ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.
Available from http://www.i6doc.com/en/.

Navigational Freespace Detection for Autonomous Driving in Fixed Routes

Aparajit Narayan[1] and Elio Tuci[2] and William Sachiti [1] and Aaron Parsons [1] *

1- Academy of Robotics
33 Cathedral Road, CF11 9HB Cardiff, Wales (UK)

2- Faculty of Computer Science, University of Namur
Rue Grandgagnage 21, 5000 Namur, Belgium

Abstract. Vision-based modules are largely exploited by autonomous driving vehicles to identify the road area and to avoid collisions with other vehicles, pedestrians and obstacles. This paper illustrates the results of a comparative study in which eight different vision-based modules are evaluated for detecting free navigational space in urban environments. All modules are implemented using Convolutions Neural Networks. The distinctive and innovative feature of these modules is the manner via which navigational freespace is identified from image inputs. The modules generate the coordinates of a triangle, whose area represents the navigation freespace. The relative position of the triangle top corner with respect to the image centre points toward the vehicle direction of motion. Thus, when trained on a fixed route, these modules are able to successfully detect the road-freepsace and to make appropriate decisions concerning where to go at roundabouts, intersections etc., in order to reach the final destination.

1 Introduction

The detection of drivable areas on a road (road detection) from camera images is a crucial aspect of autonomous driving. The problem is essentially a matter of identifying which portion/s of an image refers to freespace areas of a road where a vehicle can move on without incurring into any collision. Although various solutions have been proposed, the problem remains particularly challenging due to the fact that real-world conditions exhibit a great degree of environmental variation with regards to lighting, road-structure, traffic etc. In this paper, we illustrate a novel approach to the design of road detection from camera images for autonomous driving vehicles which indicates the image freespace area with a scalene triangle. Since the triangle top corner points towards the direction of motion, this approach can potentially integrate road detection and route planning in a single vehicle control module. That is, the module denotes free driveable space on the road and make decisions concerning how to drive in roundabouts, which road to take at junctions. The route planning aspect of our module can be exploited to keep the vehicle on a predefined fixed route.

A significant amount of academic studies have been dedicated to the problem of road detection and route planning, generally by treating the two aspects as separated problems to be tackled by distinctive modules. In this paper, we focus

*This work is funded by Academy of Robotics.

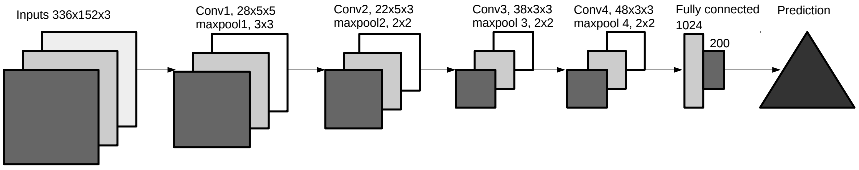

Fig. 1: CNN structure used in Module I, II, and III.

on the design of modules for the automatic road detection problem, and we show how these modules can be also exploited to solve the route planning problem assuming that the vehicle is travelling on a fixed predefined route. As far as it concerns road detection, the large majority of the works in the literature concern the detection of lane boundaries in structured environments such as highway roads [1, 2]. Initial approaches using manually specified filters for edge-extraction, were erorr-prone due to shadows, vehicle occlusion and sections where the lane marking disappeared due to weathering. Current state of the art works such as [3] have moved towards the use of deep convolutional neural networks (CNNs) which are better suited for handling the aforementioned issues. CNNs, with their ability to learn a robust hierarchy of features, offer a more attractive solution to the issue of detecting complex road scenes as shown in [4, 5, 6, 7].

In this paper, we illustrate and compare three different CNN modules which are trained to solve the road detection and the route planning tasks on images taken from a vehicles travelling on a fixed route. In particular, we recorded five videos with a GoPro camera mounted on a car, while driving around a fixed route that starts and finishes in the same location. These five videos has been recorded by driving the car in different dates and in different time of the day. This is meant to capture the environmental variation that is present within the same route of roads. Variation with regards to traffic, lighting conditions, presence of pedestrians were observed in this compilation of these videos. Frames for each drive video were extracted at 4 fps and organised in five datasets (I,II,III,IV,V): with 1341, 1276, 1299, 1205 and 1038 images respectively. The drives took place in a mostly residential area of Surrey, London (UK), and involve the vehicle being driven from a fixed starting point, exiting the residential area which is a one lane road onto the high-street which is a two-lane road, going 360° at a round-abound and returning to the starting-point travelling in the opposite direction. The route also features two junctions where the correct turn needs to be made on the way forth and back. The residential sections of the road have restricted space with rows of cars often parked on the side. On-road traffic increases significantly after the second turn onto the high-street. In section 2, we describe the three modules. In section 3, we illustrate the results of our study, and in section 4 we draw our conclusions.

2 Methods

We compare the performances of three CNN Modules, which share the same network structure, and generate same output. Figure 1 shows the CNN structure common to all three Modules. This CNN structure was fixed after a period of experimentation with different structures. The number of layers in the network are explicitly kept small in consideration of the fact that these Modules are meant to be deployed in a real-time software pipeline and will be one of the several CNNs that need to run their update cycle multiple times per second. While the size of the input images is the same for all modules (i.e., 336 x 152 pixels), the way in which colours are represented in these images is different for each Module. In Module I, the inputs are raw camera images in standard RGB colour model. In Module II the input are images presented in an hybrid colour model referred to as HSA. This colour model is made by combining Hue (H) and Saturation (S) from the HSV colour model, and the *a channel from the L*a*b colour model. Contrary to RGB, HSA allows to treat separately information concerning colour and brightness of the images. Thus, it should represent a more robust road detection module to allow the vehicle to operate in extreme lighting conditions. In Module III, the input are images generated by merging, using a regression model, the output of three different CNNs already pre-trained for carrying out different image segmentation tasks. These three CNNs take as input the raw images from our dataset and perform the following segmentation tasks. The first CNN is the YOLOV3 from the darknet framework (see [8]). This CNN performs an object detection/classification task by generating bounding box coordinates for every object detected in an image. The boxes are represented by opaque rectangles of different colours on each image. General objects (traffic lights, road sign) are coloured as pink; cars, trucks, buses are marked blue, two-wheelers including cycles are coloured red and pedestrians are marked yellow. The second CNN is the ICNET for scene segmentation (see [9]). Each pixel of the original image is coloured as one of 18 colours according to the "CITYSCAPES" dataset labelling convention. The third is a CNN for lane detection (see [3]). This CNN outputs a final frame with coloured pixels predicting the road lanes and black pixels for all other areas.

Each Module outputs four real numbers used to draw a triangle in the raw camera image. The first number x_1 refers to the width of the leftmost corner of the triangle with pixel coordinate (x_1, 580). The second number x_2 refers to the width of the rightmost corner of the triangle with pixel coordinate (x_2, 580). The third (x_3) and the fourth number y refer to the width and the height of the top corner of the triangle with pixel coordinate (x_3, y). The coordinate (0,0) corresponds to the top leftmost pixel. We exclude from the freespace area prediction algorithm the portion of the images directly in front of the vehicle (that is, the portion of the image with a row coordinate bigger than 580 pixels). The position of the triangle indicates the drivable freespace in the raw camera image and the shape of the triangle indicates the direction of motion. How does the triangle top corner indicate the direction of motion? Assuming the middle column of pixels in the image plane (pixel column 672) corresponds to the vehicle pointing straight, a simple trajectory can be generated by considering two vectors, the first one originating in the image coordinate (672, 608) with the head in the image coordinate representing the triangle centroid; and the second one originating in the image coordinate representing the triangle centroid with the head in the image coordinate representing the triangle top corner/vertex. Thus, for every time step the vehicle needs to make adjustments to the trajectory (steering, speed), a motion control module can generate actions that take the vehicle initially in the direction of the first vector and then in the direction of the second vector. We propose a proportional differential control scheme wherein the trajectory is a function of current and prior values of the two vectors. Further methodological details can be found in the supplementary paper link https://bit.ly/35bUro0.

3 Results

Besides presenting the results of evaluation tests of the three Modules illustrated in section 2, we have also evaluated the freespace prediction generated by merging the output of Module I, II, and III for each frame. We refer to the merging approach as Module IV. We implement five techniques for merging the three Modules frespace prediction. The first technique, referred to as Module IVa, is a simple averaging scheme where each of the four output (that is x_1, x_2, x_3, and y) is generated by averaging the three corresponding real numbers generated by each Module. The other four merging techniques are different regression schemes implemented using the python scikit-learn library. In particular, we used the **k-nearest neighbour** regression (Module IVb), the **random forest** regression (Module IVc), the **decision tree** regression (with the depth set at 8, Module IVd), and the **gradientboost** regression (Module IVe). The regression models are first trained on sample data before being used to combine the predictions of the individual networks on the test data. Because of the limited amount of images, we employed a scheme wherein we chose a sample of 1000 images from the training set. The regression models were then trained using the annotated freespace area as ground-truth and the predicted triangle coordinates generated by I, II, and III as input.

We carried out two training runs. Modules are first trained on datasets III, IV, and V and tested on datasets I and II. In the second run modules are trained on datasets I, II, and III and tested on datasets IV and V. Two different evaluation metrics are used to compare the position of the Modules' generated freespace triangles with the human ground-truth annotation indicating the drivable freespace. Metric A refers to average percentage of annotated triangle (i.e., the ground-truth) covered by the Modules' generated triangles. Metric B refers to average percentage of Modules generated freespace triangles that lies outside the ground-truth triangles. Metric A and B provide evidence about the extent to which the Modules' generated freespace triangles matches the annotated ground-truth triangles. These percentages are computed using functions of the OpenCV

M.	**Dataset I**				**Dataset IV**				**Dataset V**			
	A (%)		**B** (%)		**A** (%)		**B** (%)		**A** (%)		**B** (%)	
	m	*std*	*m*	*std*	*m*	*std*	*m*	*std*	*m*	*std*	*m*	*std*
I	11.9	32.4	39.6	36.5	19.2	25.1	49.6	36.2	12.3	23.5	43.3	34.1
II	18.5	27.1	45.0	43.9	15.2	24.7	58.1	36.6	15.3	23.8	57.3	36.2
III	19.3	25.7	21.1	35.8	21.3	25.7	38.1	35.8	24.2	24.5	28.1	33.2
IVa	19.0	20.3	20.3	34.1	21.8	22.7	27.3	34.7	21.9	20.9	21.4	32.6
IVb	24.6	25.0	16.1	34.6	24.5	25.7	16.1	34.3	24.8	25.3	15.0	32.5
IVc	17.5	31.4	19.2	25.5	17.9	19.3	14.9	35.7	24.3	19.2	10.6	31.6
IVd	29.0	26.2	17.5	34.9	26.2	26.0	17.8	36.5	27.3	19.9	10.5	34.3
IVe	29.0	27.08	14.8	36.0	21.7	21.0	14.7	34.9	26.8	19.6	10.8	32.0

Table 1: Table showing median (m) and standard deviation (std) of the metric A and B for eight different Modules (M) tested on datasets I, IV, and V.

library. For each image we scan through all the pixels and we test firstly if a pixel lies within the contours of the annotated triangle and/or within the contours of a Module's generated triangle. Metric A is derived from the proportion of pixels within the contours of the annotated triangle that are also within the contour of the Module generated triangle. Metric B is derived from the proportion of pixels within the contour of the Module generated triangle that are not within the contours of the annotated triangle. The optimum for metric A is 100%, the optimum for metric B is 0%.

Table 1 illustrates the results only for tests on Dataset I, IV, and V. The results of the evaluations on dataset II can be found at `https://bit.ly/35bUro0` together with images illustrating the Modules' freesapce predictions, results of two other Metrics (C and D) and further methodological details of this research work. The results in Table 1 indicate that, first, all different flavours of Modules IV tend to do better than Modules I, II, and III. This suggest that there is merit in combining the detections of the three individual Modules using the regression approach. We notice however that, even for all Modules IV, the generated triangles tend to cover, on average, a relatively small portion (between 20% and 30%) of the annotated triangles (see Metric A in Table 1 for all Modules IV and for all Datasets). The best matching is registered for Module IVd, and IVe for Dataset 1, where the percentage are slightly below 30%. Metric B however confirm that on average, for all different flavours of Modules IV, only a small proportion of the generated triangles lies outside the annotated triangles (between 15% and 20%, see Metric B in Table 1 for all Modules IV and for all Datasets). Visual examination of frame sequences pointed to the fact that even though they are almost always positioned within the annotated triangle the Modules' generated triangle tend to be smaller than the annotated one. Indeed, the predicted freespace shapes, irrespective of the Modules used are generally narrower at the base compared to annotated triangles. This is the main cause to account for the relatively small values recorded for metric A, and also for the relatively high standard deviations in metric A. In summary, Modules IVd and IVe produce the best results by generating triangles that, although smaller than the annotated one, they lie within the annotated drivable area of the road, clearly pointing to the correct direction of motion. Results for metrics C and D concerning the direction of motion are shown in `https://bit.ly/35bUro0`.

4 Conclusions

The methodologies presented in this work have laid the foundation for an interesting avenue of research which can provide a novel means for translating camera inputs to directional cues that integrate mapping/route-planning information. Despite limitations arising from limited training images and noisy annotation, all Modules were found to be generally capable of learning road freespace. Module III was found to be more robust and less prone to including areas beyond the ground-truth adding weight to our method of using fused perception outputs of other neural networks as the input image. We also observe that combining the freespace predictions of these individual networks via simplistic regression models increase the performance. The final freespace triangle prediction arising from this regression-based combination scheme negates errors that may be present in individual network detections. Future work will focus on the development of the control component to generate the vehicles actions (i.e., acceleration and heading direction) from the freespace triangle generated vectors.

References

[1] S. Thrun et al. Stanley: The robot that won the DARPA grand challenge. *Journal of Field Robotics*, 23(9):661–692, 2006.

[2] M. Ososinski and F. Labrosse. Automatic driving on ill-defined roads: An adaptive, shape-constrained, color-based method. *Journal of Field Robotics*, 32(4):504–533, 2015.

[3] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards end-to-end lane detection: an instance segmentation approach. *CoRR*, abs/1802.05591, 2018.

[4] C. Chen, A. Seff, A. Kornhauser, and J. Xiao. DeepDriving: Learning affordance for direct perception in autonomous driving. In *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*, pages 2722–2730, 2015.

[5] J.M. Alvarez, T. Gevers, Y. LeCun, and A.M. Lopez. Road scene segmentation from a single image. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *Proc. of the European Conf. on Computer Vision*, pages 376–389. Springer Berlin Heidelberg, 2012.

[6] A. Narayan, E. Tuci, F. Labrosse, and M.H.M. Alkilabi. Road detection using convolutional neural networks. In *Proc. of the 14^{th} European Conference on Artificial Life (ECAL)*, pages 314–321, 2017.

[7] A. Narayan, E. Tuci, F. Labrosse, and M.H.M. Alkilabi. A dynamic colour perception system for autonomous robot navigation on unmarked roads. *Neurocomputing*, 275:2251 – 2263, 2018.

[8] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018.

[9] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. Icnet for real-time semantic segmentation on high-resolution images. In *The European Conference on Computer Vision (ECCV)*, September 2018.