order to change the positions of the points, while keeping lines parallel. Initial points are fixed to form a triangle with its bottom side aligned to the bottom of the image. This has been tested to provide good transformations (similar to those that could happen when driving a car). Final points are drawn from a 2D Normal distribution centered in each initial point. Although these transformations could be applied independently, better results are obtained with combined affine transformations due to the high variability.
- Mirroring: apart from the affine transformations, horizontal flipping is performed independently to easily double the size of the training set.
- Cropping and scaling: the original image is cropped and scaled 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 are selected empirically on road limits. The final positions are calculated adding Gaussian noise to the original ones with two restrictions. First, the shift of top points is the opposite of that of the bottom ones. Second, top points should not cross each other to prevent reflected images.

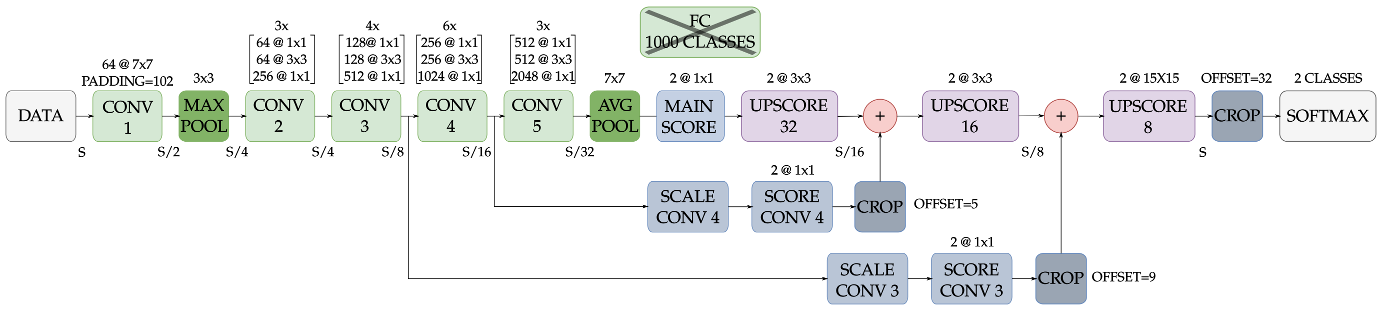


Fig. 2. High-level schematic of the CNN-based road detector, implementing the FCN-8s architecture [17] on a ResNet-50

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

- Noise: random addition of Gaussian, speckle, salt & pepper noise, generation of an image with signal-dependent Poisson noise. Gaussian and speckle noises are additive: the first one is defined by a given standard deviation, and the second one is generated by a uniform distribution with a given variance and a magnitude proportional to the image itself. Salt & pepper noise is generated by setting to white or black random pixels with a given (low) probability. Regarding Poisson noise, a new image is generated drawing each pixel from a Poisson distribution with lambda (mean) proportional to the image value in that point.
- Blur: the filters are applied independently to the image, creating a blur effect. The selected filters are: Gaussian, diagonal motion (left-to-right or right-to-left, at random), box, median and bilateral filtering.
- 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 [40]. 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 [10] for performing slight alterations in RGB space. It is based on a previous PCA analysis of RGB values throughout the training subset and 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 CNNs, 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 [41]. Although it is not possible to change the original pre-trained ResNet-50 structure, we can add PReLUs to the new score layers. Training alternatives involve trying different learning rates and learning rate policies, such as decreasing the learning rate when the training stalls in previous trials, or doing a *warmup* stage [12] at a reduced learning rate until error goes under (20%). Other alternative is to have a higher learning rate for score layers, which are learned from scratch, and a lower rate for inherited layers. Moreover, in [17], 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 [19] 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 training approach uses images in perspective view and obtains detections in this space. However, since KITTI benchmark evaluates its results with the F1-measure in Bird's Eye View (BEV) [42], we also train the network model directly in BEV. In this case, a less aggressive data augmentation strategy is used since geometric transformations in BEV creates important distortions.

## D. Deeper models

A ResNet-101 [12] model has been adapted using the same procedure applied to the ResNet-50, to test a deeper model in this problem. On the one hand, this model has an increased learning capacity. On the other hand, the risk of overfitting becomes more relevant.

## E. Upsampling variants

Apart from the schematics presented in Figure 2, 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 [18] and its improved version [19], 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 would 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 3).

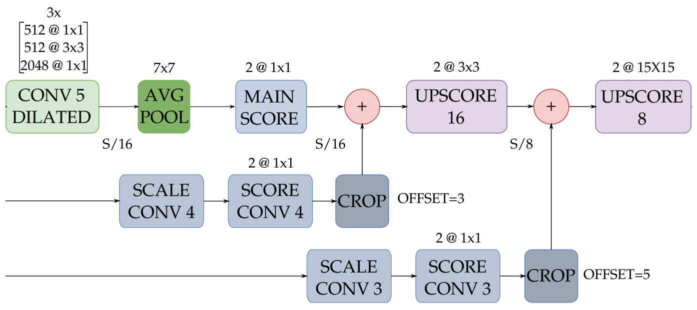


Fig. 3. Detail of the scoring stage with dilated convolution in CONV 5. It removes the necessity of using the UPSCORE 32 layer.

## IV. High Level Interpretation Module

### A. Digital Navigation Maps

There are two main type of maps. The first ones are navigation maps, which provide information about the steps to reach our destination. The second ones are high definition maps, which provide 3D information of the environment with centimeter precision. Most of the autonomous navigation vehicles are based on these types of maps [2]. In contrast to that, our approach is closer to the human way of drive. Human drivers do not need high definition maps. They drive using visual perception and local navigation methods. The only information they need are the indications and steps on the navigation map to reach the destination.

The digital navigation maps used for our approach are Open Street Map. These collaborative maps are created by a large community around the world and all the information stored in the map is editable and it is freely accessible. The map consists of a list of streets called ways. Every way is composed of a list of nodes with a location and its relations with the other nodes and ways. Thanks to the location and relation between the nodes, the shape of the current street and its surroundings can be estimated. In addition, digital maps include the number of lanes, road type, etc.

### B. Road map modeling

The features extracted from the cameras (including the features obtained in [6]) are fused with digital navigation maps to obtain a high level interpretation of the urban scene. The map includes relevant information about the presence of railways, parking areas, buildings or intersections, which are key points for a correct scene interpretation. Most of the elements in the map model are fixed (buildings, gardens, etc.). However, the road width should be adapted depending on the road type (residential, highway, intersection, etc.). Our proposed model has 6 degrees of freedom: number on lanes, lane width ($w$), lateral offset ($y$), longitudinal offset ($x$), angular offset ($a$) and curvature radius in intersections ($r$), see Figure 4.

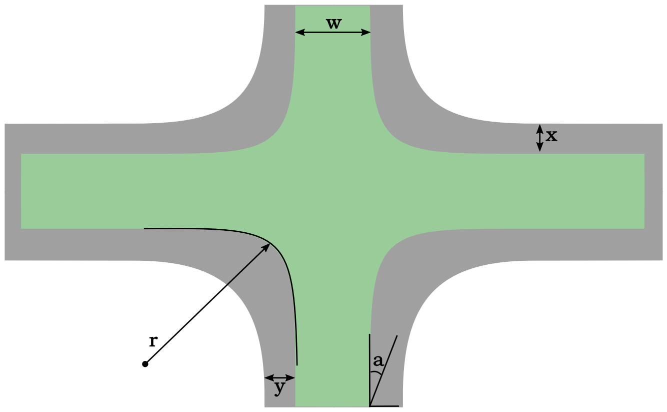


Fig. 4. Parameters to adjust in the proposed model of the urban road. The real scene is painted in grey and the model to adjust is shown in green.

It is assumed that the number of lanes is determined by the map. Nowadays, most of the maps indicate the number of lanes and the lane you should drive to reach your destination. The other parameters are evaluated in two steps, the first one for a coarse adjustment and the second one for a fine adaptation. Table I shows the range of every parameter, obtaining more than $\sim 800K$ and $\sim 15K$ combinations in the coarse and fine adjustment respectively. The integration of the vehicle ego-motion with the previous models along the time creates a prior knowledge where the model should be. This prior knowledge removes the coarse adjustment and the fine adjustment could be extended. The selected option to reduce the number of combinations divides the process in three steps: the first step combines lane width and lateral displacement, the second one adjusts the angular offset and finally the third step combines the longitudinal offset and the curvature radius. This process reduces the number of combinations from more than $\sim 815K$ to only 512.

The metric to evaluate the best adjustment is calculated using equations 1 and 2, where the precision and recall are estimated by matching the model and the sensing of the environment. The matching is evaluated in 4 different groups ($G$). The first one compares vegetation areas (garden, grass, forest) and obstacles (barriers, buildings, walls) [6]. The second one is the road provided by the CNN model. The third group is composed of curbs and road markings [6]. The last one only compares curbs [6] in order to reinforce the correct

TABLE I
MODEL PARAMETER RANGE OF VALUES

| | COARSE | | | FINE | |
|---|---|---|---|---|---|
| **PARAM** | START | RANGE | STEP | RANGE | STEP |
| Lane Width | 3.20 m | ±0.40 m | 0.20 m | ±0.20 m | 0.10 m |
| Lateral Displacement | 0.0 m | ±8.00 m | 0.50 m | ±0.50 m | 0.10 m |
| Angular Offset | 0.0° | ±5.00° | 0.25° | ±0.20° | 0.10° |
| Longitudinal Offset | 0.0 m | ±8.00 m | 0.50 m | ±0.50 m | 0.10 m |
| Curvature Radius | 7.00 m | ±3.00 m | 1.0 m | ±1.0 m | 0.50 m |

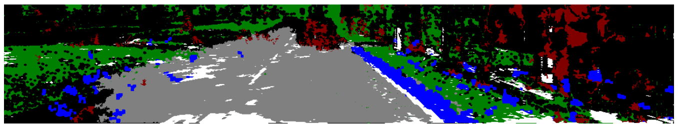
adjustment of the road boundaries. The weights of every group in the final score $(S)$ are set after a training stage to optimize the correct adjustment. The combination of the parameters with the highest score $(S)$ generates a map-based model which is the output of the algorithm. This map is then provided at the same space of the images obtained from the cameras.

$$F_{score} = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{1}$$

$$S = \sum_i w_i \cdot F_{score}(G_i) \tag{2}$$

As an example, we depict in Figure 5 the results of the sensing stage. The map model is adapted to the current scenario taking into account every detected feature and their correspondence in the map model.

(a) Input image.


(b) Sensing result with road in grey, road markings in white, curbs in blue, obstacles in red and vegetation areas in green.

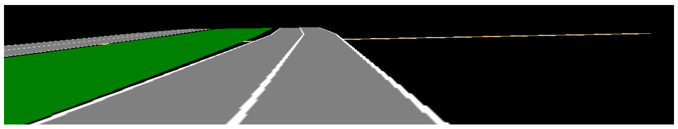

(c) Road model generated from the sensing results and the information from the digital navigation map. Road is painted in grey, lanes are delimited in white, vegetation and obstacles are painted in green and red respectively.

Fig. 5. Results of the high level interpretation module.

Note that the resulting model is based on static elements such as the road, vegetation areas, buildings, etc. This is a clear advantage in order to maintain an updated and enhanced version of the high-level structure of the digital navigation maps. When performing autonomous navigation, the drivable area is directly provided by the CNN-based road classifier which will not include dynamic objects such as pedestrians, bicycles and cars. Previous hand-crafted [43]–[46] and more recent deep learning-based approaches [16] can be here adopted to detect dynamic obstacles.

## V. RESULTS

### A. Experimental Setup

Our CNN-based road segmentation model was trained on the KITTI dataset, which is composed of 289 images manually labeled with two classes: road and non-road. 50% of the images were used for training the net, and the remaining 50% are kept aside for validation.

More specifically, the ResNet-50 model previously trained on ImageNet was used for weight initialization, and then the full net is fine-tuned on the road detection task. This is performed with stochastic gradient descent at a constant learning rate of $5 \cdot 10^{-5}$ (except for the bilinear filters, which are fixed), weight decay of $5 \cdot 10^{-4}$, one image per iteration and high (0.99) momentum. This scheme is referred as *heavy learning* in [17]. The training is run for 24K iterations, with validation checkpoints every 4K iterations.

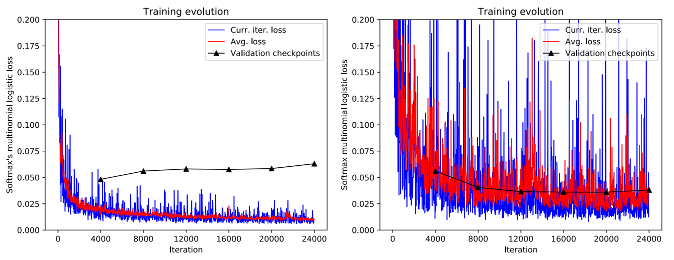
The Caffe framework [47] was used for the network prototype definition and the control of training and testing processes. In addition, a Python input data layer, adapted for KITTI, is used for loading images and labels into the net: each training image is randomly picked along with its corresponding label, and some minimal preprocessing must be done. The ImageNet per-channel pixel mean is subtracted, and the label images are converted into a 1 $\times$ height $\times$ width integer array of label indexes to be compatible with the loss function. As stated before, instead of passing the original image to the network, some data augmentation operations were 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 2-3 hours to complete the standard training on a single Titan X GPU.

The parameters of the high-level interpretation module were estimated using the same images used to train the ResNet segmentation module from KITTI dataset. This module was implemented in C/C++. Training stage takes less than 1 minute and on-line estimation is performed in real time ($<$ 5ms).

### B. Road Segmentation Results

In this subsection we present the results obtained from the different variations on the network previously mentioned. As proposed in [42], 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.

*1) Data augmentation:* We can see, in the training and validation losses curves (Figure 6), that the use of data augmentation prevents the network from overfitting, since the gap between training and validation losses disappears: training losses rise slightly whereas validation losses decrease. We have observed that geometric transformations introduce higher variability than pixel value changes, and using both kinds we obtain the best results. We transform the full image with a single random operation each time. This way, we get a high variability, as we note from the width of the losses curve in 6b, but with an acceptable level of noise.



(a) Training without data augmentation (b) Training with data augmentation

Fig. 6. Comparison of the training and validation losses without data augmentation and with data augmentation.

With this method we can achieve an improvement of approximately 1% in F-measure when training in perspective space (from 94.59% to 95.76%), and 2% when training in Bird's Eye View, which will be discussed later on.

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 7 demonstrates that data augmentation makes the model more robust against illumination, texture, perspective and orientation changes.

*2) Network Components and Training Variants:* The upscore described in Figure 2 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. Neither finetuning the bilinear filters (slower convergence and we get the same kernels in the end) nor learning them from scratch (smoother convergence and we get different interpolation kernels that use information from both classes' scores, and scattered road limits) improve our previous performance 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 ($5 \cdot 10^{-5}$).

Since we are performing a fine-tuning with few iterations, changing the learning rate does not seem to have positive effects: decreasing policies can yield a monotonically decreasing validation losses curve, but the final losses and the F-measure are not better, and if the decrease is too abrupt, the training will become unstable, probably due to the high momentum; with the *warmup* scheme, there are not improvements either. It also appears to be better to let the whole net adapt to the new task of per-pixel road detection, instead of using a reduced ($0.1\times$) learning rate for inherited layers. Finally, accumulated learning policies lead to worse results and are also much slower (proportionally to the batch size). The polynomially decreasing learning rate variant from [19] is better but still not superior to *heavy learning* in our trials.

*3) 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%). Analyzing 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.

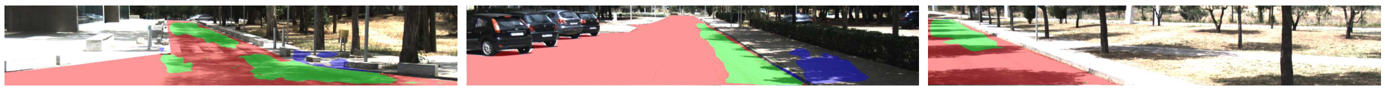
*4) Deeper Models:* The training tests in perspective space over deeper models establish that the ResNet-101 achieves slightly better results over ResNet-50, which are obtained consistently with fewer iterations. As a drawback, the training takes slightly more time to complete than with ResNet-50 and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

TABLE II
COMPARATIVE RESULTS OF DEEPER MODELS PERFORMANCE.

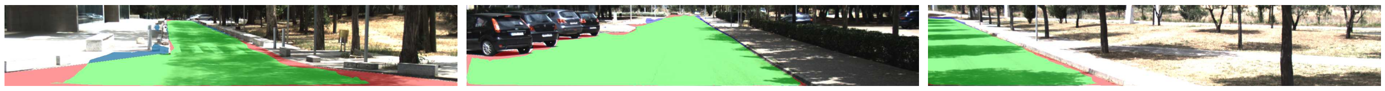
| Model | F-measure | Training Time | Iterations | Memory |
|---|---|---|---|---|
| ResNet-50 | 95.76 | 2h00 | 24K | 7GB |
| ResNet-101 | 95.88 | 2h30 | 20K | 10GB |

In the experiments evaluated in BEV, it is observed overfitting 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 (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.

*5) 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 both in perspective and 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 also similar results. In particular, the



(a) Training without data augmentation



(b) Training with data augmentation

Fig. 7. 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.

method from [18] 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.

*6) Global Road Segmentation Results:* Table III 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 2.

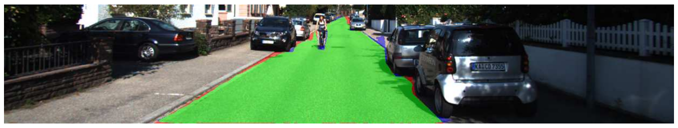
TABLE III
QUANTITATIVE RESULTS IN F-MEASURE ON KITTI DATASET FOR THE MAIN NETWORK VARIANTS EVALUATED.

| 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%** |

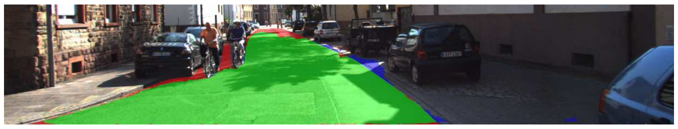
The two best-performing methods, namely the ResNet-101 with data augmentation and four-step interpolation are trained with perspective and BEV images. Small obstacles such as pedestrians, cyclists or cars are well differentiated from the road areas (Figure 8a), although two cyclists riding together are considered as a single obstacle (Figure 8b) 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 sometimes leave FN gaps (Figure 9a and Figure 10a on top-right corner), as well as FP patches outside road limits (Figure 9d) that could be filtered with some post-processing methods. However, the model trained in BEV seems to be better delimiting road limits in the same image (Figure 10b) because in this representation they are straighter.

It can be seen that the BEV-trained model is better at detecting irregular road limits (Figures 10c and 10d) than the perspective-trained one (Figures 9c and 9d). However, the main problem of the BEV-approach is that in some particular cases, the resulting image is so distorted and the net confuses buildings with the continuation of the road (Figure 10e). This would be very unlikely to happen if the image was analyzed in perspective space.



(a) Single cyclist well segmentated



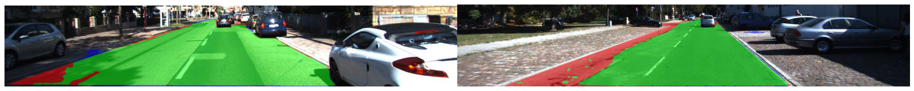
(b) Group of cyclists considered a single obstacle

Fig. 8. Results from the perspective-trained model in a scene with cyclists.

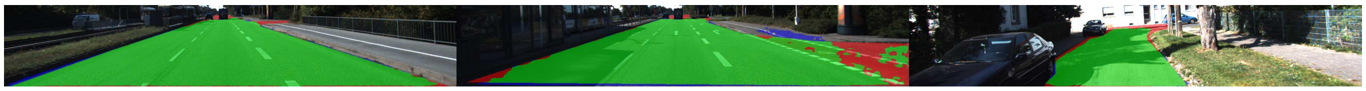
## C. High-level Interpretation Results

Considering the hand-crafted features of our scene interpretation module, we remark the following statements. The curb detection method was analyzed in detail in [7]. The algorithm was compared using different sources for the 3D cloud data (LIDAR and stereo), obtaining a lateral RMSE of 12cm in a range from 6m to 20m. The proposed algorithm can deal with curbs of different curvature and heights, from as low as 3cm, in a range up to 20m whenever that curbs are connected in the curvature image. Finally, the use of fixed or empirical thresholds is avoided given that the proposed function is adapted automatically for different road scenes depending on the predominant curvature value. The boosting classifier was described and tested in [6], [8]. The weight of each feature in the final road classification reveals that 3D features (Y and Z coordinates) and its 2D representation (column and row) are the most discriminant features. However some of the other 2D features are still important such as the grey value of (HSV) or the vegetation. Some of the road misclassifications are produced in sidewalks, where the only difference between the road and sidewalk is a small curb. Furthermore, in challenging urban scenarios the limit of a drivable area is difficult to distinguish from the non-drivable area, such as a cyclist lane. In some cases the limit is just a road marking or a variation in the pavement texture. It is remarkable that road markings have a low weight in the final response.

According to the results obtained in [6], [8], the weight of curbs and road markings in the road classification is

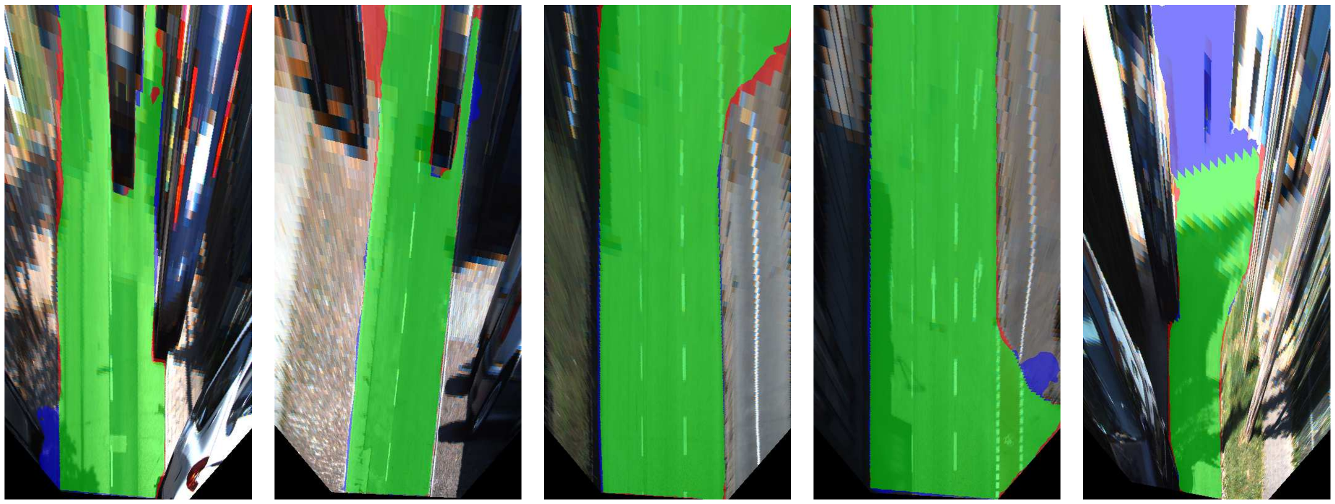


(a) FN gaps inside road area (b) FN areas due to incorrect road limits detection



(c) Missdetection of a distant intersection (d) FN of an incoming lane (e) Correct detection of the building

Fig. 9. 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. 10. Results from the BEV-trained model

very small because that type of features describe road limits instead of the road surface. The proposed algorithm follows a human reasoning. Therefore road markings and curbs should have a relevant role in the interpretation of the environment. That is the reason to match the map model with 4 different groups of features: vegetation + obstacles, road, curbs + road markings, curbs. After a weight training stage for each group, the qualitative results demonstrate the effectiveness of the proposed method to infer complex urban environments.

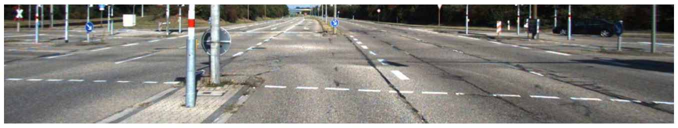
TABLE IV
QUANTITATIVE RESULTS ON F-SCORE IN KITTI DATASET.

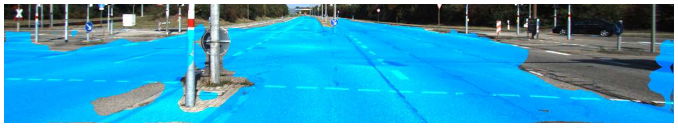
| | Image plane | | | | Bird Eye View | | | |
|---|---|---|---|---|---|---|---|---|
| **Classifier** | **UM** | **UMM** | **UU** | **All** | **UM** | **UMM** | **UU** | **All** |
| Boosting | 85.24 | 93.09 | 82.08 | 87.06 | 85.15 | 89.37 | 62.60 | 79.05 |
| Map Model | 89.35 | 88.56 | 90.20 | 89.37 | 85.17 | 84.76 | 86.66 | 85.53 |
| CNN | 96.64 | 95.36 | 94.89 | 95.50 | 95.09 | 94.94 | 93.39 | 94.59 |

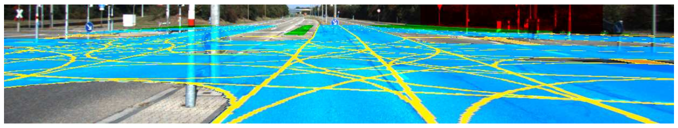
When evaluating our previous Boosting-based classifier [6] with the new CNN-based approach, we can state that our CNN model with data augmentation clearly outperforms the classic hand-crafted features + classifier approach. As can be observed in Table IV, where the F-measure is obtained using the training set of the KITTI dataset (50%/50% training/test) both in perspective view and BEV, the CNN-based road segmentation provides an overall F-measure a 8.44% and a 15.54% better than the value provided by our previous approach in the perspective image and BEV respectively.

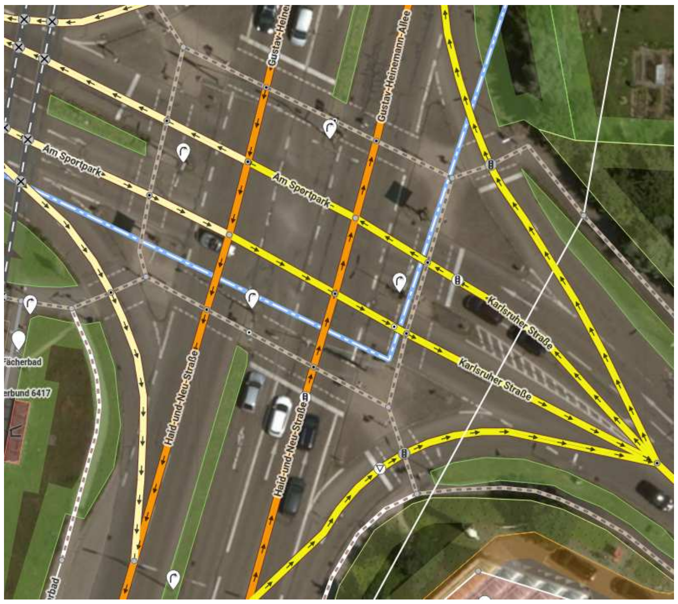
We have also obtained the F-measure of the high-level interpretation module (see Table IV) yielding overall values of 89.37% and 85.53% in perspective view and BEV respectively (note that we have not been able to obtain results from the KITTI test dataset since GPS positions are not available). These values are a 6.13% and 9.06% worse than the CNN classifier in the image plane and BEV respectively. This is obviously an unfair comparison since the high-level road estimation module is not a pixelwise approach trained with the ground truth location of the road. It does not include dynamic obstacles, and in some cases, it provides some regions of the road that are not even labeled in the ground truth (opposite lanes, or not visible intersections). In general, pixelwise classifiers outperform model-based approaches because there is not any model that fits as close to the ground truth as the pixelwise classifiers. Figure 11 shows an example of a very complex intersection where the pixelwise classifier outperforms the map model due to its inner architecture. The ways are usually centered with respect to the center of the real road, but this scene has ways that have different number of lanes before and after the intersection. In addition, the lanes for left turning are overlapped with each other, which is impossible to adjust to the map architecture. However, the map model includes relevant information to be supplied to any navigation module of an autonomous navigation system.

In some other scenarios, the map-based model improves pixelwise classifiers. Figure 12 shows an example of an urban scene where the CNN road classifier without data augmentation has many FN on the left and right boundaries of the road. The road model fills the missing pixels close to the road boundaries and it obtains a shape that fits better to the real scenario.


(a) Input image.


(b) CNN classifier.

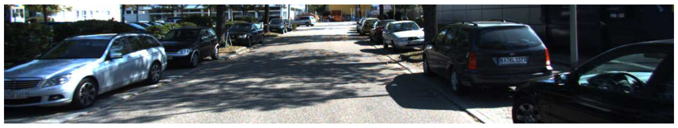

(c) Map Model.

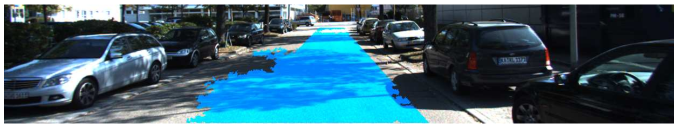


(d) Open Street Map diagram with aerial imagery layer.

Fig. 11. Example of a very complex intersection where the map-based model is not well fitted to the road limits. However, it provides richer high-level information to be used in autonomous navigation systems.

The qualitative results show the added value of the interpretative approach presented in this paper. Figure 13 shows an urban street with one lane for each direction separated by a fence. Furthermore, on the right side of each lane, there are slots for parking and buildings. Even with a coarse detection


(a) Input image.


(b) CNN segmentation without data augmentation.

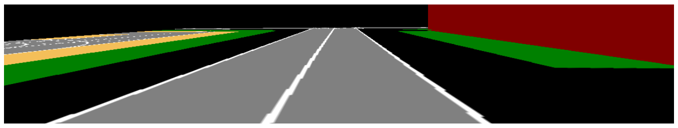
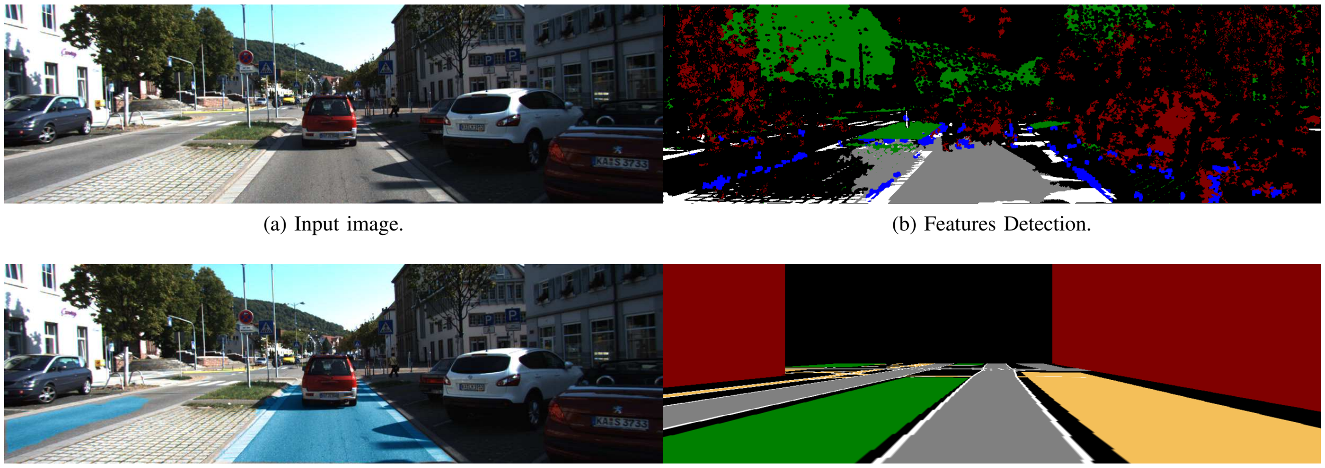

(c) Map Model.

Fig. 12. Example of a scene where the pixelwise classifier has a large number of false negatives and the use of a map-based model improves the result.

of the road (Figure 13b) or neither detect the road, the map model fits well to the scene and add high level information to the system. The use of different features for the map matching, increase the robustness of the method because the absence of one of them does not make the system to fail. The scene represented in Figure 14 highlights the use of high level information extracted from the map. The use of map information reduces the possibilities to infer the cycle lane as a road lane and because of that, the railways and the cycle lane are correctly labeled. Finally, Figure 15 shows an urban street where the map model fits well to the environment but the presence of parked vehicles creates a high number of FP. Dynamic obstacles, such as cyclist and vehicles are effectively removed and not considered as drivable by the CNN-based road segmentation system. The high-interpretation module maintains the actual structure of the road besides the dynamic obstacles.

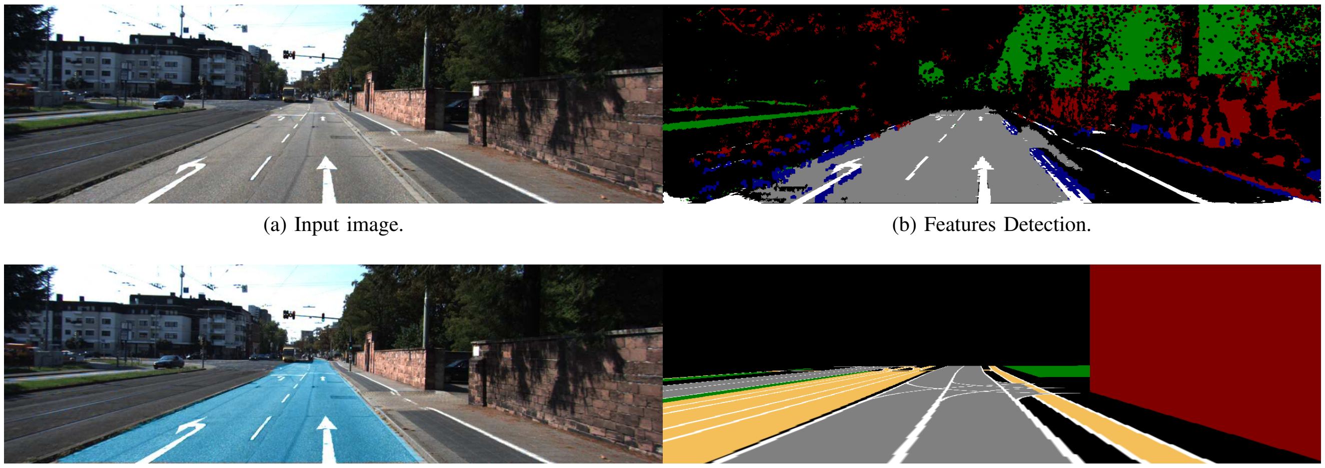
## VI. Conclusions & Future Work

A novel high-level interpretation approach that integrates pixelwise road segmentation, a set of hand-crafted features that describe road limits (vegetation, road markings and curbs) and enhanced data provided by digital navigation maps was presented. A ResNet-50 network model with a fully convolutional architecture and three interpolation steps, which are finetuned in perspective KITTI images, was used to obtain an accurate detection of the road, which represents the drivable area. Several variations were introduced to improve the training such as data augmentation, training in BEV space, training parameters tuning, using deeper models and other upsampling architectures. Data augmentation offers a significant improvement between of 2% in F-measure. The use of a ResNet-101 model with a four-step upsampling scheme, trained directly in BEV with data augmentation, improves the results up to a 96.31% in the validation subset. The proposed approach clearly outperforms our previous Boosting-based road detection approach [6].


(a) Input image. (b) Features Detection.

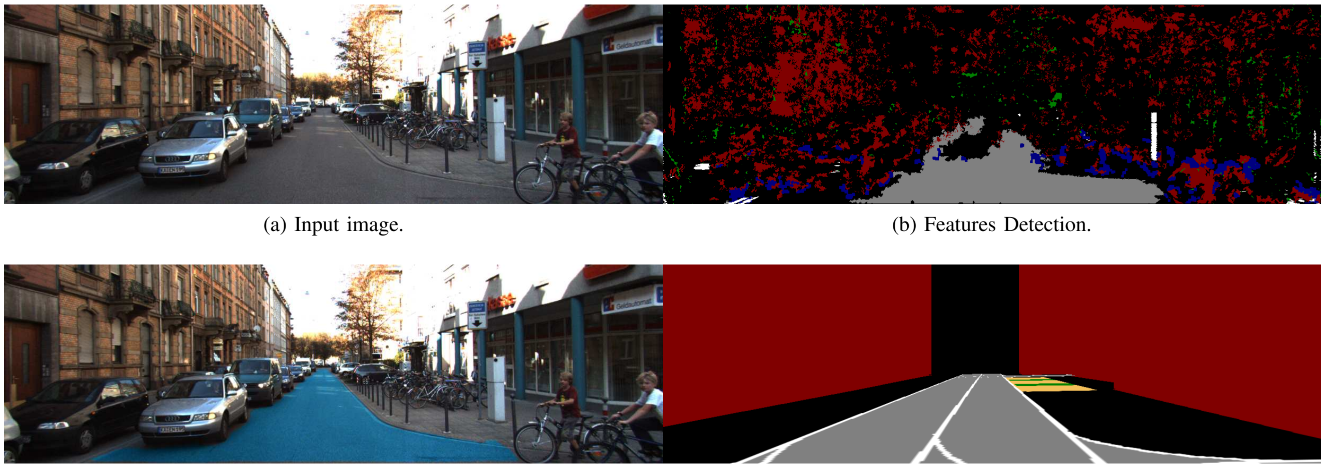
(c) CNN road classifier. (d) Map Model.

Fig. 13. Results in a scene with parking slots on the right and vegetation on the left. Even with a coarse detection of the road (Figure 13b), the map model fits well to the scene and add high level information to the system.


(a) Input image. (b) Features Detection.

(c) CNN road classifier. (d) Map Model.

Fig. 14. The use of a map model increases the robustness of a road classifier because the other boundary features compensate some road missclassifications.


(a) Input image. (b) Features Detection.

(c) CNN road classifier. (d) Map Model.

Fig. 15. Dynamic obstacles (cyclist and vehicles) are effectively not considered as drivable by the CNN-based road segmentation system. The high-interpretation module maintains the actual structure of the road besides the dynamic obstacles.

The presented approach can be applied to update digital navigation maps using floating vehicles equipped with a pair of stereo cameras. In parallel, an accurate estimation of the drivable area is supplied by our CNN-based road segmentation module.

Future works will be devoted to obtain smoother road detection results by adding a post-processing layer into the system. In addition, due to the new advances in semantic segmentation [16], the data provided by digital navigation maps, and the output given by the high-level interpretation module will be enriched with new variables, including traffic lights, traffic signs, or even urban furniture (benches, bins, bus stops, etc.).

## Acknowledgment

This work was funded by Research Grants SEGVAUTO S2013/MIT-2713 (CAM), DPI2017-90035-R (Spanish Min. of Economy), BRAVE Project (H2020, Contract No. 723021) and UAH (CCGP2017-EXP/055). In addition, this project has received funding from the Electronic Component Systems for European Leadership Joint Undertaking under grant agreement No. 737469 (AutoDrive Project). This Joint Undertaking receives support from the European Unions Horizon 2020 research and innovation programme and Germany, Austria, Spain, Italy, Latvia, Belgium, Netherlands, Sweden, Finland, Lithuania, Czech Republic, Romania, Norway.

## References

[1] “Analytical report on road safety,” European Commission, Tech. Rep., 2010.

[2] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, E. Kaus, R. G. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knoppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, and E. Zeeb, “Making bertha drive. an autonomous journey on a historic route,” *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, Summer 2014.

[3] P. Czerwionka, “A Three Dimensional Map Format for Autonomous Vehicles,” Master’s thesis, Intelligent Systems and Robotics, Freie Universität Berlin, Germany, 2014.

[4] E. Guizzo, “How google’s self-driving car works,” *IEEE Spectrum*, 2011. [Online]. Available: http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works

[5] I. Parra, R. izquierdo, J. Alonso, A. García-Morcillo, D. Fernández-Llorca, and M. A. Sotelo, “The experience of drivertive-driverless cooperative vehicle-team in the 2016 gcdc,” *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–13, 2017.

[6] C. Fernández, D. Fernández-Llorca, and M. A. Sotelo, “A hybrid vision-map method for urban road detection,” *Journal of Advanced Transportation*, vol. 2017, no. 7090549, pp. 1–21, 2017.

[7] C. Fernández, D. F. Llorca, C. Stiller, and M. A. Sotelo, “Curvature-based curb detection method in urban environments using stereo and laser,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*, June 2015, pp. 579–584.

[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 *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Sept 2015, pp. 719–724.

[9] C. Fernández, R. Izquierdo, D. Fernández-Llorca, and M. Á. Sotelo, “Road curb and lanes detection for autonomous driving on urban scenarios,” in *17th International IEEE Conference on Intelligent Transportation Systems, ITSC 2014, Qingdao, China, October 8-11, 2014*, 2014, pp. 1964–1969.

[10] 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.

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

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[13] 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.

[14] 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.

[15] J. Redmon and A. Farhadi, “YOLO9000: better, faster, stronger,” *arXiv preprint arXiv:1612.08242*, 2016.

[16] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, “Mask R-CNN,” *CoRR*, vol. abs/1703.06870, 2017. [Online]. Available: http://arxiv.org/abs/1703.06870

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

[18] 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.

[19] 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.

[20] H. Noh, S. Hong, and B. Han, “Learning deconvolution network for semantic segmentation,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1520–1528.

[21] 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.

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

[23] 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.

[24] C.-A. Brust, S. Sickert, M. Simon, E. Rodner, and J. Denzler, “Convolutional patch networks with spatial prior for road detection and urban scene understanding,” *arXiv preprint arXiv:1502.06344*, 2015.

[25] R. Mohan, “Deep deconvolutional networks for scene parsing,” *arXiv preprint arXiv:1411.4101*, 2014.

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

[27] 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.

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

[29] 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 *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[30] H. Kong, J. Y. Audibert, and J. Ponce, “General road detection from a single image,” *IEEE Transactions on Image Processing*, vol. 19, no. 8, pp. 2211–2220, Aug 2010.

[31] J. C. McCall and M. M. Trivedi, “Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 7, no. 1, pp. 20–37, 2006.

[32] S. Nedevschi, R. Schmidt, T. Graf, R. Danescu, D. Frentiu, T. Marita, F. Oniga, and C. Pocol, “3d lane detection system based on stereovision,” in *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*, Oct 2004, pp. 161–166.

[33] A. Wedel, H. Badino, C. Rabe, H. Loose, U. Franke, and D. Cremers, “B-spline modeling of road surfaces with an application to free-space estimation,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 4, pp. 572–583, Dec 2009.

[34] L. Bentabet, S. Jodouin, D. Ziou, and J. Vaillancourt, “Road vectors update using sar imagery: a snake-based method,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 8, pp. 1785–1803, Aug 2003.

[35] J. Beck and C. Stiller, “Non-parametric lane estimation in urban environments,” in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, June 2014, pp. 43–48.

[36] A. V. Nefian and G. R. Bradski, “Detection of drivable corridors for off-road autonomous navigation,” in *Image Processing, 2006 IEEE International Conference on*. IEEE, 2006, pp. 3025–3028.

[37] F. Moosmann and C. Stiller, “Velodyne slam,” in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, June 2011, pp. 393–398.

[38] R. Valencia, E. H. Teniente, E. Trulls, and J. Andrade-Cetto, “3d mapping for urban service robots,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2009, pp. 3076–3081.

[39] P. Pfaff, R. Triebel, C. Stachniss, P. Lamon, W. Burgard, and R. Siegwart, “Towards mapping of cities,” in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, April 2007, pp. 4807–4813.

[40] 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.

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

[42] J. Fritsch, T. Kuhnl, and A. Geiger, “A new performance measure and evaluation benchmark for road detection algorithms,” in *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*. IEEE, 2013, pp. 1693–1700.

[43] M. A. Sotelo, J. Nuevo, L. M. Bergasa, I. Parra, and D. Fernández, “Road vehicle recognition in monocular images,” in *Proceedings of the IEEE International Symposium on Industrial Electronics. ISIE 2005*. IEEE, 2005.

[44] S. Álvarez, M. A. Sotelo, I. Parra, D. F. Llorca, and M. Gavilán, “Vehicle and pedestrian detection in esafety applications,” in *Proceedings of the World Congress on Engineering and Computer Science.*, 2009.

[45] D. F. Llorca, V. Milanés, I. P. Alonso, M. Gavilán, I. G. Daza, J. Pérez, and M. Á. Sotelo, “Autonomous pedestrian collision avoidance using a fuzzy steering controller,” *IEEE Trans. Intelligent Transportation Systems*, vol. 12, no. 2, pp. 390–401, 2011.

[46] V. Milanés, D. F. Llorca, J. Villagra, J. Pérez, C. F. Lopez, I. Parra, C. González, and M. Á. Sotelo, “Intelligent automatic overtaking system using vision for vehicle detection,” *Expert Syst. Appl.*, vol. 39, no. 3, pp. 3362–3373, 2012.

[47] 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 *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.