Miloš Mitrović[1]
Vladimir Popović[2]
Dragan Stamenković[3]

JEL: R4
DOI: 10.5937/industrija50-41958
UDC: 629.331:004.89]:656.05
Original Scientific Paper

Implementation of traffic sign recognition on the scaled vehicle model

Article history:
Received: 3 July 2020
Sent for revision: leave blank
Received in revised form: leave blank
Accepted: leave blank
Available online: leave blank

Abstract: *The popularity of autonomous vehicles has grown in the past few years as autonomous systems are more and more present on vehicles. The most accessible way for students of mechanical and software engineers to learn about autonomous vehicles is by applying algorithms and systems necessary for autonomous driving on the scaled vehicle model. These models are, as in this case, and are equipped with all systems necessary for autonomous driving, such as a four-wheel drive powertrain, a suspension system, an electrically controlled steering system, a brain-computer and a camera. The goal of projects such as this one is to make the vehicle capable of autonomous driving on a designated track, obeying regular traffic rules and signs (for example, the vehicle has to perform a full stop when it approaches the stop sign). To make this possible, it is necessary for a vehicle to "know" which traffic sign is nearby, i.e., traffic sign recognition is required. For this purpose, traffic sign recognition is done by an artificial neural network. The training process of the proper artificial neural network will be shown in this paper.*

Keywords: *traffic sign recognition, artificial neural network, artificial neural network training*

[1] University of Belgrade, Faculty of Mechanical Engineering, mmitrovic@mas.bg.ac.rs
[2] University of Belgrade, Faculty of Mechanical Engineering
[3] University of Belgrade, Faculty of Mechanical Engineering

Primena prepoznavanja saobraćajnih znakova na skaliranom modelu vozila

Apstrakt: *Popularnost autonomnih vozila je u poslednjih nekoliko godina značajnije porasla, pošto su sistemi koji omogućavaju određene autonomne funkcije na vozilu sve prisutniji. Studentima mašinskog i softverskog inženjerstva najpristupačniji način da steknu osnovna znanja o autonomnim vozilima predstavlja primena algoritama i sistema neophodnih za autonomnu vožnju na skaliranom modelu vozila. Ovakvi modeli, kao i u ovom slučaju, su opremljeni sistemima neophodnim za autonomnu vožnju, kao što su pogon na sva četiri točka, sistem oslanjanja, elektronski sistem upravljanja, računar kao „mozak" vozila i kamera. Cilj projekata, kao što je i ovaj, je da se osposobi vozilo za autonomnu vožnju na unapred predviđenoj stazi, poštujući sve saobraćajne propise i znakove (na primer, vozilo mora da se potpuno zaustavi ispred znaka stop). Kako bi ovo bilo moguće, neophodno je da vozilo „zna" koji se saobraćajni znak nalazi u njegovoj neposrednoj blizini, tj. neophodno je prepoznavanje saobraćajnih znakova. U ovom slučaju je korišćena veštačka neuronska mreža za prepoznavanje saobraćajnih znakova. U radu je opisan proces obučavanja odgovarajuće neuronske mreže.*

Ključne reči: *prepoznavanje saobraćajnih znakova, veštačka neuronska mreža, obučavanje veštačke neuronske mreže*

1. Introduction

Traffic sign recognition is very important, as obedience to traffic signs and all other traffic rules has a decisive influence on traffic safety. For this reason, systems that include traffic sign recognition are implemented in vehicles from various manufacturers (for example, *Mercedes-Benz Active Speed Limit Assist and Traffic Sign Recognition with a warning on Ford and Volkswagen cars)* (Stamenković D., 2022).

Traffic sign recognition can be divided into two basic tasks – traffic sign detection and traffic sign classification. Traffic sign detection implies a set of methods aimed to determine the traffic signs' presence (and position) on a video stream from a camera, while traffic sign classification can be defined as a set of methods for determining what sign is exactly on the video stream (Escalera, Barò, Pujol, Vitrià, & Radeva, 2011). For traffic sign detection and classification purposes, many algorithms have been developed.

Algorithms for traffic sign detection can be divided into algorithms based on the colour of traffic signs and algorithms based on the shape of traffic signs (Escalera et al., 2011).

The algorithms based on the colour of traffic signs perform a segmentation of the picture (which is a frame of a video stream) to find the segment which has the same colour as the traffic sign (red, blue, yellow, white and black colour). These algorithms are based on setting the colour threshold and are implemented in various colour spaces, such as *RGB, HSV, YUV, CIECAM97, YCbCr* and *CIELab* (Escalera et al., 2011), (Paul, Nandi, Saif, Zubair, & Shubho, 2018), (Gao, Podladchikova, Shaposhnikov, Hong, & Shevtsova, 2005), (Lopez, Fuentes, 2007). The main difficulty with traffic signs detection based on their colour is the change in lighting during the day.

Traffic sign detection algorithms based on the shape of traffic signs trace the edges of traffic signs. All edges which fit the shape of traffic signs (circle, rectangle, triangle, octagon), are potential traffic signs. Edge tracing, in these algorithms, is mainly done by Hough transformation. For this traffic sign detection method, variable lighting is not a problem but obscured signs by other objects are (Escalera et al., 2011).

To overcome the problems of both approaches (variable lighting and obscurity), hybrid methods are developed, where are both approaches combined for traffic sign detection (Escalera et al., 2011).

Traffic sign classification is mainly done by two methods: template-based and classifier-based. The template-based methods compare zones of interest (output of traffic sign detection) with traffic sign templates, while classifier-based methods represent some sort of deep machine learning algorithm. Artificial neural networks, which can perform traffic sign detection and classification simultaneously are included in the latter (Stamenković D., Popović V., & Blagojević I, 2017).

Traffic sign recognition has a long history. The first study about traffic sign recognition came out back in 1984 in Japan, while the first autonomous vehicle prototype was built in 1995. within a European project named PROMETHEUS powered by Daimler-Benz. The first vehicle with a built-in traffic sign recognition system was Opel Insignia released in 2009. Since July 2022 all new vehicles must have a built-in system with traffic recognition that can alert the driver about the presence of speed-limit signs.

2. About the scaled model and the track

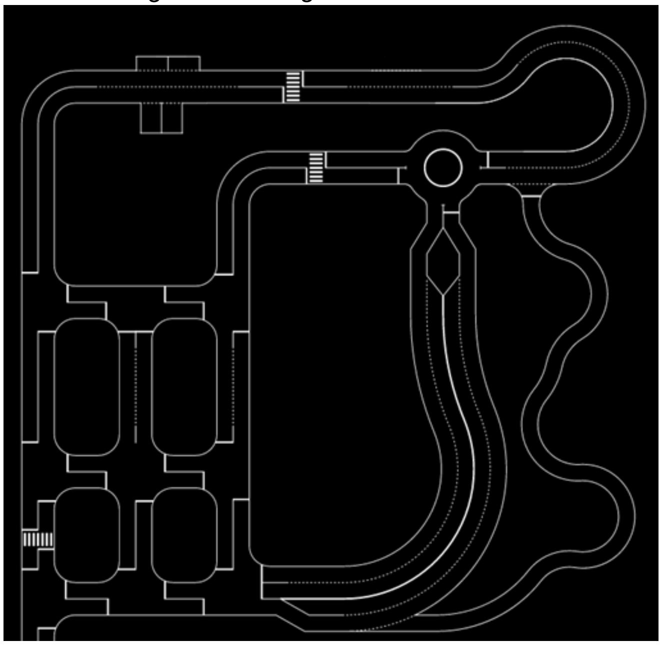
The model is on a scale 1:10 and is equipped with all systems necessary for autonomous driving, which include:

- Four-wheel drive powered by an electric motor,
- Suspension system,
- Steering system operated by servo motor,
- Raspberry Pi 4 computer,
- STM32 Nucleo microcontroller,
- Pi camera,
- BNO 05 – Inertia Measurement Unit,
- Encoder,
- Battery,
- H-bridge motor driver,

– DC/DC converters.

The track consists of straights, curves with various radii, four-way and tree-way intersections, roundabouts, highway (where vehicle should visibly accelerate), parallel and perpendicular parking spots and crosswalks. The configuration of the track is shown in Fig. 1.

Figure 1. Configuration of the track



3. Development of a traffic sign recognition algorithm

For a successful autonomous run, i.e., to obey all traffic signs, the vehicle must "know" which sign is in front of it. In this project, there are 9 possible traffic signs that can be found on the track (stop sign, priority sign, crosswalk sign, parking sign, roundabout sign, the start of a highway sign, end of a highway sign, prohibited direction sign, one-way traffic sign), which are selected according to track needs. Also, traffic lights, immobile obstacles, other vehicles (that can vary in colour) and scaled models of pedestrians can be found on the track, so the vehicle has to "recognise" those objects too. It is decided to perform the traffic sign recognition by the artificial neural network (ANN) because it is the most reliable way, but also the most hardware demanding. Other traffic sign recognition methods (detection by colour and template-based classification, for example) were also considered, but recognition of other vehicles and pedestrians would be difficult with those methods, due to irregular shapes and unpredictable colours, so they were thrown away.

Lane detection also can be achieved with ANN (Stamenković D., Popović V., & Vorotović G., 2016), which was another reason to choose ANN for traffic sign detection.

Every traffic sign recognition algorithm for input has a stream from the camera, which is divided into frames, each frame being input for ANN. The algorithms differ from each other in a few main ways, such as the way how ANN is called, frame processing before loading into the ANN, and ANN outputs reading and processing.

3.1. Implementation of existing ANN

In order to shorten the process, it is attempted to implement an ANN which is already available on the Internet. Instruction for implementation is found on the Digi-Key web page (Hymel S., 1995). In this case the COCO[4] SSD[5] Mobile Net V1 convolutional neural network (CNN). This CNN was only trained for recognition of stop sign and traffic light (CNN recognises only the presence of traffic lights in the frame, not which light is turned on). Although this CNN is not good enough for this purpose, it served as confirmation that traffic sign recognition via ANN is possible on the Raspberry Pi 4 computer.

3.2. Training ANN with dataset available on the Internet

Training can be expressed as finding the right values of weight and bias factors (Lee, & Song, 2019). The weight factor defines how the input of the nod affects its output (Alammar J., 2018), while the bias factor is a constant that is added to the sum of all inputs of the nod. The influence of these factors can be described with the formula:

$$y = f\left(\sum w_i x_i + b\right)$$

where:
y – nod output,
x_i – nod input,
w_i – weight factor,
b – bias factor,
f – activation function.

The right values of weight and bias factors are those which produce a low enough loss value (less than 0,01 [-]) (Lee et al., 2019). Loss is the difference between ANN output and real output and is calculated by various loss functions (Brownlee J., 2019). Training is done by the repetitive presentation of images of the object (which should be recognized) to the ANN. The process of training

[4] CNN trained on the “Common Objects in Context” (COCO) database.
[5] Single Shot Detector.

the ANN is very hardware demanding, and if it is not done by utilising the GPU it can be time-consuming. GPU utilisation for ANN training purposes can be enabled only by installing software provided by the GPU's manufacturer.

For the first attempt of training, the two image datasets were used, the German Traffic Sign Recognition Benchmark dataset (GTSRB) and the Belgium Traffic Sign dataset (BelgiumTS), both available on the respective official web page (Institut Für Nuroinformatik, 2010), (Timofte R., n.d.). These datasets consist of close-up traffic sign images, with various sizes, qualities and lighting. In this case, Mobile Net V2 1.00 244 architecture of ANN was used, which is similar to CNN mentioned in section 3.1. After training loss had a value of 0,008. When newly acquired ANN was tested on the vehicle, traffic signs were recognised only at very close range (less than 100 *mm*), which is not acceptable (the vehicle cannot react on time). This occurred due to the nature of the images in the dataset.

3.3. Training ANN with the personal dataset

To make recognition possible on larger distances it is necessary to train ANN with images where traffic signs are placed on large backgrounds. This kind of training can only be achieved if all images are annotated. This means that the position of an object of interest (traffic sign in this case) is marked on the image. Annotation is done with open-source software named LabelImg. In this case, training is done by code found in the Google Colaboratory notebook written by TensorFlow officials (The TensorFlow Authors, 2021). In this notebook, it is recommended that the dataset contains at least 80 images per object that should be recognised by ANN. Architecture Efficient Det – Lite0 was used for training, which is also recommended by the same notebook. This architecture is different to the previous two CNNs, so the code from the TensorFlow repository on GitHub (Daoust M., & Wang T., 2022) was used for recognition on the vehicle. The dataset was edited through 8 iterations and contained both individual images of traffic signs and images that were frames of videos taken by phone camera. Each version of the dataset contained more than 130 images per object that should be recognised.

Iteration No. 1 – The dataset in the first iteration contained individual photos with various backgrounds and also frames of videos taken mainly on dark backgrounds. This version of the dataset contained only images of traffic signs. After training, the loss had an acceptable value of 0,007. When ANN acquired by this training was tested on the vehicle, it was partially successful, it only recognised the traffic signs on a dark background (mainly on black). This was caused by the fact that the dataset was mainly made of images of traffic signs on dark backgrounds. Also, there was a mixing up of similar traffic signs, such as stop sign and parking sign (because of the letter P on them), parking sign and one-way traffic sign (same colour scheme), crosswalk sign and roundabout sign (white triangle on both signs), stop sign and prohibited direction sign (same colour scheme).

Iteration No. 2 – The second iteration of the dataset was essentially the same as the first one. Frames with different backgrounds, frames on which were similar traffic signs at the same time and frames with a red and green traffic light were added to the dataset. After training, no significantly better results were recorded. The ANN still mixed-up similar traffic signs and the colour of the traffic light was recognised from time to time.

Iteration No. 3 – In this iteration new approach is tested on only four traffic signs. This was due to the fact that annotation of all images consumes a lot of time. Images in this version of the dataset were only frames but taken on various backgrounds. After the training, the ANN performed much better, it only mixed-up stop sign and parking sign, when the parking sign is distanced more than 400 mm from the camera, which is acceptable.

Iterations No. 4 and No. 5 – Iteration No. 5 was the first try of pedestrian recognition. The dataset of this iteration contained only pictures of pedestrians and it was an unannotated dataset called Penn-Fudan available on the link (Wang L., Shi J., Song G., & Shen I., 2007). The ANN was so successful in the recognition after training, that it recognised pedestrians even when only the limb of the pedestrian was in the frame.

The dataset of iteration No. 5 was a combination of the previous two datasets. After the training with this dataset, ANN recognised traffic signs as in iteration No. 3 and pedestrians as in iteration No. 4.

Iteration No. 6 – In this iteration, the frames with all the rest traffic signs, immobile obstacle, and red and green traffic lights were added to the previous dataset. In this case, the videos are not only taken by phone camera, but also with the vehicle's Pi camera. Also, the images of other vehicles were added. These images were part of a dataset named Object Detection CrowdAI available on a link (Rakshit S., 2022). After the training, the value of the loss was only 0,002. The ANN performed very well, it did not have any difficulties with the recognition of traffic signs, traffic lights, immobile obstacles and pedestrians. Recognition of another vehicle was not successful. ANN's good performance on the vehicle was the result of the nature of the dataset, i.e., because the dataset contains frames from the vehicle's camera.

Iterations No. 7 and No. 8 – The only difference between datasets of iterations No.6 and No. 7 was that images from the Object Detection CrowdAI dataset were replaced with frames from both the phone and the vehicle's camera (the video of the vehicle was taken). Now the ANN recognised well even other vehicles.

In this version of the dataset, images of pedestrians were replaced with frames with the model of the pedestrian that can be found on the track. This was necessary because other persons than the model can be present on the track and in the camera's frame. This would make the vehicle perform a reaction on pedestrians even when it is not necessary (due to ANN performance mentioned in section 3.3.4.). After training, the ANN recognised all necessary objects within a distance of 600 mm from the camera.

4. Impact of ANN on overall vehicle performance

Besides traffic sign detection, a few other algorithms necessary for the vehicle's autonomous run are simultaneously executed on the vehicle's computer, such as lane keeping, steering angle calculations and decision-making by fuzzy logic. All those algorithms are more or less hardware-demanding, but the most demanding one is the ANN. When all mentioned algorithms are running on the vehicle, the voltage drop occurs due to Raspberry Pi's CPU overload, which also leads to its overheating. Also, ANN utilises CPU indirectly, with image pre-processing, i.e., with image resizing and normalisation. Image resizing is necessary due to the nature of ANN's input, while normalisation is not, so it is turned off in the main code.

The problem of performance decline with can be solved with a hardware upgrade. One possibility is to add one more Raspberry Pi 4 computer to the vehicle, which only would be used for traffic sign recognition purposes. Another solution is to add a Tensor Processing Unit (TPU), a processor intended for deep machine learning developed by Google. Table 1 shows the minimum and recommended number of TPUs for running ANN from the Efficient Det Lite ANN family, which was recommended by TensorFlow.

Table 1. Numbers of TPUs

Neural network	Minimum number of TPUs	Recommended number of TPUs
Efficient Det Lite0	1	1
Efficient Det Lite1	1	1
Efficient Det Lite2	1	2
Efficient Det Lite3	2	2
Efficient Det Lite4	2	3

One more solution is to use other methods of traffic sign recognition, while the position of the rest of the objects can be obtained from environmental servers, but this would make the vehicle less reliable.

5. Conclusions

Traffic sign recognition via ANN proved to be a reliable method, as the final version does not make mistakes with recognition within an acceptable distance from the camera.

However, the other methods of traffic sign recognition should be reconsidered, as ANN drains significant resources. This primarily refers to the vehicle's hardware overload, which can be solved by hardware upgrade.

Also, a high-performance computer is required for ANN training process purposes, so the process would not consume a huge amount of time. Besides that, a large amount of time is required to prepare the dataset, but at another hand, it is relatively easy to add more objects that should be recognised.

For ANN implementation for traffic sign recognition on serial vehicles, it is necessary to acquire images of all existing traffic signs with different lighting (at different parts of the day, at night), images where traffic signs are obstructed by another object and images where traffic signs are at a bad angle relative to the camera. The case scenario is that the training photos are recorded with the same camera with which the recognition will be performed and in the same image quality.

References

Alammar J. (2018). *Weight (Artificial Neural Network)*. Retrieved from http://jalammar.github.io/visual-interactive-guide-basics-neural-networks/ in September 2022.

Brownlee J. (2019). *Loss and Loss Function for Training Deep Learning Neural Networks. Retrieved from* https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/ in September 2022.

Daoust M., & Wang T. (2022). *TensorFlow Lite Python object detection example with Raspberry Pi.* Retrieved from https://github.com/tensorflow/examples/tree/master/lite/examples/object_detection/raspberry_pi in February 2022.

Escalera S., Barò X., Pujol O., Vitrià J., & Radeva P. (2011). *Traffic-Sign Recognition Systems.* London, England: Springer.

Gao X., Podladchikova L., Shaposhnikov D., Hong K., & Shevtsova N. (2005). Recognition of traffic signs based on their colour and shape features extracted using human vision models. *Journal of Visual Communication and Image Representation, 17, 675-685. ELSEVIER.*

Hymel S. (1995). *How to Perform Object Detection with TensorFlow Lite on Raspberry Pi. Retrieved from* https://www.digikey.com/en/maker/projects/how-to-perform-object-detection-with-tensorflow-lite-on-raspberry-pi/b929e1519c7c43d5b2c6f8 9984883588 in December 2021.

Institut Für Nuroinformatik (2010). *German Traffic Sign Recognition Benchmar.* Retrieved from https://benchmark.ini.rub.de/

Lee H., & Song J. (2019). Introduction to convolutional neural network using Keras; an understanding from statistician*. Communications for Statistical Applications and Methods Journal, 26(6), 591-610.*

Lopez, L.-D., Fuentes, O. (2007). Color-Based Road Sign Detection and Tracking. In: Kamel, M., Campilho, A. *Image Analysis and Recognition. ICIAR 2007, 1138–1147. Lecture Notes in Computer Science, 4633, Springer.*

Paul P., Nandi D., Saif S., Zubair K., & Shubho S.-A. (2018). Traffic Sign Detection Based on Colour Segmentation of Obscure Image Candidates: A Comprehensive Study. *International Journal of Modern Education And Computer Science, 6, 35-46.*

Rakshit S. (2020). Udacity Car Dataset CrowdAI. Retrieved from https://www.kaggle.com/datasets/soumikrakshit/udacity-car-dataset-crowdai in April 2022.

Stameković D., (2022). *Autonomous motor vehicle control model* (Doctoral Dissertation). University of Belgrade, Belgrade, Serbia.

Stamenković D., Popović V., & Blagojević I., (2017). *A brief review of strategies used to control an autonomous vehicle.* Paper presented at the Second Maintenance Forum 2017, Bečići, Montenegro.

Stameković D., Popović V., & Vorotović G., (2016). *Lane detection algorithm using image processing for autonomous vehicle model.* Paper presented at Euromaintenance 2016, Athens, Greece.

The TensorFlow Authors (2021). *Model Maker Object Detection for Android Figurine.* Retrieved from https://colab.research.google.com/ github/khanhlvg/ tflite_raspberry_pi/blob/main/object_detection/Train_custom_model_tutorial.ipynb#scrollTo=UNRhB8N7GHXj in February 2022.

Timofte R. (n.d.), *Belgium Traffic Sign Dataset.* Retrieved from https://btsd.ethz.ch/shareddata/ in January 2022.

Wang L., Shi J., Song G., & Shen I. (2007). *Object Detection Combining Recognition and Segmentation.* Retrieved from https://www.cis.upenn.edu/~jshi/ped_html/ in March 2022.