https://doi.org/10.21122/2227-1031-2020-19-1-85-88

UDC 349

Imitating a Safe Human Driver Behaviour in Roundabouts Through Deep Learning

Á. S. J. Cervera[1], F. J. Alonso[2], F. S. García[1], Á. D. Alvarez[1]

[1]Universidad Politécnica de Madrid, ETSI de Sistemas Informáticos (Madrid, Kingdom of Spain),
[2]Universidad Politécnica de Madrid, Escuela Técnica Superior de Ingenieros Industriales (Madrid, Kingdom of Spain)

Abstract. Roundabouts provide safe and fast circulation as well as many environmental advantages, but drivers adopting unsafe behaviours while circulating through them may cause safety issues, provoking accidents. In this paper we propose a way of training an autonomous vehicle in order to behave in a human and safe way when entering a roundabout. By placing a number of cameras in our vehicle and processing their video feeds through a series of algorithms, including Machine Learning, we can build a representation of the state of the surrounding environment. Then, we use another set of Deep Learning algorithms to analyze the data and determine the safest way of circulating through a roundabout given the current state of the environment, including nearby vehicles with their estimated positions, speeds and accelerations. By watching multiple attempts of a human entering a roundabout with both safe and unsafe behaviours, our second set of algorithms can learn to mimic the human's good attempts and act in the same way as him, which is key to a safe implementation of autonomous vehicles. This work details the series of steps that we took, from building the representation of our environment to acting according to it in order to attain safe entry into single lane roundabouts.

Keywords: deep learning, Neural Networks, driver behaviour, roundabouts, imitation learning

For citation: Cervera Á. S. J., Alonso F. J., García F. S., Alvarez Á. D. (2020) Imitating a Safe Human Driver Behaviour in Roundabouts Through Deep Learning. *Science and Technique*. 19 (1), 85–88. https://doi.org/10.21122/2227-1031-2020-19-1-85-88

Моделирование безопасного поведения водителя на перекрестках с помощью глубинного обучения

А. С. Х. Сервера[1], Ф. Х. Алонсо[2], Ф. С. Гарсиа[1], А. Д. Альварес[1]

[1]Мадридский политехнический университет, Высшая техническая школа компьютерных систем инженерии (Мадрид, Королевство Испания),
[2]Мадридский политехнический университет, Высшая техническая школа промышленной инженерии (Мадрид, Королевство Испания)

Реферат. Кольцевые транспортные развязки обеспечивают безопасное и быстрое движение, а также ряд экологических преимуществ. Но водители, придерживающиеся ненормативных правил поведения при вождении по ним, могут вызвать проблемы с безопасностью, что приводит к несчастным случаям. В статье предлагается способ обучения водителя автономного транспортного средства с целью обеспечения правильного и безопасного поведения при въезде в кольцевую транспортную развязку. Поместив несколько камер в транспортное средство и обработав видеозапись

Адрес для переписки
Альваро Сан Хуан Сервера
Мадридский политехнический университет,
Высшая техническая школа компьютерных систем инженерии
Карретера де Валенсия (А3) 7-й км,
28040, г. Мадрид, Королевство Испания
Тел.: 0034 91 336-53-00
alvaro.sanjuan@upm.es

Address for correspondence
Alvaro San Juan Cervera
Universidad Politécnica de Madrid,
ETSI de Sistemas Informáticos
Carretera de Valencia (A3) km. 7,
28040, Madrid, Kingdom of Spain
Tel.: 0034 91 336-53-00
alvaro.sanjuan@upm.es

видеопотоков с помощью ряда алгоритмов, включая и машинное обучение, можно получить представление о состоянии окружающей среды. Затем используется другой набор алгоритмов глубокого обучения для анализа данных и определения наиболее безопасного пути кругового движения с учетом текущего состояния окружающей среды, включая ближайшие транспортные средства с их предполагаемым местоположением, скоростью и ускорением. Анализируя многочисленные примеры безопасного и опасного поведения водителя во время движения по кольцевой транспортной развязке, предлагается второй набор алгоритмов, который позволяет моделировать правильное поведение водителя, что и является главным условием безопасного применения автономных транспортных средств. В статье подробно описываются все этапы работы, начиная от построения рассматриваемой окружающей среды и заканчивая соответствующим поведением в зависимости от ситуации, что позволяет обеспечить безопасное движение в кольцевой развязке с одной полосой движения.

Ключевые слова: глубинное обучение, нейронные сети, поведение водителя, кольцевая транспортная развязка, имитационное обучение

Для цитирования: Моделирование безопасного поведения водителя на перекрестках с помощью глубинного обучения / А. С. Х. Сервера [и др.] // *Наука и техника*. 2020. Т. 19, № 1. С. 85–88. https://doi.org/10.21122/2227-1031-2020-19-1-85-88

Introduction

Computer Vision is a field that is undergoing a tremendous evolution. Over the last few years Artificial Neural Networks (from now on will be referred also as ANN or NN) are being used to process images due to their high accuracy and fast performance, easily outperforming more traditional approaches in benchmarks, while being relatively easy to implement and reusable.

The goal of this investigation is delivering a system capable of determining whether or not is it safe to enter a roundabout given the current traffic conditions. In order to achieve that, we use the footage from a single camera installed in our vehicle instead of relying on LIDAR (Light Detection and Ranging) data, which would be a more traditional approach.

LIDAR is a surveying method that used a series of laser beams to measure the distance to a target by tracking the time the light takes to bounce back to the sensor. Having one of these sensors on top of our vehicle spinning multiple times per second (normally at a rate of 20 spins per second), creates a 3*D*-scan of the environment in real time. The downside to this approach is that this kind of sensors are expensive and not as common as a regular camera (although prices are coming down in recent years).

Traditional cameras are very common type of sensor, so being able to rely on its data and leaving aside the expensive LIDAR, will enable future investigation and implantation in autonomous vehicles to become much more approachable. Doing this task should be feasible as a human subject is capable of performing the same task using just one eye, relying on years of experience perceiving environments.

From the images recorded by this monocular camera (which we will call "red-green-blue images, or rgb"), we use multiple Artificial Neural Networks to perform various operations, including vehicle detection, distance estimation, environment recreation and motion tracking to obtain their relative speeds and accelerations. All this work has been done using Carla Simulator [1] for the testing environment and Keras as the Deep Learning Framework.

Environment perception with Neural Networks

Artificial Neural Networks are systems that resemble the biological neural networks in our brains (although they're not identical). These systems can "learn" how to perform a task given a correct design (called architecture) and a learning process (commonly referred to as training). When these systems complete their learning process, they build a model that we can use to perform the task that they were created for.

Their usage in image recognition is becoming popular in the last ten years. They can be used for trivial tasks like, for example, analyzing a series of images and detecting which ones contains dogs. They are widely used in ways out of the scope of this investigation, such as spam detection, shopping recommendations, voice recognition, photo enhancement and creating complex virtual assistants.

In this project, we are using various Neural Networks working together to attain our goal. First, we feed each of the frames captured by the camera through an object detection network. Instead of designing a network architecture from scratch, we've decided to choose YOLO (version 3 [2]) as our object detection system. The reasons for this are: YOLO [3] usage is very extended throughout numerous projects and it has been proved to provide a good ratio between accuracy and performance. Keep in mind that this system is going to be used in real time on a reduced hardware environment, so high performance is absolutely necessary. In our test, using images of size 416×416, we can do the object detection task at 28 frames per second, which is well above our target 20 frames per second.

There are plenty of datasets available to use with YOLO, even pre-trained models freely available to use. In order to maximize its performance and minimize the error of the predictions, we have crafted a custom dataset using images extracted from Carla, some examples can be seen in Fig. 1.

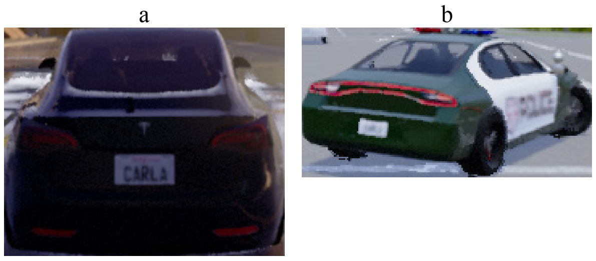

Fig. 1. Sample car images taken from our dataset

The dataset contains just four classes (types of objects that we want to identify): vehicles (including cars, buses, trucks, bikes and motorbikes), traffic lights, traffic signs and pedestrians, which is much less than the twenty general-purpose classes used in the default dataset. Fine tuning the dataset and model to our needs creates a smaller, lighter model which results in faster and more precise predictions. Keep in mind that the only class relevant to this project is the "vehicle" class, the rest of the classes are included in the dataset just for future use.

This is the first and most important NN in our system. The raw data from the camera is passed through it and the process begins. The network detects the objects in the current frame and returns the list of the detected bounding boxes. A bounding box has the following attributes: the coordinates of the object's upper left and lower right corners in the image, the object's class (vehicle, traffic light, traffic sign or pedestrian) and the confidence level of the prediction. We set a minimum confidence level threshold to validate the predictions and store all the detected bounding boxes above that threshold to further analyze them.

Our next step is motion tracking. There are two possible scenarios for each object in the frame: being a new object or being an object previously detected on another frame. We check for consistency between the new bounding boxes and the bounding boxes we had already detected. A process of trying to pair each new object with an object from the previous frame begins here: we check every bounding box of the current frame against the ones we had from the previous frame and try to pair the ones that offer the best overlap. For each object, if the best overlap ratio is lower than a threshold, we consider that the new bounding box was not on the scene before this frame, which means that the object is new, so we add this object to the scene. Conversely, if the overlap meets our threshold, the object might be one that we are already tracking. The next step to decide if it's a new object or not is to check color consistency between the two objects, the new one and the candidate from the previous frame. Only if the colours are similar we consider them to be the same object.

Here's when a second Neural Network starts to work. As we saw earlier, one important step of the process is to estimate the distance between our vehicle and the rest of the vehicles. Doing this with LIDAR would have been much easier, as it gives you the distance to all surrounding objects with a high precision and low error rate. Getting that measurements with a stereo camera setup is also feasible, as that is the way the human brain can estimate the depth of the different points of the surrounding environment. The challenge is getting this data with just one camera image as we lack a lot of the information that makes the estimation possible.

A lot of the work done on this step is based on the work done in this project [4], but we are using another network architecture to try to speed up the process while losing the minimum possible precision.

We built a dataset consisting on standard pictures captured with a camera and their corresponding depth maps. In the real world, you would need a LIDAR to get the actual measurements, but Carla provides us with tools to get this data. After getting more than two thousand pairs of standard and depth images, we train a new network to construct a depth map from any given image. Using a mean squared error metric, we get an error of 0.0058 on the training set and 0.0087 which means that our predictions should be, by average, within a 9.73 % margin of the real value.

Fig. 2 shows an example of an image, its real depth map and the predicted depth map created by our network.

Having this second network creating the necessary depth map makes possible the next step: estimating the distance between our vehicle and each one of the detected bounding boxes. This is a straightforward process: first, we get the bounding box of each object (the bounding box contains the pixel coordinates of the top-left and bottom-right corners of the object in the rgb image), we then get the average colour for that region in the depth map, (discarding atypical values in order to reduce error). Then using linear regression, we map that pixel's rgb value to a numeric value corresponding to the distance. Every object in the scene contains the history of the object's estimated distances as well as their associated timestamps.

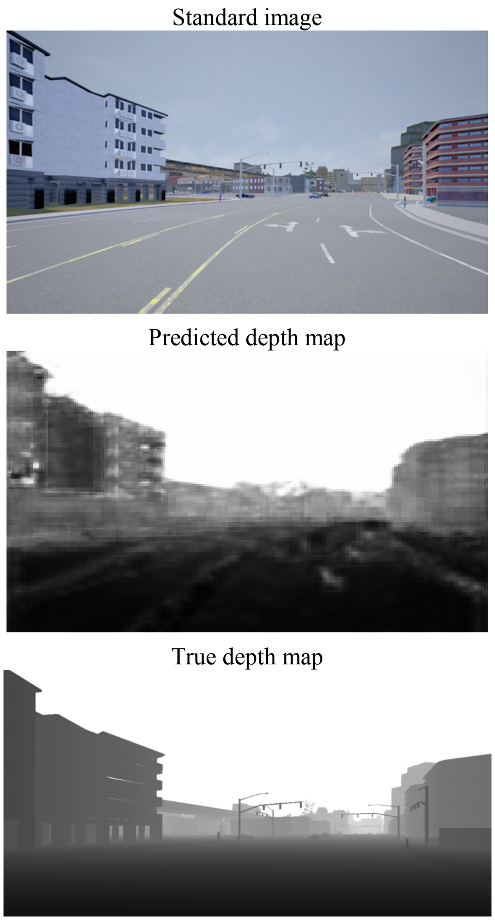

Fig. 2. Real and predicted depth maps comparison

If this is not a new object, we should have an old distance estimate accompanied by the timestamp of that measurement. Having this, we can calculate the delta distance and the elapsed time between the two instants and estimate the current relative velocity of the object (related to our vehicle). In the same way, we can estimate the current relative acceleration if we have multiple velocities recorded for the object.

Decision making with a Neural Network

Once our system is able to generate all the data we need, we can put it together into our practical case: determining whether is it safe or not to perform the incorporation into the roundabout. As we approach the roundabout, we are gathering information about the rest of the vehicles. This is where the last NN begins doing its job, but first we have to train it.

The training process is as follows: a human driver watches the environment through the camera feed and, he inputs to the training data if a incorporation would be safe given the conditions in that instant. This time, the output of the network would be a simple “yes” or “no”. The user’s elections are recorded with their time stamps, as well as the video feed from the camera. Then we use this data as training data, feeding it into our NN. Given enough data and time to compute it, out network can associate scene features to one of the output labels (yes/no) and learn how to make the predictions.

CONCLUSIONS

1. We have created a reusable framework that sets the foundation for future projects. As this system requires no specialized hardware aside from a regular camera and a computer, we can easily replicate the job done in the simulator in the real world, without the need of spending money on expensive sensors or performing irreversible modifications in a vehicle.

2. We have achieved depth estimation and 3*D*-point reconstruction with a mono camera with sufficient precision, but further refinements to the system will be done. Right now, it doesn’t take into account important factors such as camera FOV, and camera height. Taking these factors into account will improve our estimations and allow us to put the system to work on different configurations, which is impossible right now without losing accuracy.

3. Object recognition will continue to improve. Future versions of current architectures or newly developed architectures from now on will benefit this project, as its modular nature allows us to easily swap and integrate components into our system.

4. Motion tracking should be improved in the future, as it is key to speed and acceleration estimations. Using stereo cameras doesn’t add much cost to the required hardware and will give us a boost in accuracy, which will improve the data gathered by the system, although, in its current state, we can replicate the desired behavior safely.

REFERENCES

1. Dosovitskiy A., Ros G., Codevilla F., Lopez A., Koltun V. (2017) *CARLA: an Open Urban Driving Simulator*. Available at: https://arxiv.org/abs/1711.03938.
2. Redmon J., Farhadi A. (2018) *YOLOv3: an Incremental Improvement*. https://arxiv.org/pdf/1804.02767.pdf.
3. Redmon J., Divvala S., Girshick R., Farhadi A. (2015) You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2016.91.
4. Casser V., Pirk S., Mahjourian R., Angelova A. (2019) Depth Prediction without the Sensors: Leveraging Structure for Unsupervised Learning from Monocular Videos. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33, 8001–8008. https://doi.org/10.1609/aaai.v33i01.33018001.

Received: 08.10.2019
Accepted: 10.12.2019
Published online: 31.01.2020