ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/361928391

Real-time road detection implementation of UNet architecture for autonomous driving

Conference Paper · June 2022

DOI: 10.1109/IVMSP54334.2022.9816237

CITATIONS
0

READS
45

4 authors, including:

Danut -Vasile Giurgi
Université de Haute-Alsace
3 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Thomas Josso-Laurain
Université de Haute-Alsace
30 PUBLICATIONS **140** CITATIONS
SEE PROFILE

Maxime Devanne
Université de Haute-Alsace
34 PUBLICATIONS **610** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

KERAAL - Kinesiotherapy and Rehabilitation for Assisted Ambient Living View project

Deep Learning in Autonomous Driving View project

All content following this page was uploaded by Danut -Vasile Giurgi on 12 September 2022.

The user has requested enhancement of the downloaded file.

Real-time road detection implementation of UNet architecture for autonomous driving

Dănuţ-Vasile GIURGI, Thomas JOSSO-LAURAIN, Maxime DEVANNE, Jean-Philippe LAUFFENBURGER
Institut de Recherche en Informatique, Mathématiques, Automatique et Signal
(IRIMAS UR 7499), University of Haute Alsace, Mulhouse, France
E-mail: vasile.giurgi@uha.fr

***Abstract*—This paper presents a real-time implementation workflow of neural networks for autonomous driving tasks. The UNet structure is chosen for a road segmentation task, providing good performance for low complexity. The model is trained and validated using two datasets, KITTI (validation of the model with respect to state of art) and a local highway dataset (UHA dataset), collected by the laboratory research team. The performance of the model for road detection is evaluated using the F1 score metric. After a simulation validation on both sets, the model is integrated into a real vehicle through the RTMaps platform. The application is tested in real-time conditions, around the city, under various weather and light. Finally, the proposed model proves low complexity and good performance for real-time road detection tasks.**

I. Introduction

Designing autonomous driving (AD) cars requires perception, path planning, and control [1], [2]. With the increasing availability and improvement of sensors in the market, perception tasks have been upgraded considerably in the past few years. One of the most common sensors used is the camera, however self-driving performance increases when using additional information from other sensors like GPS, Radars, or LIDARs [3]. In autonomous navigation, the interest stands for detecting traffic participants such as cars, pedestrians, and objects/areas located around the car.

Processing a flow of images with respect to real-time constraints requires an important computational power that may not be present in manufactured cars. Nowadays, commercial cars are equipped with industrial embedded systems with low processing capabilities, low energy, and low memory. When dealing with advanced image processing techniques, neural networks have shown very good detection performance these last years but require to consider the computational time as a performance criterion. Thus, autonomous vehicles become more realizable. Examples can be found in the DARPA Urban Challenge competition from 2007 [4], the Google car running on highway, the development conducted by Vislab in 2013 or the Tesla autopilot in 2015.

In the academic world, research centers perform significant work. Datasets like KITTI [6], Berkeley DeepDrive [7], A2D2 or generated by CARLA simulator [5] are exploited for various autonomous driving tasks. Teichmann *et al.* explored techniques to measure computational time for semantic segmentation tasks [8] within the KITTI dataset. Neven *et al.* [9] worked on scene understanding using the Cityscapes dataset. Another real-time endeavor on the same dataset is developed using an ENet architecture [10]. An attempt using 3D LiDAR point clouds is intended for real-time semantic segmentation in the work conducted by Wang *et al.* [11] with the PointSeg architecture. Time-critical tasks performances are also exploited on the KITTI benchmark by Bai *et al.* in their road segmentation application [12]. In a further work, Jang *et al.* aimed to both explain and reduce the end-to-end delay for self-driving cars [13]. However, all of the previous works refer to simulations conducted with their proposed models applied for various datasets, yet not tested and validated into an embedded system running in real-time conditions.

This paper focuses on tests conducted on an experimental prototype equipped with a real-time embedded system. Therefore, an equilibrium between good results with respect to the state-of-the-art and a low complexity neural network (NN) that can be implemented inside the car is desired. Here, a deep learning model is integrated into a car's embedded system. Furthermore, the application is subject to testing in real-time experiments. Firstly, a UNet model is trained and validated with appropriate datasets from the existing research for self-driving cars. Once the model achieves good simulation performances, the goal is to integrate it within the car. The experimental setup will require finding the right cameras and the compatible software needed for real-time data acquisition. Finally, the overall target is to develop an end-to-end real-time running application for road detection tasks.

The paper is structured as follows: Section II describes the experimental setup (platform); Section III stands for Real-time Implementation (NN model and integration); the fourth section, Road Detection Results, illustrates the results obtained (simulation and real-time obtained results); and finally, Section IV highlights the Conclusions.

II. Artemips experimental platform

A. Experimental Setup

Concerning the experimental setup, the IRIMAS laboratory possesses a vehicle called *ARTEMIPS* equipped for autonomous driving development. Figure 1 illustrates the features of the car.

Different devices are interconnected within the car. These are the main computer (Control Process Unit), LIDAR Units (2 Velodyne LIDARs + 2 IBEO Lux LIDARs), global navigation satellite system (GNSS), radar sensors, and cameras (Manta).

In this work, only the RGB camera is used for road detection tasks. To assess the real-time constraint, the computational power is limited to a CPU-based embedded computer. The

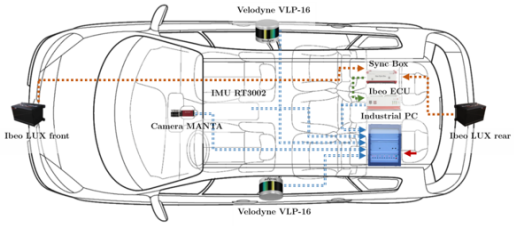


Fig. 1: ARTEMIPS Car

photographic device illustrated in Figure 1, is mounted behind the windshield as in the next image. The camera is a Manta G-507 (Allied Version) incorporated with a Sony IMX264 sensor that runs 23.7 frames per second at a 5.1-megapixel resolution. The connection between the camera and the computer is of Gigabit Ethernet (GigE) type, allowing the image recorder to transmit data at a rate of 1Gb/s.

For the implementation of the road detection task, the proposed architecture is developed using Tensorflow and Keras. Consequently, the model is trained out of the prototype on the Graphics Process Unit (GPU) board, NVIDIA RTX 2080. For the experimental testing, the model is incorporated into the main computer (CPU) with the Real-Time Multisensor Applications (RTMaps) platform.

B. Datasets

The considered datasets include KITTI [1] and a dataset directly created from the task and based on images from a local highway in the city of Mulhouse, France. Therefore, considering KITTI, the focus is on urban unmarked lane category images, while the collected dataset from the highway is annotated within the entire road as a navigable area. For the KITTI dataset, 98 images have been considered, whereas, in the set of data collected by the IRIMAS research team (UHA dataset [2]), 200 images have been investigated. The data split ratio is 80% for the training (80 frames for KITTI, respectively 160 for highway) and the remaining 20% for validation (18 frames for KITTI and 40 images for the UHA dataset).

III. Real-Time Implementation

A. UNet Architecture

In segmentation and computer vision applications, deep learning represents an important contributor. In 2015, Ronneberger *et al.* [14] developed the UNet architecture, which belongs to the Convolutional Neural Network (CNN) family and provides great results in medical imaging segmentation tasks. Contemporary, UNet performs well and shows strong capabilities for delimiting surfaces, since it considers the spatial correlation and the geometric information of the structure to where it is applied to. git The UNet model (Figure 2) is divided in two parts: *encoder* (left side) and *decoder* (right side) united by a fixed block in between often called *bottleneck*. These parts represent the contracting side (encoder) and the expansive side (decoder).

[1] http://www.cvlibs.net/datasets/kitti/eval_road.php
[2] https://github.com/vasigiurgi/RT-road-detection-with-Unet-for-AV

For the implementation, images are downsampled to a lower resolution to decrease the complexity. Therefore, from an original size of $2464H \times 2056W$, the new resolution becomes $1016H \times 1232W$. The model uses a classical UNet architecture with concatenations. The entire architecture uses fourteen convolutional layers, halved in the two parts. In the encoder three blocks are used to downsample the information. Each one of the blocks is composed of two convolutional operations with the same padding, rectified by a ReLu objective function. The convolutional layers are followed by a max-pooling block with a pool size of 2×2. The max-pooling layers are used for translation invariance. At the end of each block, there is a Dropout regularizer. In this way, the network is simplified and unimportant neurons are shut down randomly after each layer. The opposite side represents the reflection of the encoder. The decoder uses three modules to upsample back the data and rebuild the image. Hence, data is interchanged between the encoder and decoder through the concatenation connections. As for the decoder part, each big block is composed of two convolutional layers to propagate features, followed by upsampling layers to reconstruct the features. In the encoder, the layers are rectified by the ReLU functions, keeping the same padding. Figure 2 shows the block architecture of the proposed model. The part between the encoder and decoder

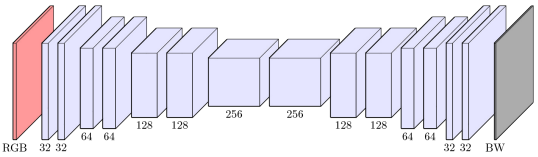


Fig. 2: The proposed UNet Model Arhitecture

represents the bottleneck which has two 256-channels convolutional layers each. The role of this bottleneck is to limit the NN to learn the compressed information. It is intended that this part consists only of meaningful data for the reconstructing part. The architecture ends with a sigmoid function and another Dropout regularizer. The sigmoid computes results between $(0$ and $1)$ so that the neural network can separate the features in regions that belong or do not belong to the road. The entire architecture has 3×3 kernel filters and multiple channels. The channels vector distribution $[32, 64, 128, 256,\ 128, 64, 32]$ can be observed from left to right. In the contracting component, the network is downsampled and encodes from 32 to 128 channels. Therefore the bottleneck continues with two layers of 256 channels each. The expansive part starts upsampling and decoding from 128 channels to 32. In the encoder, the input image comes as 3D data (RGB imge) and after decoder operations, the output becomes a 2D tensor (representing the monochrome prediction).

B. Integration within RTMaps platform

In the autonomous vehicle prototype, the RTMaps treats the information generated by the sensors. It receives the camera images with corresponding timestamps in real-time and computes them into the model that has been developed. For the RTMaps integration, the neural network model is adapted for a Python Bridge Component (Figure 3). The Python Bridge is a block that contains a Python file, written in a manner

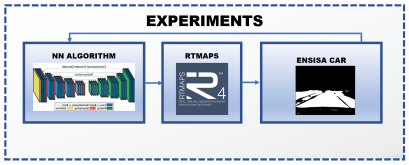


Fig. 3: RTmaps Workflow Integration

that allows running the model in a loop and integrates the input-output interfaces with other components. Therefore, the Python Bridge allows only models of frozen graph type, so the classic NN model needs to be transformed into a frozen graph model to execute the road prediction based on real-time images. The conversion method is illustrated in the following algorithm (Alg. 1):

Algorithm 1 Conversion to the frozen graph

function EXPORT KERAS MODEL AS A FROZEN GRAPH ($keras\_model$)

I. Freeze the Keras model using TF 2. x:

$$SavedModel \Rightarrow GraphDef$$

II. Optimize the *frozen graph* using TF 1.x:

$$GraphDef \Rightarrow GraphDef$$

III. Convert the optimized *frozen graph* back to *SavedModel*

$$GraphDef \Rightarrow SavedModel$$

end function

Consequently, after the model is integrated into the Python bridge, the block is merged within the full diagram, as depicted in Figure 3 and it is compiled to a RTMaps package.

IV. ROAD DETECTION RESULTS

A. Simulation Results

Following Section 2, the results are divided into two categories: the results for unmarked lane images from KITTI dataset, and the results for highway images collected on the ARTEMIPS platform.

To evaluate the road detection tasks, accordingly to the KITTI workbench, Fritsch *et al.* propose the F1 score metric [6]. Considering precision and recall (Equation 1), the interest is for calculating the F1-measure score [15].

$$\text{Recall} = \frac{TP}{TP+FN}, \quad \text{Precision} = \frac{TP}{TP+FP} \tag{1}$$

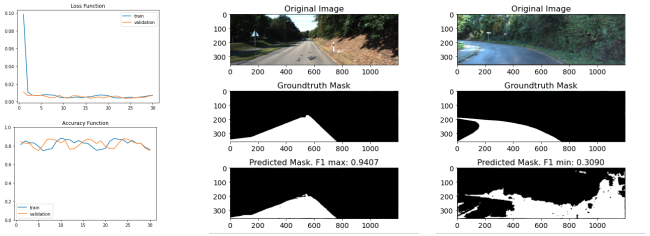
Equation 2 is computed as a harmonic mean and reflects the F1 score:

$$F1 = \left(1+\beta^2\right) \frac{\text{Precision} \cdot \text{Recall}}{\beta^2\ \text{Precision} + \text{Recall}} \tag{2}$$

where $\beta = 1$, and the other two metrics (precision and recall), are represented of the following: TP variable represents the true positives, FP false positives, and FN is used for false negatives.

Initially, the UNet model is trained on a partition of road detection KITTI dataset, 98 unmarked lane images: 80 images training and 18 images for validation. The model is constructed based on a UNet architecture described in the implementation part, but with adapted dimensions of the input data ($360H \times 1200W$). Here, the model is trained for 30 epochs using Adam optimizer (Figure 4a). The validation trend follows the training trend and the prediction accuracy stabilizes around 0.80.

On the KITTI website, for the same category of unmarked lanes, the best score obtained is realized by the method *SNE-RoadSeg+* [16] with a Max F1 score of 0.9704. Figure 4 shows results obtained by the proposed model on the KITTI dataset.



(a) Accuracy (b) Best Case (c) Worst Case

Fig. 4: Kitti Dataset Results

Figure 4b represents the best case for the validation dataset and it has a Max F1 score equal to 0.9407, whereas the right side (Figure 4c) is the worst F1 score case, equal to 0.3090. The average score for the validation dataset is 0.8475 which means most of the predicted images are closer to the best F1 score than to the worst F1 score. Conclusively, the model achieves good performance with a Max F1 score close to the top-20 on the KITTI dataset. The performance could be improved by working on the worst cases that decrease the average, however, to keep a network that can be embedded, the choice is made not to improve the architecture, i.e. increasing the number of layers and the number of neurons, so increasing the complexity. Keeping in mind that the best scores from the KITTI website might have behind robust and high-complexity neural network models, the simple UNet architecture that has been proposed proves to have good results for the embedded part. The model is applied to the UHA dataset. The original dimensions of the raw images are adjusted from ($2056H \times 2464W$) to ($1016H \times 1232W$) for computational reasons. For this experiment, 50 images have been considered to be tested. Thus, with respect to the ground truth, the model has to predict the entire road, including the opposite way and line markers.

Figure 5a represents the original image of the best prediction results. For the UHA dataset, the model provides efficient F1 score values. The results obtained from over 50 images from the validation dataset are the following: Mean F1 Score: 0.9798, Max F1 Score: 0.9922, Min F1 Score: 0.9480. Figure 5, respectively Figure 6 represent the best case and the worst case prediction from the validation datasets. The reference mask is shown between original image and predicted image. On the right side, there is the predicted data. For the best case, the F1 Score is 0.9912, which is very close to 1. This means that the road has been predicted smoothly, almost one-to-one with the ground-truth image.

Consequently, the worst-case prediction case is presented in Figure 6, where the F1 score is 0.9480.

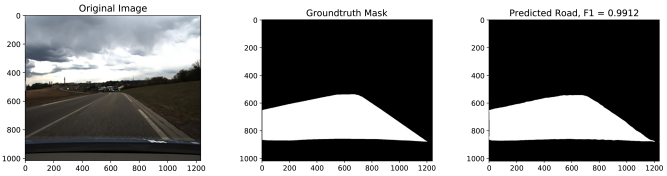


Fig. 5: Best predicted F1 score

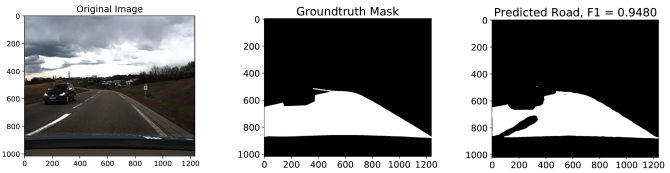


Fig. 6: Worst predicted F1 score

The worst score is around 5% less accurate than the best score, but the road is still predicted more than 90% of its surface. Moreover, the average score of the predicted images is 0.9798 which is closer to the best case prediction. Another important aspect of the worst-case prediction is related to the prediction of the line markers from the road, which have not been taken into account within the annotation process. Therefore, the model should predict the navigable area including the line markers, with respect to the annotations (lane delimitation markers have been annotated as part of the road). However, considering the metric evaluation, the lane markers that have not been predicted as the navigable area will be considered as the wrong prediction, which justifies the score in the worst case. Furthermore, the model performs well for the car object. The car coming from the opposite direction has been successfully predicted as non-road. To validate the work two sets of experiments have been done. Firstly the model has been validated on the offline dataset from the highway, in the validation subset; secondly, the model has been validated on the road in real-time tests on the ARTEMIPS car. The next table resumes the results presented in this work:

Dataset	Worst F1 Score	Best F1 Score	Mean F1 Score
KITTI	0.3090	0.9704	0.8475
UHA dataset	0.9266	0.9922	**0.9798**

TABLE I: Results UNet against KITTI and UHA dataset

B. Real-time results: Validation on real-time enviroment

The practical experiment represents the validation of the project's goals. In this way, the experiments validate an artificial neural network model's performance in the real world. For the practical part, the model is integrated with the RTMaps platform. The camera is connected to the main PC for real-time image acquisition. Due to the car's limitations, a CPU (Control Process Unit) computer is set to run behind the experiments. Therefore, the proposed model is tested in real-time trials under various weather (similar to dataset collection) and light conditions. The vehicle speed on the highway during the test oscillates between 50 and 110 km/h. The model performs well for the real-time images and proves significant accuracy for most of the images, however, it performs less accurately than for offline simulations with images of the same category. An image with an example of the real-time results is presented in Figure 7. On the left side, the camera provides real-time images. On the right side is the real-time prediction. Even if the output image is good, the prediction has a little delay due to embedded system's computational limitations.

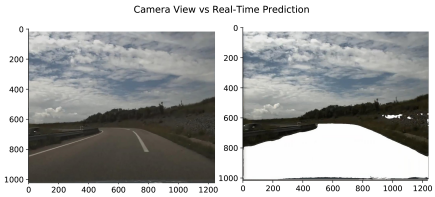


Fig. 7: Real-Time Experiments on the highway

The prediction has been challenged in a few scenarios. For example, when the light conditions suffer a delayed update (e.g. under the bridges or in ramp spots) or when the highway is crowded, the model encounters problems in predicting accurately. In this case, the real-time efficiency could not be properly evaluated since the car computer did not have GPU computational resource. Consequently, the prediction is obtained with an execution time of $0.16 - 0.25$ frames per second (fps). The hypothesis is made that with a GPU embedded computer, the performance can reach satisfying results (a prediction in an acceptable delay regarding the vehicle speed).

V. Conclusions

In this work, a real-time application concerning autonomous driving functionalities in an embedded system has been developed. The implemented model succeeds to perform road segmentation in real-time (an execution time of $0.16 - 0.25 fps$), based on successive image acquisition. Additionally, the project addresses a simple implementation with a low complexity using a UNet architecture but manages to acquire good results on both KITTI and highway datasets. The model has performed accurately and efficiently in both in simulation and real-time experiments. In the validation results, the UNet model reached an average F1 Score equal to 0.9798, a best prediction F1 Score of 0.9922, and a worst-case prediction with an F1 Score equal to 0.9480. Regarding the real-time experiments, qualitatively the results have been accurate, but GPU features could improve significantly the performance. The delay has increased due to the low computational power resources. To gather faster results a knowledge distillation approach can be considered to simplify the model. In this way, together with a GPU experimental setup, the model could predict the road in a faster way.

Acknowledgments

The authors would like to thank the French National Research Agency (ANR) project, the University of Haute-Alsace, and the Pierre-et-Jeanne Spiegel Foundation for all the support provided in the realization of this project.

REFERENCES

[1] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 1, pp. 33–55, 2016.

[2] H. Laghmara, M.-T. Boudali, T. Josso-Laurain, J. Ledy, R. Orjuela, J.-P. Lauffenburger, and M. Basset, "Obstacle avoidance, path planning and control for autonomous vehicles," 06 2019, pp. 1–7.

[3] J. Vargas, S. Alsweiss, O. Toker, R. Razdan, and J. Santos, "An overview of autonomous vehicles sensors and their vulnerability to weather conditions," *Sensors (Basel, Switzerland)*, vol. 21, pp. 1–22, 08 2021.

[4] M. Buehler, K. Iagnemma, and S. Singh, "The darpa urban challenge: Autonomous vehicles in city traffic, george air force base, victorville, california, usa," in *The DARPA Urban Challenge*.

[5] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

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

[7] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," 2018. [Online]. Available: https://arxiv.org/abs/1805.04687

[8] M. Teichmann, M. Weber, J. Zöllner, R. Cipolla, and R. Urtasun, "Multinet: Real-time joint semantic reasoning for autonomous driving," pp. 1–10, 12 2016.

[9] D. Neven, B. Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool, "Fast scene understanding for autonomous driving," pp. 1–5, 08 2017.

[10] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," pp. 1–10, 06 2016.

[11] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu, "Pointseg: Real-time semantic segmentation based on 3d lidar point cloud," pp. 1–10, 07 2018.

[12] L. Bai, Y. Lyu, and X. Huang, "Roadnet-rt: High throughput cnn architecture and soc design for real-time road segmentation," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–11, 11 2020.

[13] W. Jang, H. Jeong, K. Kang, N. Dutt, and J.-C. Kim, "R-tod: Real-time object detector with minimized end-to-end delay for autonomous driving," pp. 1–14, 10 2020.

[14] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015.

[15] D. M. W. Powers, "Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation," pp. 2–3.

[16] H. Wang, R. Fan, P. Cai, and M. Liu, "Sne-roadseg+: Rethinking depth-normal translation and deep supervision for freespace detection," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1–7.

View publication stats