Neurocomputing 594 (2024) 127874

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Autonomous driving in traffic with end-to-end vision-based deep learning

Sergio Paniego *, Enrique Shinohara, JoséMaría Cañas

Dep. Teoria de la Señal y de las Comunicaciones y Sistemas Telemáticos y Computación, Universidad Rey Juan Carlos, Madrid, Spain

ARTICLE INFO

Communicated by J. Hu

Keywords:
End-to-end autonomous driving
Imitation learning
Deep learning
Lane-following

ABSTRACT

This paper presents a shallow end-to-end vision-based deep learning approach for autonomous vehicle driving in traffic scenarios. The primary objectives include lane keeping and maintaining a safe distance from preceding vehicles. This study leverages an imitation learning approach, creating a supervised dataset for robot control from expert agent demonstrations using the state-of-the-art Carla simulator in different traffic conditions. This dataset encompasses three different versions complementary to each other and we have made it publicly available along with the rest of the materials. The PilotNet neural model is utilized in two variants: the first one with complementary outputs for brake and throttle control commands along with dropout; the second one incorporates these improvements and adds the vehicle speed. Both models have been trained with the aforementioned dataset. The experimental results demonstrate that the models, despite their simplicity and shallow architecture, including only small-scale changes, successfully drive in traffic conditions without sacrificing performance in free-road environments, broadening their area of application widely. Additionally, the second model adeptly maintains a safe distance from leading cars and exhibits satisfactory generalization capabilities to diverse vehicle types. A new evaluation metric to measure the distance to the front vehicle has been created and added to Behavior Metrics; an open-source autonomous driving assessment tool built on CARLA that performs experimental validations of autonomous driving solutions.

1. Introduction

The advances in autonomous vehicle development are in high demand as society grows, raising the need for safer and more convenient transportation, reducing accidents, and alleviating traffic congestion [1]. Releasing human drivers from the chore of driving is also a goal. Leading companies such as Waymo and Tesla are actively developing and deploying intelligent vehicles with these capabilities, including autonomous taxi services in various cities [2,3]. To keep track of the improvements in these technologies, SAE International (Society of Automotive Engineers) introduced a classification standard in 2014 that has been widely adopted. The levels of autonomy, outlined in SAE International's J3016 article [4], range from total driver control with passive safety systems (Level 0) to fully autonomous driving without driver supervision (Level 5).

There are two major approaches for generating solutions in autonomous driving, modular and end-to-end. First, the modular approach combines smaller independent components that communicate with each other [5]. Each component is responsible for one particular task: perception, planning, mapping, control... [6]. Moreover, modules can be developed to detect crucial environmental elements such as traffic lights, pedestrians, or other vehicles to prioritize safety and respond accordingly. This is the most widespread approach for solving autonomous driving problems, in part, thanks to its flexibility.

Second, the end-to-end approach directly generates outputs from the raw input data in a single forward pass. It delegates tasks like detection, tracking, path planning [7], and control to a machine learning model. For instance, a deep learning (DL) model. This model has the responsibility of processing inputs, interpreting and analyzing them, and making informed decisions to control the vehicle without relying on other modules [8–13].

In this study, we follow the second approach using deep and imitation learning to train an end-to-end neural network model. The ego vehicle is equipped with a controller, the vehicle's central processing unit. It generates control commands based on the sensory input data to make the car steer, accelerate, and brake as needed. The sensory input consists of real-time visual data gathered from a frontal camera attached to the car. Inside the controller, a deep learning model will be responsible for performing the tasks of perception and decision-making from the perceived information, hence the end-to-end approach. Moreover, the vehicle will need to respond appropriately when encountering other vehicles in its lane. This may involve bringing the vehicle to a complete stop to wait until the obstacle is cleared, avoiding it [14].

* Corresponding author.
E-mail addresses: sergio.paniego@urjc.es (S. Paniego), e.shinohara.2021@alumnos.urjc.es (E. Shinohara), josemaria.plaza@urjc.es (J. Cañas).

https://doi.org/10.1016/j.neucom.2024.127874
Received 8 November 2023; Received in revised form 23 February 2024; Accepted 13 May 2024
Available online 17 May 2024

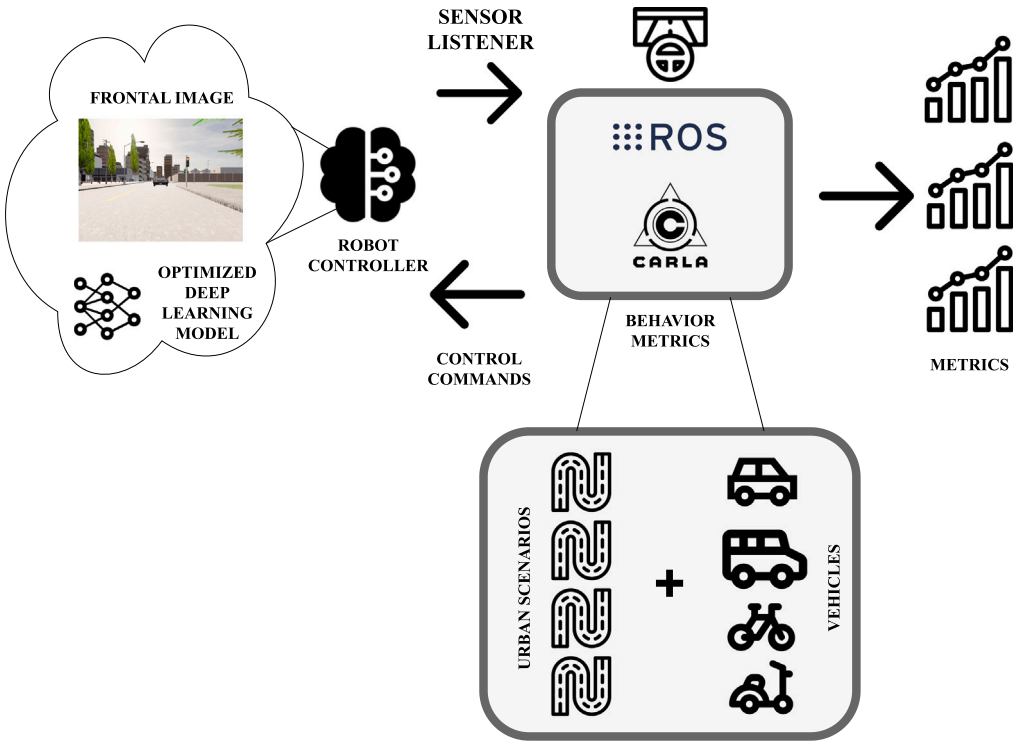

Fig. 1. Behavior Metrics evaluation software tool architecture with different urban scenarios and vehicles.

The deep learning model inside the vehicle's controller will be trained to consider this

Inside imitation learning, one of the principal methods is known as behavior cloning [15]. This method is employed in robotics and machine learning to teach an agent or system to mimic a desired behavior by learning from examples [16]. In autonomous driving, there are also some examples of successful use of imitation learning-based approaches [17–19]. These approaches usually rely on vision-based end-to-end solutions, which are still limited [20] but they are undergoing an important development lately [21].

Advanced technologies often combine imitation learning with reinforcement learning (RL) or other techniques, creating hybrid systems capable of handling complex tasks [22–26]. Other situations, like road intersections have also been successfully tackled using deep RL [27].

PilotNet [28] is an important end-to-end model in the field of vision-based end-to-end control for autonomous driving. This network receives visual data from a camera as input and generates steering control commands as outputs. We use this model as a baseline for our work, introducing refinements.

Simulation plays an important role in data collection, testing, and validation in the autonomous driving field. SUMO [29] is an open-source software simulator that mimics realistic traffic situations, enabling the evaluation of traffic efficiency and management strategies. CARLA Simulator [30], another open-source software, offers a customizable virtual environment for testing and validating autonomous driving algorithms, providing multiple vehicle and sensor models for generating synthetic sensor data. Its versatility and customization allow easy integration with other common tools in robotics like ROS [31], which facilitate the creation of applications [5] and easier transfer from simulation to real-world scenarios. TORCS [32] is also a relevant simulator with a focus on autonomous driving.

The autonomous driving field already presents a variety of curated datasets for a diverse range of tasks involved in the domain. For example, KITTI [33] and nuScenes [34] for visual perception or comma AI [35] for lane-following (lane-keeping) in a real-world scenario. These datasets, along with others [36–38], have become benchmarks for researchers and developers in the field. In the present work, we generate a dataset with three complementary versions for lane-keeping considering traffic in simulation that is extracted from expert agent demonstrations. We collected this dataset since the currently provided ones by the community are not directly suitable for the particular task of this work. More details about this are provided in Section 2.1.

The research question faced in this paper examines whether a visual end-to-end DL imitation learning shallow model can successfully drive an autonomous car to follow its lane in urban scenarios without colliding with other vehicles in the lane. In certain scenarios, such as when the hardware is constrained (e.g. edge devices), having a small, fast, and reliable model is essential [39]. Consequently, we prioritize simplicity in our models. We introduce and experimentally compare two variants of the PilotNet model, called PilotNet* and PilotNet**. Both variants can drive in simulation in the state-of-the-art simulator CARLA, keeping the lane and the latter also managing traffic conditions with front vehicles. These models generate throttle, brake, and steering control from the visual information provided by a frontal camera installed onboard the vehicle. The models are trained using an imitation learning procedure from a supervised dataset, generated from data collected from an expert agent driving in a training scenario. Adding the throttle and brake to the baseline, the model can drive following the lane in urban scenarios, also considering turns. Including the speed into the model, the vehicle can also negotiate scenarios with leading vehicles, stopping when encountering a vehicle in front and resuming driving when the short-term path is clear again. The models are validated experimentally under different conditions in test urban scenarios, using a variety of front vehicles and proving the generalization of the model to never-seen situations. The online experimental validation is conducted using the comparison software tool Behavior Metrics (see Fig. 1 for architectural details), which we have updated including evaluation metrics suitable for the presented problem of follow-lane in traffic situations. These metrics complement the common offline evaluation metrics used in machine learning, adding a broader context to each evaluated model's benefits and possible limitations. We consider online evaluation as the one conducted in simulation

Fig. 2. Detail of included vehicles in each dataset version.

and offline evaluation as the one conducted comparing the model to supervised data. We keep other autonomous driving tasks out of the scope of the present work, such as negotiating road intersections or considering traffic signals.

One contribution of this paper is the proposal of slight modifications to the shallow baseline PilotNet model, which demonstrates that with small-scale changes, it is possible to expand its application area widely. They allow the autonomous car to deal with different ahead vehicles in the same lane successfully, including those it has never encountered before, slowing down or stopping before them, resuming the movement, and following them when the safety standards are satisfied. We experimentally validate this assumption extensively in Section 4 and its generalization to new scenarios. Another contribution is the new fine-grain metric in the assessment tool that measures the distance to other vehicles based on the data extracted from CARLA, which gives a better intuition about how the robot controller behaves. We provide all models, architectures, and datasets as open-source, along with the comparison software tool [40] for validation and extension, which are also contributions. As a result, researchers may leverage this advancement to extend or develop an enhanced version of the models, architectures, dataset, or software.

2. Imitation learning for driving in traffic

In this section, we will discuss the three primary components of our work. Including the generated dataset versions for imitation learning, the modifications of the baseline deep learning model PilotNet created for this work, and the training procedure followed in the development of the final models.

2.1. Dataset and versions

The supervised dataset is collected on an urban scenario (Town02) of the CARLA simulator [30] using an imitation learning approach. The expert agent is an integral component of the simulator with access to privileged simulator data and it bases its behavior on hand-crafted rules. It is set to follow a specific route keeping the lane and covering the urban scenario while the visual data is generated by the camera, and the corresponding control commands are recorded. The agent's maximum velocity is approximately 30 km/h.

The route includes turning situations and some encounters with front vehicles but without possible intersection situations or consideration of traffic lights/signals, which are not in the scope of this work. For simplicity, we focus only on urban scenarios rather than including highways. Nevertheless, the ideas presented here are also applicable to these and other possible scenarios. Through this process, we generated a dataset of 140K images along with supervised demonstration data. Approximately, 47% of them are images containing other vehicles present in it.

We include three dataset versions that complement each other. The first, Traffic-0, does not consider any traffic factors. The second version, Traffic-1, includes the former and traffic examples with only one type of front vehicle, a typical small urban car. The third version, Traffic-6, encompasses both previous versions and examples with five other vehicle types including two vans and three urban cars of different sizes and colors (see Fig. 2. for details about which vehicles are included in each dataset version).

For training and testing, the ego vehicle is equipped with an on-board RGB camera that is oriented forward, capturing images with dimensions of 480 pixels in height and 650 pixels in width (480×650).

The image that the deep learning model processes contains a significant amount of information, but from that frontal image of the urban scenario, not all the content is relevant. We crop the image to reduce its complexity. By cropping the image to exclude elements such as the sky and buildings, which are not needed for generating control commands in this simplified context, unnecessary data is effectively removed. In addition to the cropping, the image is compressed. As a result, the input images are reduced to a size of 66 pixels in height and 200 in width, representing only about 4% of the original image size. This preprocessing optimizes the data and reduces the dataset size facilitating faster training.

2.2. Baseline model and its modifications

The two models proposed for this project are variations of the baseline PilotNet model [28] (see Fig. 3 for details about each architecture), built using the Tensorflow [41] framework. The baseline PilotNet network consists of 9 layers which include a batch normalization layer, 5 convolutional layers, and 3 fully connected layers. The first part of the model is responsible for extracting features from the input visual

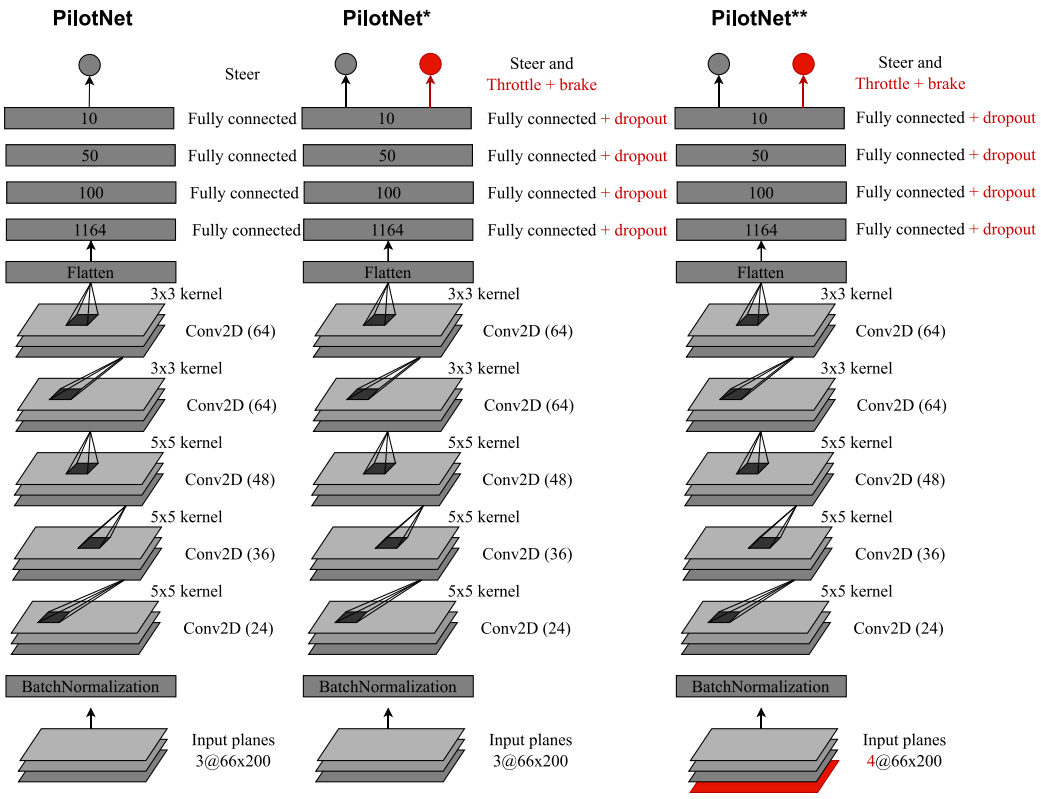

Fig. 3. PilotNet baseline model and its variations PilotNet* and PilotNet**. In red, introduced changes are highlighted.

data and the latter part generates the final control commands from that extracted features.

We introduce enhancements to the baseline model, which are specific to the current project. The primary motivation behind these modifications is to investigate whether a shallow, established model can significantly broaden its applicability through minor enhancements.

In the first variant, called PilotNet*, an extra output is added alongside the steering command. This new output generates control signals for the throttle and brake. The architecture also includes regularization techniques, specifically batch normalization and dropout layers [42] with a 0.1 rate. The batch normalization layers were already proposed in the original PilotNet work while the dropout layers are inserted between the final dense layers of the PilotNet* model to make it more suitable for supervised data and prevent overfitting. The dropout rate is determined through experimental validation. Increasing it further has been found to yield catastrophic results. The objective behind the development of this model lies in determining whether a unified architecture can effectively handle multiple outputs of varied natures, such as throttle, steering, and brake, thereby significantly enhancing the overall behavior of the model.

In the second model, called PilotNet**, we build upon the PilotNet* model by including the vehicle's velocity for each time step. This velocity is added to the input alongside the image, resulting in a final input image of shape (66, 200, 4). The extra channel is uniformly filled with the normalized speed, scaled between 0 and 1. In real cars, this speed may be taken from an onboard speedometer. The rationale behind this addition is to ascertain whether a model equipped with knowledge of its speed outperforms a model lacking such information and to examine its impact on the system's performance in the presence of other vehicles in the scenario. In Fig. 3 details about each architecture are displayed, showing their differences.

2.3. Training procedure

During the training procedure of each of the models, we introduce a dataset pre-processing stage. In this stage, the data is transformed and prepared for training, including regularization techniques such as data augmentation. We also include early stopping as a regularization technique. In this case, we include data augmentation techniques for adjusting brightness and contrast, modifying the image colors such as hue, saturation, and value components, adding blur, and simulating various weather conditions like rain, snow, fog, sun flare, and shadows. All these techniques are common in data augmentation frameworks, like Albumentations [43] which is the one used here.

The generated dataset is unbalanced, including a lot of straight-lane samples where the steering is not relevant. This situation is common in autonomous driving and some techniques have been already presented to address this issue, like Dagger [44]. To overcome these situations, we introduce oversampling of turn situations and generate more data for particularly relevant urban areas, such as curves. By doing so, the dataset is more balanced.

For training, the hardware used includes an NVIDIA 3060 GTX GPU. This hardware is the same used in the experimental validation of the presented models.

Table 1 presents the Mean Squared Error (MSE) and the Mean Absolute Error (MAE) of the three proposed models. The loss function used for training is MSE. While these metrics offer insight into how the models have been trained and their capabilities, they do not suffice to draw a definitive conclusion regarding the overall performance and generalization capabilities of each model in a robotics closed-loop problem such as this one. To further expand the scope of assessment, the use of online evaluation metrics is crucial. Behavior Metrics [45] assists in this endeavor by examining in detail the behavior of each model in real-time scenarios and providing detailed insights into their respective performance and adaptability.

To gain a better understanding of PilotNet**'s behavior and to effectively spot any unusual behaviors, we utilize activation heat maps, a technique explored in prior studies [46]. This visualization method allows for an understanding of where the neural network places its attention, providing useful insights into the decision-making process.

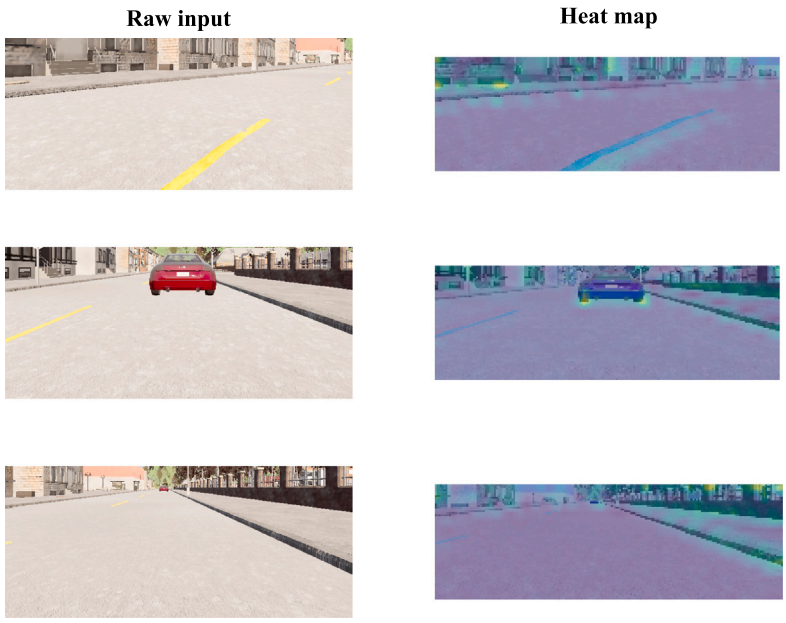

Fig. 4. Activation heat map visualization from the last CNN layer of PilotNet** model.

Table 1
Offline evaluation measures of the model's performance.

Model Training dataset	PilotNet* Traffic-1	PilotNet** Traffic-1	PilotNet** Traffic-6
MAE	0.04685	0.04121	0.03842
MSE	0.01138	0.00930	0.00706

In this case, we obtained the activation heat map using the Grad-CAM [47] algorithm for PilotNet** trained with Traffic-6. Fig. 4 visually represents the more relevant parts detected by the activations of the last convolutional layer of the network. PilotNet** recognizes the lane markings and edges and highlights the wheels of the front car. The attention placed on the wheels is a result of training with diverse vehicles, enabling effective generalization and reducing the risk of collisions with any on-road vehicle.

3. Assessment of autonomous driving in traffic

In this section, we describe the assessment framework created for this project. In the evaluation of deep learning models, there are common metrics like MSE that are used for model validation. For the particular case of autonomous driving, these metrics could be not enough to understand the overall performance of the model in an online evaluation. For example, if we look at Fig. 4 we can have a good understanding of which part of the visual input is more relevant for the control decision; but we lack some context about how the model performs driving the car in a test scenario with test front vehicles.

Quantitative evaluation allows us to measure and analyze data to assess model performance, effectiveness, and quality. More importantly, this evaluation allows us to compare different neural models and determine whether changes in training, dataset, or model lead to performance improvements or not. CARLA provides valuable data about vehicle performance in specific situations, aiding in evaluating our models. It offers data such as:

1. **Collisions**: number of bumps occurred between our ego vehicle and any actor in the simulation, whether it is another vehicle, a building, or a prop in the environment.
2. **Lane invasions**: number of times the ego vehicle crosses a lane marking.
3. **Vehicle position**: gives us the information on the location and rotation of any given actor in the simulation.

Merely having access to this data can be limiting if we want to analyze more deeply the performance of our models. To enhance the analysis capabilities, we use the Behavior Metrics evaluation software tool [45] (see Fig. 1. for architectural details). This tool provides software for testing various approaches for end-to-end imitation learning easily. It also allows us to assess and evaluate complex behaviors for different autonomous robots, using machine learning and deep learning techniques. It generates several evaluation metrics for each possible model. Using the raw information given by the CARLA simulator (e.g., collisions, lane invasions, vehicle position) on the Behavior Metrics tool, we can obtain more interesting metrics that are much more useful to assess the behavior of the ego vehicle.

The CARLA Autonomous Driving Leaderboard[1] is a commonly used evaluation framework for autonomous driving behavior. Since the goal of the present project is not directly covered in this framework, we use Behavior Metrics, including the relevant measurement metrics that give a better understanding of the model performance in the in-traffic follow-lane autonomous driving task. These metrics include the previously presented data extracted from CARLA and some other complementary metrics:

1. **Success rate**: this percentage represents the rate of the vehicle completing one lap. In this case, a lap is considered complete when the ego vehicle successfully drives from the starting point, around a preset route and returns to the starting spot without colliding with other vehicles, objects, buildings, or props along the way. The routes form closed loops.
2. **Mean position deviation (MPD) per km**: the average deviation of the vehicle, in meters, from the center of its designated lane(*centerOfLane*) per kilometer. Measured in meters. It is calculated using the mean of all the points obtained using the minimum Euclidian distance (*MinED*) of each traversed position (*EgoVehiclePosition*) to the center of the lane (*centerOfLane*). This equation can be expressed as follows:

$$\mathrm{MPD} = \frac{1}{N}\sum_{i} \mathrm{MinED}(\mathrm{EgoVehiclePosition}_i, \mathrm{centerOfLane}) \qquad (1)$$

[1] https://leaderboard.carla.org/

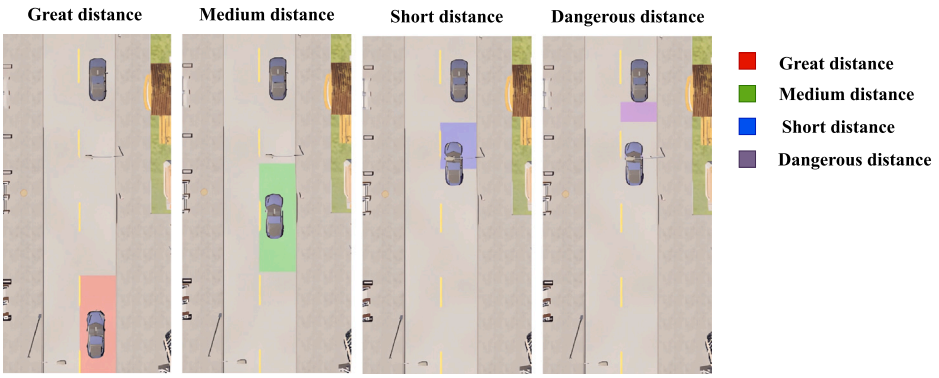

Fig. 5. Great, medium, short, and dangerous distances to the front car.

3. **Lane invasions per km**: number of times the vehicle crosses into the opposite lane per kilometer.
4. **Distance to the front car**: this metric is used for measuring the ego vehicle's handling of obstacles (see Fig. 5 for details). Initially, we obtain the distance to the nearest vehicle and the yaw degree difference between the two cars. This information helps to determine whether the nearest vehicle is truly ahead of the ego vehicle or merely passing it in the opposite lane.
   This distance is discretized into four bins: great distance (20-50 m), medium distance (15-20 m), short distance (6-15 m), and dangerous distance (0-6 m). This metric aims to observe how the ego vehicle maintains a specific distance while approaching the front vehicle. By analyzing whether the ego vehicle enters a dangerous distance and to what extent, we can determine if it progressively slows down when the front car halts or if it fails to make any effort to avoid a collision, leading to an accident. This discretized metric establishes a structured framework for evaluating the ego vehicle's safety performance, gaining a more comprehensive understanding of how the ego vehicle adapts to diverse scenarios.

These metrics are enough for evaluating the in-traffic follow-lane task, as they indicate the car's adherence to its lane and the quality of the other vehicle's handling, including the Distance to the front car.

4. Experiments

In this section, we conduct an experimental validation of the two models: PilotNet* and PilotNet** for the task of follow-lane in traffic situations. These experiments provide relevant results about each model behavior in three different conditions: the first two focused on typical executions without and with traffic, and the third one testing the models' generalization capabilities for obstacle avoidance with a varied set of vehicles.

For these experiments, we have used the CARLA simulator and Behavior Metrics evaluation tool running at 10 Hz. The hardware used for the experiments includes an NVIDIA 3060 GTX GPU. The evaluation takes place in Town02 where the training data was collected; and in Town01, which was never used to train the model. We provide both scenarios to showcase the differences between a circuit already employed in the dataset generation and a never-seen scenario. For each experiment, models are trained with a particular version of the dataset, also showcasing their differences.

The car starts from a fixed position in the urban scenario, driving clockwise and anticlockwise. These starting points form a closed loop route, so the vehicle can complete a lap reaching the starting point again after some time.

We omit a comparison with prior baseline methods as our research question specifically focuses on whether a shallow visual-based end-to-end DL model, utilizing imitation learning, can autonomously navigate without colliding with other vehicles. While existing state-of-the-art models are designed for a broader range of tasks, they often incorporate more sensors, deeper and more complex architectures, and utilize large datasets. Prior studies [39] underscore the significance of small and efficient deep learning networks for autonomous driving, networks suitable for deployment across various devices with differing hardware capabilities and we prioritize the simplicity of our models.

4.1. Typical execution without traffic

In this experiment, the model underwent three clockwise laps and three anticlockwise laps, traversing designated routes within each town.

This experiment tests the follow-lane capacities of the models. Specifically, we consider two models to see if they can still effectively follow the lane in the absence of oncoming cars. The PilotNet* model trained with the Traffic-1 version of the dataset and the PilotNet** model trained with the Traffic-1 and Traffic-6 versions of the dataset. The results for these experiments are provided in Table 2, including the evaluation metrics presented in the previous section. All three models drive successfully keeping the lane without any missed attempts, but we can already see some differences. We can see that PilotNet** trained with Traffic-6 and Traffic-1 is better in terms of mean position deviation and lane invasions number than PilotNet*. This fact proves that adding speed is valuable even in the simplest task of following the lane without traffic. Furthermore, for the PilotNet**, training with a diverse dataset like Traffic-6, compared to Traffic-1, can improve the vehicle's perception of its surroundings in the town and reduce the possibility of confusing certain urban sections with front vehicles, making its driving more confident and smooth.

4.2. Typical execution with traffic

In this experiment, we test the models in a more difficult situation, with one front vehicle to understand the implications of this new scenario. The typical execution with traffic consisted of a total of 12 runs, with the same settings as without traffic. We evaluate the model's performance in a full lap simulation, specifically focusing on both lane-keeping and obstacle avoidance. The additional vehicle moves independently around the scenario, always preceding the ego vehicle, and it will drive slowly to disturb the ego vehicle as much as possible. Although the ego vehicle ignores traffic lights and signals, the additional front vehicle considers traffic lights and follows their indications. As in the previous experiment, the models are trained with the Traffic-1 and Traffic-6 versions of the dataset. The front vehicle used in the Traffic-1 dataset is the same as the one used for this particular experiment. In Table 3, the results are shown. We can see a clear difference now between PilotNet* and PilotNet** where the latter outperforms the former. This difference in performance dealing

Table 2
Metrics for two different towns and models in free-road conditions. Success rate: the higher the better; the rest: the lower the better.

	Town01			Town02		
Model	PilotNet*	PilotNet**	PilotNet**	PilotNet*	PilotNet**	PilotNet**
Training dataset	Traffic-1	Traffic-1	Traffic-6	Traffic-1	Traffic-1	Traffic-6
Success rate (%)	100	100	100	100	100	100
MPD	0.33	0.3	**0.19**	0.84	0.49	**0.32**
Lane invasions	14.884	10.02	**4.75**	26.56	15.4	**3.42**

Table 3
Metrics for two different towns and models in in-traffic conditions.

	Town01			Town02		
Model	PilotNet*	PilotNet**	PilotNet**	PilotNet*	PilotNet**	PilotNet**
Training dataset	Traffic-1	Traffic-1	Traffic-6	Traffic-1	Traffic-1	Traffic-6
Success rate (%)	0	16	**81**	0	83	**100**
MPD	43.12	18.07	**0.26**	50.57	1.97	**0.32**
Lane invasions	28.87	25.84	**6.54**	69.65	21.15	**1.48**

with other vehicles ahead is what makes the PilotNet** a promising model to be trained with Traffic-6 to generalize to different types of front vehicles and is more promising than PilotNet* which fails in many runs.

We observe that the Success rate is higher for PilotNet** trained with Traffic-6, completing each experiment in Town02 and the majority of the experiments in Town01, which is the actual test scenario. On the contrary, PilotNet* results are abruptly worse, without any successful experiment. These results show that PilotNet* always collides with the front vehicle at some point in the experiment. The results for the rest of the evaluated metrics follow a similar pattern. Looking at MPD, we can see that PilotNet** generates great results, so we can consider that it can follow the lane correctly while keeping a low number of lane invasions.

Comparing both PilotNet** models trained with different datasets, we observe that the version trained with Traffic-6 outperforms the one trained with Traffic-1. This is evidenced by its more robust understanding of the environment, effectively distinguishing between front vehicles and structures such as buildings. The discrepancy in performance between Town02 and Town01 may stem from its inability to isolate the front vehicle from the urban background

This experiment, not only highlights the importance of having a much more varied dataset so that the model can drive with other vehicles in never-before-seen towns but also proves the importance of adding the speed to the model architecture for optimal performance in traffic situations. The rationale behind this approach is that by incorporating the speed into the model, the ego vehicle can enhance its control over its speed with greater precision compared to scenarios where this information is not considered. This level of control allows the ego vehicle to swiftly respond to nearby vehicles by reducing its speed when necessary, contributing to overall safety and smoother driving behavior.

In contrast, PilotNet* does not consider the speed of the ego vehicle during the experiment. It generates control decisions solely based on instantaneous visual data, which poses challenges in adjusting its velocity. The visual input appears to be deficient in critical information when encountering another vehicle, as the ego vehicle lacks crucial data regarding its state and current speed.

4.3. Generalization for different front vehicles

In the final experiment, we test the generalization capabilities of the models to never-seen front vehicles. For this experiment, we discarded the PilotNet* model because of its poor performance in the previous experiment. Instead, we use the PilotNet** trained for the second experiment with the Traffic-1 dataset, and we compare it to a fine-tuned version of PilotNet** trained with the Traffic-6 dataset.

To assess the generalization capabilities, it was unnecessary to conduct full-lap trials. Instead, we tested the ego vehicle with a series of vehicles in front, which moved at a significantly slow pace, stopping at red traffic lights and resuming driving when the lights turned green.

In this experiment, a total of 12 distinct vehicles were employed, comprising 8 novel vehicles unseen during the training phase, and 4 vehicles previously encountered in the training dataset (see Fig. 6 for details of used vehicles). We conducted tests with each leading vehicle traversing the same route in both clockwise and anticlockwise directions. Finally, we carried out this experiment three times to ensure the results were reliable. This gave us a total of 72 runs for each model concluding with a total of 144 runs. Each run gave us two types of metrics: Success rate and Distance to the front car. The dangerous distance should be avoided as it is deemed unacceptable in real-world scenarios which we consider unsafe. Complying with these distances is crucial for safe driving in real traffic. The Success rate metric measures cases where the ego vehicle encounters the front vehicle without any collisions, indicating the ego vehicle's ability to detect and respond to obstacles in its path.

Table 4 presents the ratio of the ego vehicle's distance behind the front car for a one-kilometer distance, allowing us to assess the behavior of the ego vehicle. It may be observed that the PilotNet** trained with Traffic-6 does a better job than PilotNet** trained with Traffic-1 by spending less time at dangerous distances from the front car.

Additionally, we can analyze how the vehicle transitions from a far distance to a close distance by examining the values in Table 4. The PilotNet** [48] trained with Traffic-6 exhibits a progressive descent of the speed from greater distances to closer ones, spending more time the farther it is from the front car and reducing its distance as it approaches, which leads to the expected stopping behavior. On the other hand, the PilotNet** trained with Traffic-1 shows almost the same amount of time spent on medium and short distances. It lacks progressive advancement and struggles when confronted with an obstacle ahead.

Table 4 also shows the Success rate of each PilotNet** and the better generalization capacities from the PilotNet** trained with Traffic-6 compared to the PilotNet** trained with Traffic-1.

The fact that this performance is retrieved from Town01 proves that the models can generalize to other never-seen scenarios. Again, we have also proved the importance of adding speed to the architecture for a better understanding of the world that the vehicles use for making their control decisions.

PilotNet** trained with Traffic-6 dataset demonstrates its capabilities in lane-keeping and adaptive behavior when encountering vehicles of different shapes and colors. The experimental evaluation shows that it demonstrates an ability to understand its surroundings and has proven its effectiveness in maintaining appropriate distances and

Fig. 6. Detail of vehicles used for experimental validation.

Table 4
Metrics for the distance to the front vehicle.

	Town01	
Model	PilotNet**	PilotNet**
Training dataset	Traffic-1	Traffic-6
Dangerous distance	6%	**2%**
Short distance	25%	16%
Medium distance	27%	30%
Great distance	42%	52%
Success rate	16%	**86%**

Table 5
Success rate metric for each of the 12 vehicles.

	Town01	
Model	PilotNet**	PilotNet**
Training dataset	Traffic-1	Traffic-6
vehicle.mini.cooper_s	50%	83%
vehicle.volkswagen.t2	0%	100%
vehicle.micro.microlino	50%	100%
vehicle.carlamotors.carlacola	0%	100%
vehicle.jeep.wrangler_rubicon	0%	100%
vehicle.citroen.c3	17%	100%
vehicle.toyota.prius	0%	83%
vehicle.dodge.charger_police	33%	83%
vehicle.kawasaki.ninja	0%	100%
vehicle.diamondback.century	0%	50%
vehicle.ford.ambulance	50%	66%
vehicle.carlamotors.firetruck	0%	66%

avoiding collisions with various vehicles, such as other cars and motorcycles, including vehicles that the model has never seen before in training (displayed on the last 8 rows of Table 5). The reasoning behind this result is that, due to exposure to a wider range of vehicles, the model trained with Traffic-6 demonstrates improved generalization capabilities, as it gains a deeper understanding of the concept of a vehicle and develops more optimal strategies when encountering them.

However, there are also some limitations when faced with previously unseen road users, such as cyclists, ambulances, and firetrucks (see Fig. 6.). The Success rate for each front vehicle, as displayed in Table 5, not only highlights the overall superiority of the PilotNet** trained with Traffic-6 compared to Traffic-1, as indicated in Table 4 but also underscores its mentioned limitations. Particularly noticeable are the lower Success rates associated with the Diamondback Century (bicycle), ambulance, and firetruck.

Due to its reliance on the visual cues provided by the wheels of the leading vehicle, as outlined in Fig. 4, the ego vehicle encounters difficulty in accurately determining whether to stop or proceed when confronted with a cyclist ahead. This challenge arises from the narrower tires typically found on bicycles compared to those of conventional road vehicles.

The challenge posed by ambulance and firetruck detection, as depicted in the last two images of Fig. 6, arises from the partial concealment of their wheels from the ego vehicle's perspective. This ambiguity complicates the model's capacity to consistently distinguish between car tires and other objects, although it occasionally manages to stop for such vehicles.

5. Conclusions

This paper presents a proposal for safe autonomous driving in traffic scenarios following an end-to-end vision-based approach with imitation learning and deep learning. First, a large supervised dataset has been generated with many examples of the onboard camera images and the corresponding control commands recorded from the expert agent while the car was driving in free-road or in-traffic conditions within the Carla simulator. This dataset is publicly available [40], along with models, architectures, and software tool, for replicating results and further research from the community.

Second, we have developed two deep learning models based on PilotNet, adding in PilotNet* new dropout layers and outputs for controlling throttle and brake, not only steering commands, and also including in PilotNet** the speed as input.

Third, those shallow networks slightly modified from the baseline were trained with this supervised dataset using data augmentation and balancing the raw dataset. They were experimentally evaluated and validated. Beyond low values of the loss function in the test dataset, the system has been validated online with the state-of-the-art Carla simulator, in several Towns, using objective, holistic, and quantitative metrics

from Behavior Metrics tool. For instance: mean position deviation from lane center, lane invasions, and distance to the front vehicle.

The experimental results show that the Pilotnet** model, when trained with the Traffic-6 dataset, successfully drives the car in traffic conditions without sacrificing performance in free-road conditions. It also keeps safety distance from the oncoming cars and even properly generalizes to several types of front vehicles, including vehicles never seen before in the training stage. These results are observed with PilotNet** model proving that those model modifications, despite being slight and applied to an apparent simple model, are good contributions and enough to achieve the new desired 'drive in traffic' capability beyond the basic lane-following behavior.

As future lines, we are working to extend the end-to-end approach to deal successfully with traffic lights and crossroads and to transfer this learning from the simulator to a real car. For this purpose, a new dataset of real-world scenarios and further training for this new approach would be needed. In addition, more recent state-of-the-art baseline models will be explored and assessed for the same 'drive in traffic' functionality, using both the same training dataset and others from different expert agents.

CRediT authorship contribution statement

Sergio Paniego: Conceptualization, Formal analysis, Investigation, Methodology, Software, Writing – original draft, Writing – review & editing. **Enrique Shinohara:** Conceptualization, Formal analysis, Methodology, Software, Writing – original draft, Writing – review & editing. **JoséMaría Cañas:** Conceptualization, Funding acquisition, Methodology, Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We have shared the link to our data/code as a citation.

Acknowledgments

This work is supported by (GAIA) Gestión integral para la prevención, extinción *y* reforestación debido a incendios forestales, Proyectos de I+D en líneas estratégicas en colaboración entre organismos de investigación *y* difusión de conocimientos TRANSMISIONES 2023. Ref PLEC2023-010303 (2024–2026) by Agencia Estatal de Investigación de España.

References

[1] T. Litman, Autonomous vehicle implementation predictions implications for transport planning, Tech. rep., Victoria Transport Policy Institute, 2022.

[2] J. Fingas, Waymo trials fully driverless rides in San Francisco, Engadget (2022) https://www.engadget.com/waymo-fully-driverless-rides-san-francisco-183703989.html, [Online; accessed 16-Feb-2024].

[3] R. Akhtar, Tesla FSD on its way to Europe as Test Operators hiring spree begins, The Driven (2023) https://thedriven.io/2023/06/07/tesla-fsd-on-its-way-to-europe-as-test-operators-hiring-spree-begins/, [Online; accessed 16-Feb-2024].

[4] SAE International, Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles, 2021, URL https://www.sae.org/standards/content/j3016_202104/.

[5] C. Gómez-Huélamo, A. Diaz-Diaz, J. Araluce, M.E. Ortiz, R. Gutiérrez, F. Arango, Á. Llamazares, L.M. Bergasa, How to build and validate a safe and reliable Autonomous Driving stack? A ROS based software modular architecture baseline, in: 2022 IEEE Intelligent Vehicles Symposium, IV, 2022, pp. 1282–1289, http://dx.doi.org/10.1109/IV51971.2022.9827271.

[6] E. Yurtsever, J. Lambert, A. Carballo, K. Takeda, A survey of autonomous driving: Common practices and emerging technologies, IEEE Access 8 (2020) 58443–58469, http://dx.doi.org/10.1109/ACCESS.2020.2983149.

[7] P. Wu, Y. Cao, Y. He, D. Li, Vision-based robot path planning with deep learning, in: M. Liu, H. Chen, M. Vincze (Eds.), Computer Vision Systems, Springer International Publishing, Cham, 2017, pp. 101–111.

[8] S. Casas, A. Sadat, R. Urtasun, MP3: A unified model to map, perceive, predict and plan, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2021, pp. 14398–14407, http://dx.doi.org/10.1109/CVPR46437.2021.01417.

[9] W. Zeng, S. Wang, R. Liao, Y. Chen, B. Yang, R. Urtasun, DSDNet: Deep structured self-driving network, in: A. Vedaldi, H. Bischof, T. Brox, J.-M. Frahm (Eds.), Computer Vision – ECCV 2020, Springer International Publishing, Cham, 2020, pp. 156–172, http://dx.doi.org/10.1007/978-3-030-58589-1_10.

[10] P. Karkus, B. Ivanovic, S. Mannor, M. Pavone, DiffStack: A differentiable and modular control stack for autonomous vehicles, in: 6th Annual Conference on Robot Learning, 2022, URL https://openreview.net/forum?id=teEnA3L4aRe.

[11] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, A. Geiger, TransFuser: Imitation with transformer-based sensor fusion for autonomous driving, IEEE Trans. Pattern Anal. Mach. Intell. 45 (11) (2023) 12878–12895, http://dx.doi.org/10.1109/TPAMI.2022.3200245.

[12] P. Wu, X. Jia, L. Chen, J. Yan, H. Li, Y. Qiao, Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline, in: A.H. Oh, A. Agarwal, D. Belgrave, K. Cho (Eds.), Advances in Neural Information Processing Systems, 2022, URL https://openreview.net/forum?id=DhmYYrH_M3m.

[13] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang, L. Lu, X. Jia, Q. Liu, J. Dai, Y. Qiao, H. Li, Planning-oriented autonomous driving, in: 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2023, pp. 17853–17862, http://dx.doi.org/10.1109/CVPR52729.2023.01712.

[14] L. Tai, S. Li, M. Liu, A deep-network solution towards model-less obstacle avoidance, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2016, pp. 2759–2764, http://dx.doi.org/10.1109/IROS.2016.7759428.

[15] F. Torabi, G. Warnell, P. Stone, Behavioral cloning from observation, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18, International Joint Conferences on Artificial Intelligence Organization, 2018, pp. 4950–4957, http://dx.doi.org/10.24963/ijcai.2018/687.

[16] T. Pearce, J. Zhu, Counter-strike deathmatch with large-scale behavioural cloning, in: 2022 IEEE Conference on Games (CoG), IEEE, 2022, pp. 104–111.

[17] F. Codevilla, M. Müller, A. López, V. Koltun, A. Dosovitskiy, End-to-end driving via conditional imitation learning, in: 2018 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2018, pp. 4693–4700.

[18] M. Bansal, A. Krizhevsky, A. Ogale, ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst, 2018, http://dx.doi.org/10.15607/RSS.2019.XV.031.

[19] C. Diehl, J. Adamek, M. Krüger, F. Hoffmann, T. Bertram, Differentiable constrained imitation learning for robot motion planning and control, 2023, arXiv preprint arXiv:2210.11796.

[20] F. Codevilla, E. Santana, A.M. López, A. Gaidon, Exploring the limitations of behavior cloning for autonomous driving, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 9329–9338.

[21] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, H. Li, End-to-end autonomous driving: Challenges and frontiers, 2023, arXiv preprint arXiv:2306.16927.

[22] D. Garg, S. Chakraborty, C. Cundy, J. Song, S. Ermon, IQ-learn: Inverse soft-q learning for imitation, Adv. Neural Inf. Process. Syst. 34 (2021) 4028–4039.

[23] J. Ho, S. Ermon, Generative adversarial imitation learning, Adv. Neural Inf. Process. Syst. 29 (2016).

[24] M. Toromanoff, E. Wirbel, F. Moutarde, End-to-end model-free reinforcement learning for urban driving using implicit affordances, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2020.

[25] M. Jaritz, R. de Charette, M. Toromanoff, E. Perot, F. Nashashibi, End-to-end race driving with deep reinforcement learning, in: 2018 IEEE International Conference on Robotics and Automation, ICRA, IEEE Press, 2018, pp. 2070–2075, http://dx.doi.org/10.1109/ICRA.2018.8460934.

[26] A. Amini, I. Gilitschenski, J. Phillips, J. Moseyko, R. Banerjee, S. Karaman, D. Rus, Learning robust control policies for end-to-end autonomous driving from data-driven simulation, IEEE Robot. Autom. Lett. 5 (2) (2020) 1143–1150, http://dx.doi.org/10.1109/LRA.2020.2966414.

[27] R. Gutiérrez-Moreno, R. Barea, E. López-Guillén, J. Araluce, L.M. Bergasa, Reinforcement learning-based autonomous driving at intersections in CARLA simulator, Sensors 22 (21) (2022) 8373.

[28] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L.D. Jackel, M. Monfort, U. Muller, J. Zhang, et al., End to end learning for self-driving cars, 2016, arXiv preprint arXiv:1604.07316.

[29] P.A. Lopez, et al., Microscopic Traffic Simulation using SUMO, in: 2018 21st International Conference on Intelligent Transportation Systems, ITSC, 2018, pp. 2575–2582, http://dx.doi.org/10.1109/ITSC.2018.8569938.

[30] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, V. Koltun, CARLA: An open urban driving simulator, in: Conference on Robot Learning, PMLR, 2017, pp. 1–16.

[31] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Ng, ROS: an open-source Robot Operating Storcsystem, in: ICRA Workshop on Open Source Software, Vol. 3, 2009.
[32] E. Espié, C. Guionneau, B. Wymann, C. Dimitrakakis, R. Coulom, A. Sumner, TORCS, the open racing car simulator, 2005.
[33] A. Geiger, P. Lenz, C. Stiller, R. Urtasun, Vision meets robotics: The KITTI dataset, Int. J. Robot. Res. (IJRR) (2013).
[34] H. Caesar, V. Bankiti, A.H. Lang, S. Vora, V.E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, O. Beijbom, nuScenes: A multimodal dataset for autonomous driving, in: CVPR, 2020.
[35] E. Santana, G. Hotz, Learning a driving simulator, 2016, http://dx.doi.org/10.48550/ARXIV.1608.01230, arXiv preprint arXiv:1608.01230, URL https://arxiv.org/abs/1608.01230.
[36] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. Qi, Y. Zhou, Z. Yang, A. Chouard, P. Sun, J. Ngiam, V. Vasudevan, A. McCauley, J. Shlens, D. Anguelov, Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset, in: 2021 IEEE/CVF International Conference on Computer Vision, ICCV, 2021, pp. 9690–9699, http://dx.doi.org/10.1109/ICCV48922.2021.00957.
[37] H. Caesar, J. Kabzan, K.S. Tan, W.K. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, S. Omari, NuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles, 2022, arXiv preprint arXiv:2106.11810.
[38] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, J. Hays, Argoverse: 3D tracking and forecasting with rich maps, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR, 2019, pp. 8740–8749, http://dx.doi.org/10.1109/CVPR.2019.00895.
[39] S. Paniego, N. Paliwal, J. Cañas, Model optimization in deep learning based robot control for autonomous driving, IEEE Robot. Autom. Lett. 9 (1) (2024) 715–722, http://dx.doi.org/10.1109/LRA.2023.3336244.
[40] Open-source paper resources contributors, Open-source paper resources, 2023, https://roboticslaburjc.github.io/publications/2023/autonomous_driving_in_traffic_with_end_to_end_vision_based_deep_learning, [Online; accessed 16-Feb-2024].
[41] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, TensorFlow: A system for large-scale machine learning, in: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation, OSDI '16, USENIX Association, 2016, pp. 265–283.
[42] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[43] A. Buslaev, V.I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, A.A. Kalinin, Albumentations: fast and flexible image augmentations, Information 11 (2) (2020) 125.
[44] S. Ross, G. Gordon, D. Bagnell, A reduction of imitation learning and structured prediction to no-regret online learning, in: G. Gordon, D. Dunson, M. Dudík (Eds.), Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, in: Proceedings of Machine Learning Research, Vol. 15, PMLR, Fort Lauderdale, FL, USA, 2011, pp. 627–635, URL https://proceedings.mlr.press/v15/ross11a.html.
[45] Behavior Metrics contributors, Behavior metrics, 2022, https://github.com/JdeRobot/BehaviorMetrics, [Online; accessed 16-Feb-2024].
[46] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. Jackel, U. Muller, Explaining how a deep neural network trained with end-to-end learning steers a car, 2017, arXiv preprint arXiv:1704.07911.
[47] R.R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-CAM: Visual explanations from deep networks via gradient-based localization, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 618–626.
[48] JdeRobot, Showcasing the social driving capacity with different front vehicles, 2023, URL https://www.youtube.com/watch?v=mVSfxQeWwrQ, [Online; accessed 16-Feb-2024].

Sergio Paniego Blanco is a Ph.D. student at the Department of Teoria de la Señal y de las Comunicaciones y Sistemas Telemáticos y Computación at Universidad Rey Juan Carlos. He completed his Masters in Artificial Intelligence at Universidad Politécnica de Madrid, Spain in 2019 and his B. in Computer Engineering and Software Engineering at Universidad Rey Juan Carlos in 2018. His main research areas are Autonomous Driving, Deep Learning, Model Optimization and Robotics.

Enrique Y. Shinohara Soto graduated recently this year completing his master's degree at the university Rey Juan Carlos. Before that, he obtained his bachelor's degree in Computer Science and Engineering at the university Carlos III.

JoséMaría Cañas-Plaza is tenured associate professor at Universidad Rey Juan Carlos, where he leads the RoboticsLabURJC. He received the M.S. and Ph.D. degrees in telecommunications engineering from the Universidad Politécnica de Madrid. He has done research in robotics at Carnegie Mellon University and the Georgia Institute of Technology. His research interests include AI/ML driven Robotics, Computer Vision, and the engineering education of those disciplines.