Neural Network-based lane detection and pose estimation for autonomous driving

Simon Steiger
Master Applied Research
in Engineering Sciences
Deggendorf Institute of Technology
94469 Deggendorf, Germany
Email: simon.steiger@stud.th-deg.de

***Abstract*—For automotive driver assistance systems like lane keeping or collision avoidance, it is crucial to know the position of the vehicle in it's driving lane. Due to massively diverse shapings and coloring of lane markings as well as street surfaces and lighting and weather conditions, it is a very hard task to solve purely algorithmically. In this paper, an alternative approach is proposed based on a Neural Network performing semantic segmentation on the input image of the front camera as the first step of image processing. All input data is provided by the open source vehicle simulation tool CARLA, image data transfer is handled by the Robot Operating System (ROS) and image processing is done using Python and OpenCV.**

I. Introduction

Deggendorf,
June 9, 2020

A. Prerequisites

1) Input data: The proposed approach relies solely on the image stream provided by the front camera of the vehicle. Resolution and aspect ratio can differ, being a compromise regarding the demanded field of view, additional functions (like stoplight detection) and processing time (higher resolutions tend to be more accurate but also more performance demanding). In most real world applications, the driver assistance systems do not solely rely on camera input, but also on sensors like radar and GPS. However in this paper, only the input data from the front camera is processed.

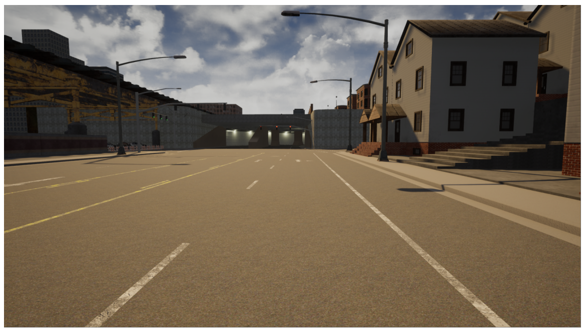

Fig. 1. Input image

2) Desired output data: The proposed algorithm is expected to deliver output data that is, in quantity as in quality, suitable to be used by a lane keeping algorithm. Several of those algorithms exist, from simpler geometric approaches to complex dynamic models using state-space control theory. However, most of them require one or more values of the same input data selection: The lateral deviation from the middle of the lane, commonly referred to as cross-track error e (at the current or future position), the angular deviation of the vehicle's pose relative to the lane Θ_e, and the lane curvature L.

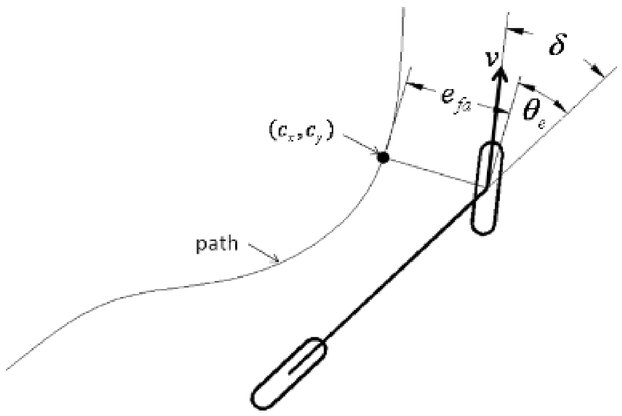

Fig. 2. Desired output data
[1]

Figure 2 shows a visual depiction of the cross-track error e a the angular deviation Θ_e. In this picture, the path can be interpreted as the middle of the lane, as this is the desired position of the vehicle while driving on a public road (as opposed tho e.g. a race track, where the path would be the so-called ideal line).

II. Algorithm Explanation

A. *Lane differentiation problem*

One of the hardest tasks in building a lane detection algorithm for autonomous driving is differentiating the ego-lane from the environment of the car. Due to the wide range of possible environments, multiple lanes, missing or weathered markings, this task can be made arbitrary complex. In addition, the reliability requirements are high and processing the image should not take too long as the steering controller needs a suitable update frequency to operate satisfactory.

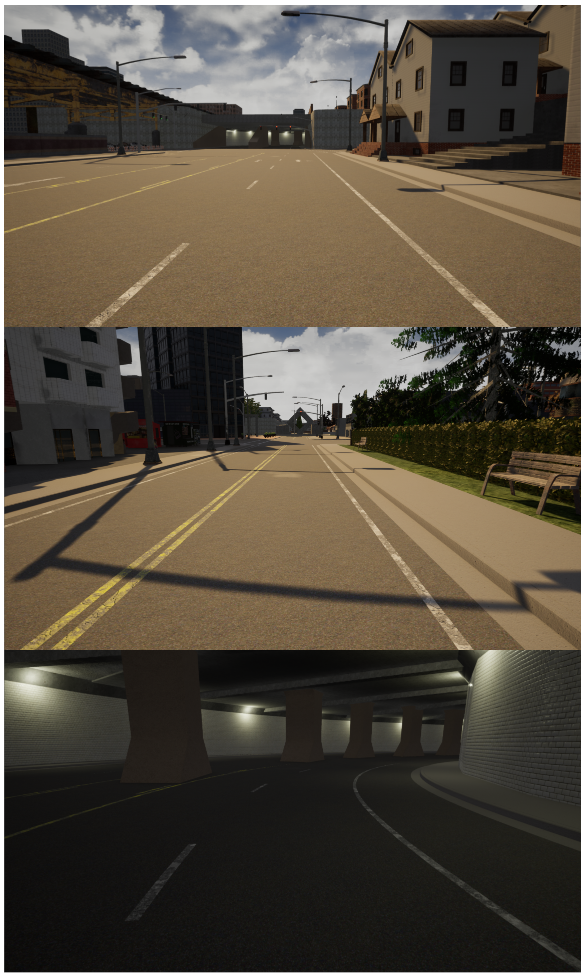

Fig. 3. Variety in input data

Figure 3 shows the broad variety in input data, even with constant type of road surface and weather condition. Lighting conditions and different colors of markings alone make the use of simpler color thresholding methods unsuitable, which shows the demand for a more sophisticated approach. While methods like edge detection and pattern recognition have been around and used for several years, the latest advances in machine learning have also opened up new opportunities, as the following chapter will discuss.

B. *Principle of Semantic Segmentation*

In order to differentiate certain areas of an image from its surroundings or background, several neural networks have been developed in the last years. In General, all those networks do the same thing by mapping each pixel of the input image to a specific class on the output image. As any other neural network, it has to be trained on manually labelled data beforehand. A common usecase of semantic segmentation in autonomous driving is pedestrian and vehicle detection, with a broad variety of datasets, e.g. Cityscapes.

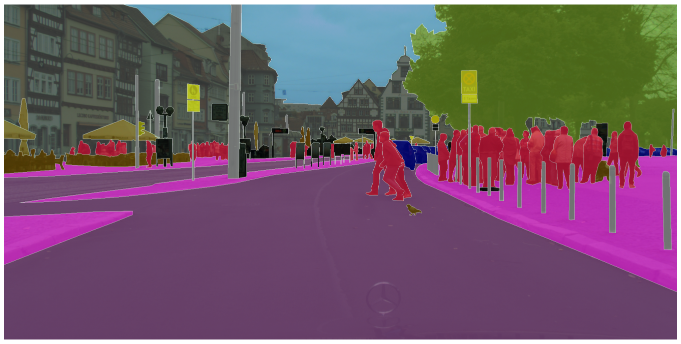

Fig. 4. Semantic Segmentation based on Cityscapes dataset [2]

Figure 4 shows the principle of semantic segmentation using the Cityscapes [2] color palette. However, this particular network does not fulfill the requirements of a lane detection algorithm, as it does not differentiate between the ego lane and the rest of the street. [4] has trained a network that is able to differentiate between the ego lane and adjacent lanes, which satisfies this requirement. Due to this and it's availability as a ROS Package, it was provisionally chosen as the solution to differentiate pixels belonging to the own lane from it's surroundings.

Figure 5 shows how despite unfavorable lighting conditions, the correct driving lane can be found using semantic segmentation as described in [4]. This particular network consists of a Deeplab V3+ on a MobileNet v2 backbone, trained on BDD100K driveable area, 513x513 input size[4]

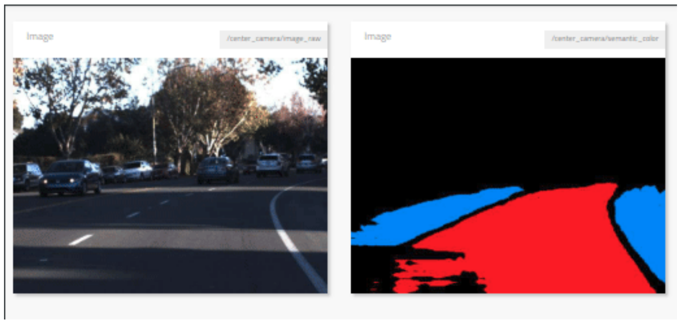

Fig. 5. Segmentation of ego lane [4]

C. Processing the segmented image

After the neural network has done it's work of segmenting the input image pixel-wise into "lane" and "surroundings", further processing steps have to be carried out in order to extract the needed output data for the steering controller. Binarizing the segmentation image leads to a binary segmentation mask that can be processed further:

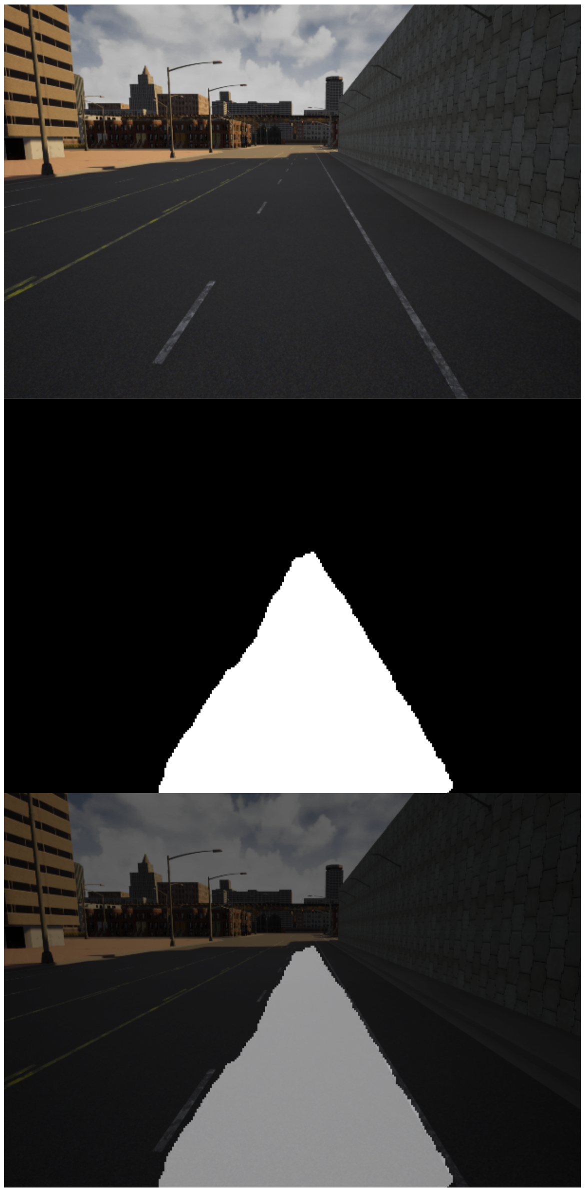

Fig. 6. Lane segmentation input image, binary segmentation mask and overlay for visualisation

Figure 6 shows the segmentation step from a standard RGB input image to a binary segmentation mask done by the neural network. The overlay in the third image is not part of the algorithm, but serves to show the accuracy achieved by the network.

1) Perspective transform: The following processing step transforms the front-facing segmentation image into a birdseye-view image using a perspective transform with a ROI (range of interest)-trapeze:

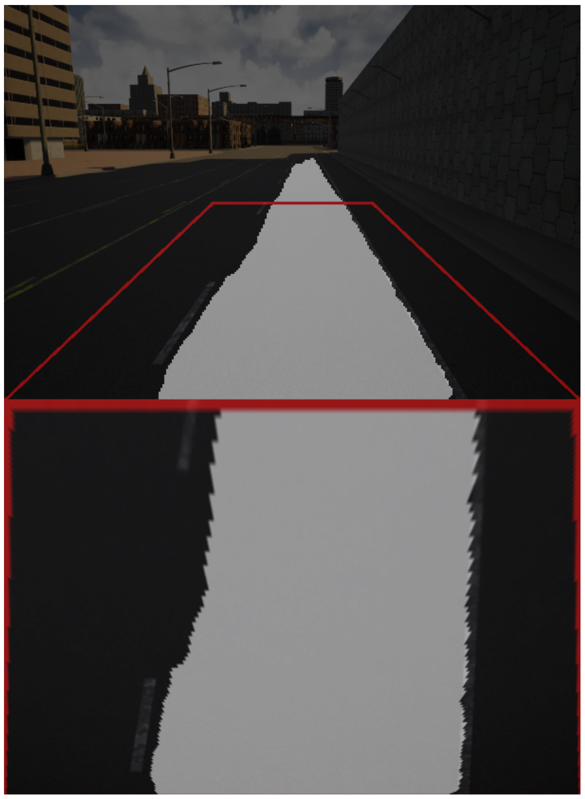

Fig. 7. Perspective transform

The perspective transformation expands the pixels inside the ROI trapeze to the full image size, resulting in a lower resolution on the narrower side of the trapeze and fewer changes in sharpness on the wider bottom side. Mathematically, the transformation step is done by multiplying the input image with a 3x3 transformation matrix (whose values are depending on the geometry of the area to transform and calculated in advance by OpenCV) so that the value of each pixel in the output image is calculated as:

$$\mathtt{dst}(x,y) = \mathtt{src}\left(\frac{M_{11}x + M_{12}y + M_{13}}{M_{31}x + M_{32}y + M_{33}}, \frac{M_{21}x + M_{22}y + M_{23}}{M_{31}x + M_{32}y + M_{33}}\right)$$

Fig. 8. Mathematical description of perspective transformation [3]

Due to the segmentation image being a binary image, the blur is of no account there. As visible in the upper image of Figure 7, the vehicle is not exactly aligned with the lane, resulting in the lane leaning slightly towards the right-hand side of the image (resulting from the vehicle facing too far left). This correlation can also be observed in the lower image, and is used in the further processing steps.Note that in the actual algorithm, only the segmentation mask is transformed. The transformation of the overlay image with the drawn was as an example for this paper for better intelligibility.

D. Information extraction

The final step is using the transformed segmentation image to extract information of the vehicle's position relative to the lane. This step is carried out by calculating the cross-track error at three points (which correspond three look-ahead distances) in the image, from a small one, a medium one to a far away one. At this development stage, the three look-ahead distances are determined experimentally and fixed, future improvement potential could lie in adjusting those values at runtime based on curvature and vehicle speed.The lane center coordinates are also smoothed by a moving average over the last 5 to 10 previous values to improve robustness against slight sporadic errors in the segmentation step as well as external disturbances. Using those three points, the angular deviation can also be calculated using angle functions. The lane curvature is approximated by a circular arc passing through all three points (and thus being clearly defined).

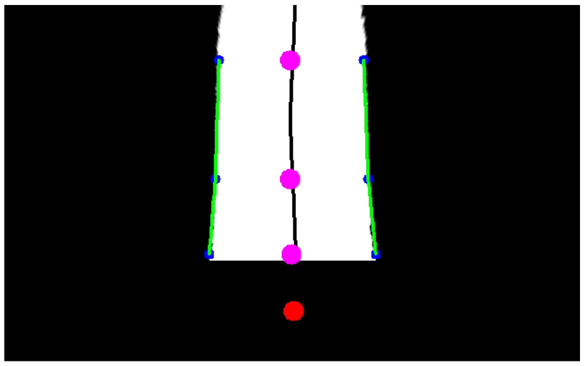

Fig. 9. Visualisation of extracted information

The three points of measurement are pictured in purple, as the center of the lane at three different look-ahead distances, while the blue dots and green lines mark the edges of the lane. Note that the cross-track error can only be measured in a certain look-ahead distance, but not directly at the vehicle due to the front camera looking ahead. However, some algorithms need the cross-track error at the vehicle's front axle as an input value. In this case, the cross-track error at the front axle can be estimated by extrapolating the lane curvature circular arc to the height of the vehicle (depicted in red).

III. Conclusion

In this paper, an approach for a lane detection algorithm using a semantic segmentation network was proposed. The prerequisites and goals of the algorithm have been presented, followed by a brief introduction into the principle of semantic segmentation. Subsequently, the perspective transformation step was depicted as well as the final step of extracting the required information from the transformed segmentation mask. The algorithm has been developed and tested using the open-source simulation software CARLA. In the main testing scenario, highway driving, it showed satisfying results in frame rate and accuracy. However, for future research, three issues shall be examined:

- In tight curves, the fixed ROI trapeze is not able to contain the whole segmentation of the lane, leading to unpredictable behaviour when the curvature radius falls below a certain point. An attempt to fix this issue could be implementing a dynamically self-adapting geometry of the ROI trapeze.
- In case of the vehicle leaving it's desired lane, e.g. due to a malfunctioning of the steering controller, external disturbances or in an overtaking scenario, the lane segmentation network can not determine which lane to segment (due to the network being trained only on images with the vehicle aligned right). Further research should go into training a segmentation network that is able to distinguish different lanes from each other, with the algorithm choosing its' desired lane for overtaking scenarios. This network should also be able to show more robust behaviour in case of the vehicle not being well aligned to the lane.
- At its current state, the algorithm fully depends on the segmentation done by one neural network. While this step, within it's current limits, showed to be fairly robust, further reasonability checks using either a second neural network or traditional imaging processing methods (e.g. color thresholding, edge detection) could prevent faulty behaviour in case of unexpected events.

Despite those remaining issues and room for improvement, the approach of using Semantic Segmentation to differentiate the ego-lane from it's surroundings has shown to be very promising. With further research going into improving robustness and adaptability, neural networks in general and semantic segmentation in particular could be a promising approach to develop autonomous driving even further.

Acknowledgment

I would like to thank my colleagues at B-Plus GmbH for their help. Further thank goes to the staff at the Laboratory of Autonomous Systems (LAS) at Deggendorf Institute of Technology. Last but not least, I want to thank my supervisor Prof. Thomas Limbrunner for his constant support and advice.

References

[1] Jarrod M. Snider. "Automatic Steering Methods for AutonomousAutomobile Path Tracking". MA thesis. Pittsburgh,Pennsylvania: Carnegie Mellon University, 2009.

[2] Marius Cordts et al. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[3] OpenCV Foundation. *Geometric Image Transformations*. URL: https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html.

[4] Dheera Venkatraman. URL: https://github.com/dheera/ros-semantic-segmentation.