Received 11 June 2022, accepted 12 July 2022, date of publication 18 July 2022, date of current version 21 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3192019

TOPICAL REVIEW

A Review of End-to-End Autonomous Driving in Urban Environments

DANIEL COELHO AND MIGUEL OLIVEIRA
Department of Mechanical Engineering, University of Aveiro, 2810-193 Aveiro, Portugal
Institute of Electronics and Informatics Engineering of Aveiro, University of Aveiro, 3810-193 Aveiro, Portugal

Corresponding author: Daniel Coelho (danielsilveiracoelho@ua.pt)

This work was supported by the National Funds through the FCT—Foundation for Science and Technology under Project UIDB/00127/2020.

ABSTRACT Autonomous driving in urban environments requires intelligent systems that are able to deal with complex and unpredictable scenarios. Traditional modular approaches focus on dividing the driving task into standard modules, and then use rule-based methods to connect those different modules. As such, these approaches require a significant effort to design architectures that combine all system components, and are often prone to error propagation throughout the pipeline. Recently, end-to-end autonomous driving systems have formulated the autonomous driving problem as an end-to-end learning process, with the goal of developing a policy that transforms sensory data into vehicle control commands. Despite promising results, the majority of end-to-end works in autonomous driving focus on simple driving tasks, such as lane-following, which do not fully capture the intricacies of driving in urban environments. The main contribution of this paper is to provide a detailed comparison between end-to-end autonomous driving systems that tackle urban environments. This analysis comprises two stages: a) a description of the main characteristics of the successful end-to-end approaches in urban environments; b) a quantitative comparison based on two CARLA simulator benchmarks (*CoRL2017* and *NoCrash*). Beyond providing a detailed overview of the existent approaches, we conclude this work with the most promising aspects of end-to-end autonomous driving approaches suitable for urban environments.

INDEX TERMS Autonomous driving, end-to-end, imitation learning, reinforcement learning, urban environments.

I. INTRODUCTION

In the last decades, the field of autonomous driving (AD) has received a massive amount of interest, both in academia [1]–[6] and in industry [7]–[10]. The principal factor that triggered this interest concerns safety issues [11]. National Highway Traffic Safety Administration (NHTSA) reported that 94% of accidents are caused by drivers [12]. Another key factor is related to the traffic flow. Replacing humans by AD systems result in an optimized traffic flow, offering both financial and environmental benefits [13]. The benefits of fully AD appear to be considerable, which is why the research on autonomous driving remains an active area [14]. One of the most difficult challenges in this field concerns AD in urban environments [15]. Compared with highway driving or lane following, urban environments pose additional obstacles due to the unpredictability and variety of agents present in the scene, as well as complex and uncertain situations, such as pedestrians crossing lanes, traffic-lights, intersections, among others. In 2007, during the DARPA Urban Challenge [16], several researchers around the world tested their AD systems in a controllable urban environment and only six teams were able to complete the event [17]. The environment used in DARPA still lacked certain aspects of the real word, such as pedestrians and cyclists. Nevertheless, the fact that six teams were able to complete the event was extraordinary, especially at that time. Despite all the impressive research in this area, fully AD systems capable of driving in complex and unknown urban environments are still years away and the main reason for this is the arduous task of

The associate editor coordinating the review of this manuscript and approving it for publication was Binit Lukose.

generalization to unpredictable situations in a short period of time [18].

AD systems are complex systems that integrate many technologies, from sensors, processing units, software, among others. Therefore, AD systems need to deal with a wide range of problems: sensor inaccuracies, hardware reliability, object detection, localization, etc. The conventional approach to tackle this diversity of problems consists of dividing the driving task into standard modules such as object detection, localization, path planning, etc. and then build rule-based methods to connect the different modules [19]–[21] (see Figure 1). This approach is commonly called modular, and is widely used in the industry [7]. The interconnectivity between different modules in a system is a problem extensively investigated in the robotics field [22]. For example, this interconnectivity between modules led to the creation of frameworks, such as Robot Operating System (ROS) [23], [24]. The modular architecture enables the development of each module in an independent fashion facilitating the collaboration between all elements of the engineering team. In addition, the development of individual and specific modules divides the autonomous driving task into a set of narrow problems widely investigated in the literature, as is the case of localization, computer vision, motion planning, among others. Finally, interpretability constitutes one of the great advantages of these modular approaches: as the entire system is divided into modules, the source of a malfunction can more easily be tracked to the responsible module.

The major disadvantage of modular systems is the arduous task of developing and maintaining the interconnection between all modules in the system. For example, different scenarios may require different connections between modules [25], which compromises the modularity paradigm. The modular architecture is also prone to error propagation [26], in which a minor error in one module can produce catastrophic results in another, for example, a misclassification of a traffic-light can influence the decision-making process to generate a path planning that leads to a collision. Additionally, as the modules are task-specialized, they may fail to generalize to unusual conditions and unexpected situations.

More recently, end-to-end approaches emerged as an alternative technique to tackle the AD problem. The end-to-end approach formulates the AD task as an end-to-end learning process, in which the objective is to learn a policy capable of transforming sensor data into control commands [14]. In general, end-to-end architectures are simpler and have fewer components than modular architectures (see Figure 1). Unlike the modular paradigm, the end-to-end paradigm also captures the human driving essence: a simultaneous perception and action [27]. The downside of end-to-end systems is the lack of interpretability [24]. It is difficult to track down the source errors or to explain certain decisions taken by the model [25]. Over the years, the interest in the end-to-end approach for AD was scarce, especially compared with the amount of research done in modular approaches. However, due to the rise of Artificial Intelligence [28]–[30] in the past years, and due to the recent developments of Deep Reinforcement Learning (Deep RL) [31]–[33] by Deep Mind [34], [35], end-to-end approaches have begun to show promising results [36], [37].

In end-to-end approaches, there are two different learning methodologies: Imitation Learning (IL) and Reinforcement Learning (RL). IL aims to learn a policy by observing the actions performed by humans. It is a supervised learning approach, in which the model tries to mimic human behavior [39]. NVIDIA achieved excellent results using this methodology by training a convolution neural network (CNN) to predict the steering angle of a vehicle [40]. One advantage of the IL methodology is that it can use solely Deep Learning (DL) and merely optimize the parameters of the model to reduce the difference between the model behavior and human behavior. However, the process of scaling an AD system based on IL is a challenging task due to the impossibility of covering all possible scenarios in the training phase [7].

RL methodologies aim to learn a policy that maximizes the cumulative rewards received by an automatic system, as it interacts with the environment [41], [42]. One variant of RL is Deep RL, which combines DL with RL [43]. Although there are some technical differences between RL and Deep RL, for the purposes of this review, which is to differentiate IL and RL, we will use the terms RL and Deep RL interchangeably. In the case of RL, there is no need to collect data from human driving, because as the agent interacts with the environment, it learns how to behave in order to maximize the reward. As the training of RL models occurs online, it is possible to explore the environment and train simultaneously, which is a great advantage compared with the IL models. The downside is that RL is less data-efficient in the training stage [24]. Liang *et al.* combined the advantages of IL and RL by creating an IL model based on labeled data, and then optimizing the policy using an online RL-based policy tuning [44]. One crucial element of RL algorithms is the definition of rewards. As the agent tries to maximize the reward, the definition of the reward function directly influences the behavior learned by the agent. One common example is to reward the movement towards the goal [7], [44], or to punish whenever a collision occurs [45]. In 2018, Kendall *et al.* demonstrated the first real-life application of RL in AD, in which they were able to train a driving policy capable of learning how to follow a lane in less than 30 minutes [7].

The application of end-to-end methodologies in AD is relatively recent: the first use case was in 2016 [40]. In the three subsequent years, several approaches were proposed focusing on simplified versions of AD, such as lane following [46]–[51]. As expected, the application of end-to-end AD in urban environments is even more recent: 65% of the works found on this topic are from 2020 or 2021. As these works are recent, the majority of reviews of AD do not include their findings [20], [52], [53]. The only review solely focused on end-to-end was proposed by Tampuu *et al.*, where the authors performed a thorough analysis of the different architectures and training methods of end-to-end approaches

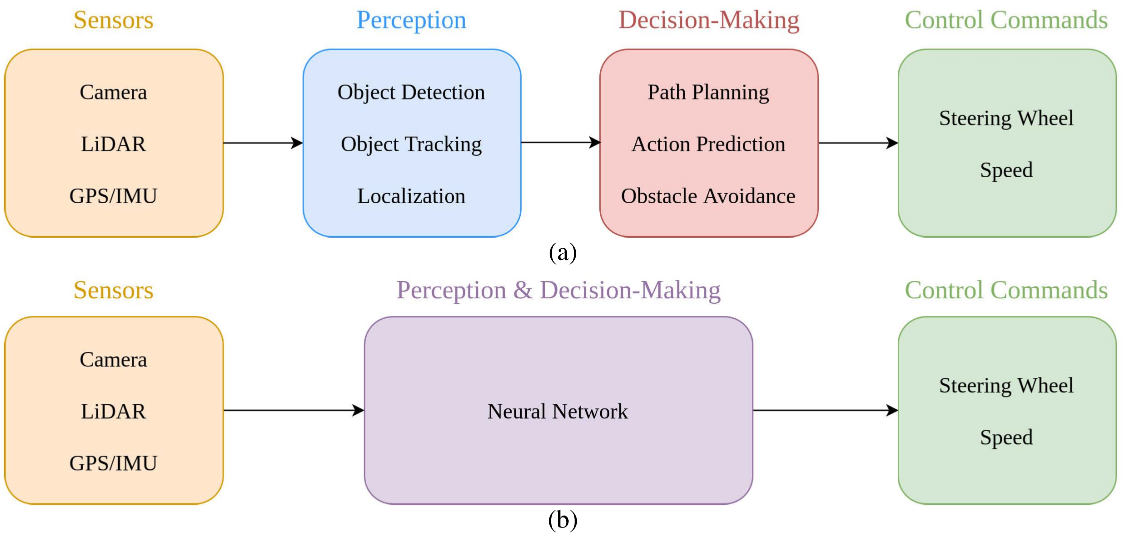


FIGURE 1. Architecture of: (a) a modular approach [38], and (b) an end-to-end approach. The modular architecture consists of several interconnected modules, whereas in the end-to-end architecture those different modules are replaced by a single, learning-based module.

applied in AD [24]. However, their work was not focused especially in urban environments, and therefore, some of the findings should not be generalized to more complex environments. This paper presents the first review targeting **end-to-end autonomous driving in urban environments**, which is an emerging topic in the literature. The key contributions of this paper are to provide:

- a description of the main differences between the successful end-to-end approaches in urban environments;
- a quantitative analysis based on two CARLA simulator benchmarks (*CoRL2017* [54] and *NoCrash* [55]).

This paper considers only systems trained and tested in the CARLA simulator for two reasons: a) CARLA is considered the state-of-the-art open-source simulator for self-driving cars [56]; b) to ensure a fair comparison between all approaches. In [56], the authors performed an extensive comparative analysis of six simulators based on features like perception, path planning, 3D virtual environments, traffic-scenario, scalability, etc. and concluded that CARLA outperforms all other simulators.

The remainder of this document is organized as follows: in Section II, we present a thorough analysis of the most successful end-end AD systems in urban environments; Section III evaluates the detailed approaches based on quantitative metrics; Section IV presents the conclusions.

II. DISCUSSION

In this section, we perform a thorough comparison of several end-to-end AD approaches in urban environments. This comparison is based on three main points: architectures, input sensor modalities, and output modalities. As discussed before, the targeted end-to-end AD approaches discussed in this work are the ones that used CARLA as the environment to train and test the models.

A. ARCHITECTURES

Due to the complexity of urban environments, it is common the usage of low dimensional intermediate representation of the environment instead of parsing the raw data from the scene. One of the options for this low dimensional intermediate representation is called affordances [57]. Sauer *et al.* proposed an AD system based on affordances, especially designed for urban environments [58]. Examples of these affordances include: presence of hazard stop, red traffic-light, speed sign, distance to vehicle, relative angle, and distance to center line. These affordances are predicted by neural networks that receive both RGB images and a navigation command, e.g., “go straight”, “turn left”, or “turn right”. The affordances are then processed by controllers in order to produce the control commands. Figure 2 depicts a simplified version of the system proposed by Sauer *et al.*

Mehta *et al.* also used affordances to aid the AD task [59]. The affordances allow to infuse human knowledge into the system instead of expecting the network to learn all relevant features for driving from scratch. Unlike Sauer *et al.*, in this case the affordances are learned by the network simultaneously with the main task of driving. The authors demonstrated that the joint learning of the auxiliary tasks and the usage of the predicted affordances in the final control commands prediction increases the performance of learning. Furthermore, the authors also claim that the usage of affordances significantly increases the level of interpretability of the system, which is, as explained in Section I, one of the shortcomings of end-to-end systems.

Chen *et al.*, in [36], instead of using human defined labels, used an algorithm to provide the true labels. First, a privileged agent is trained with access to ground-truth data to imitate an expert autopilot. Then, the authors used the trained privileged agent to train another agent with only visual input. The results

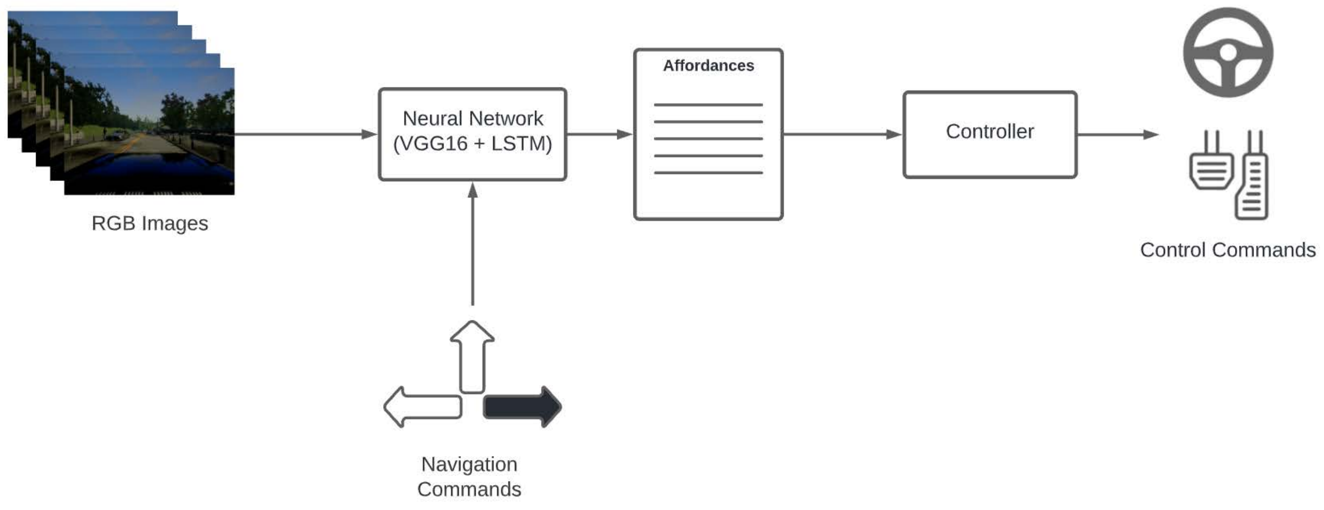


FIGURE 2. Simplified version of the system proposed by Sauer *et al.* [58]. The system receives as input RGB images and a navigation command provided by CARLA. The features are then extracted from the images using convolutional layers of a VGG16 neural network. Based on the navigational command received, the features are processed by a different block to produce the affordances. Finally, the predicted affordances are used by a controller to produce the control commands.

showed that the usage of an agent with privileged information significantly improves the vision-only driving agent.

In [60], Prakash *et al.* proposed an architecture that comprises two main blocks: a *Multi-Modal Fusion Transformer* (TransFuser) and a waypoint prediction network. The TransFuser receives data from different sensor modalities as input, and it produces a compact representation of the environment as output. This process is carried out by using the self-attention mechanisms of transformers [61] to incorporate the information between the different modalities. The compact representation of the environment is then passed into an encoder neural network to reduce its dimensionality, and therefore increase computational efficiency. The waypoint prediction network receives both the encoded version of the environment and the desired trajectory provided by a global planner. The network consists of several Gated Recurrent Units (GRUs) [62], [63], that outputs the predicted trajectory, under the form of waypoints (more details in Section II-C). A simplified version of the system proposed by Prakash *et al.* is depicted in Figure 3.

One of the major problems in applying RL in the field of AD consists of the high-dimensional sensor inputs, as is the case of RGB images. For this reason, most of the applications of RL in AD have focused on simple driving tasks, such as lane following [7], [26]. However, in the last two years, some methods have been proposed that address this problem [14], [64]–[66]. For example, in [14], Agarwal *et al.* presented a framework that creates a low-dimensional state representation that comprises a stack of bird's eye-view semantic segmented images, desired trajectory, kinematics features, and traffic-light states. Then, the low-dimensional state representation is conveyed to the RL algorithm (Proximal Policy Optimization (PPO) [67], [68]). A simplified version of the architecture of the solution is illustrated in Figure 4.

Chen *et al.*, in [64], proposed a system that encodes RGB images and global path trajectories using a CNN [69] and a LSTM (Long Short-Term Memory) [70] to extract both spatial and temporal features. Then, the policy network receives the encoded data and outputs control commands after the defuzzification procedure. The defuzzification procedure is responsible for transforming the output of the policy network into control commands.

Chen *et al.*, in [65], proposed the combination of the modular framework and the RL framework. As input, the system receives data from two sensors: a RGB camera and a LiDAR. During training, a semantic mask is obtained using some components in the modularized framework, such as object detection, mapping, and localization, and then the mask enters the system as labeled data. The policy network receives RGB images and LiDAR data and produces the control commands together with the semantic mask. The semantic mask produced by the policy network provides an interpretable explanation of how the agent understands the world that surrounds him.

The previous approaches based on RL did not have the ability to foresee the future, which is a feature that we, humans, have inherited from years of experience [71]. C. Huang *et al.*, in [72], focuses on building a RL agent capable of predicting new observations. The first layer of the system consists of a Semantic Encoding Mapping (SEM) [73] that learns a semantic representation from raw images. This representation is then sent to a Deep RL algorithm (Deep Deterministic Policy Gradient (DDPG) [74], [75]). The core element of this Deep RL is a deductive reasoner that enables policy to be learned in a model-based manner. This way, it can predict the next state and reward based on the current state and reward, which produces a more reliable driving policy.

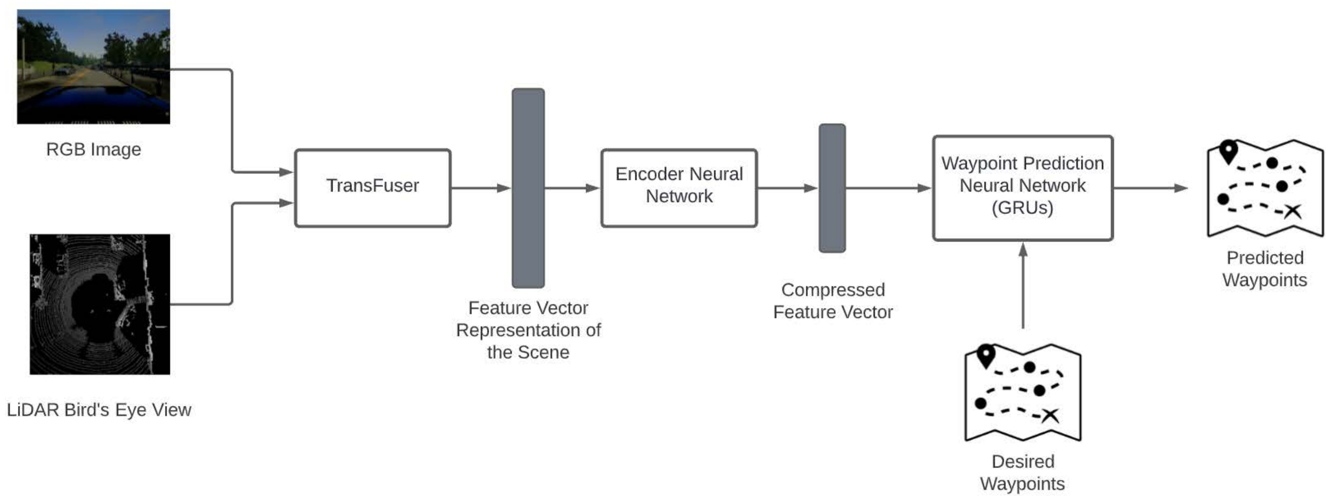

RGB Image
LiDAR Bird's Eye View
TransFuser
Feature Vector Representation of the Scene
Encoder Neural Network
Compressed Feature Vector
Waypoint Prediction Neural Network (GRUs)
Desired Waypoints
Predicted Waypoints

FIGURE 3. Simplified version of the system proposed by Prakash *et al.* [60]. The system receives a RGB image and a LiDAR bird's eye view image as input of the *Multi-Modal Fusion Transformer* (TransFuser). The fusion process is attained by using several transformer modules to combine the intermediate feature maps between both modalities. The output of the TransFuser constitutes a compact representation of the environment that comprises the global context of the scene. This compact representation it then conveyed into a encoder neural network to reduce computational efforts. The waypoint prediction neural network consists of several GRUs that receive both the output of the encoder neural network (compressed feature vector) and the desired trajectory, provided by a global planner, and outputs the future waypoints. The authors reported that vehicle measurements are also used in the TransFuser, but for clarity reasons, that information is omitted.

In [44], Liang *et al.* proposed a system called CIRL (*Controllable Imitative Reinforcement Learning*) that aims to combine the advantages of IL and RL. First a supervised network is pretrained based on human labeled data. Then a Deep RL model (DDPG) is initialized with the pretrained weights of the supervised network. The authors claim that the usage of human driving demonstrations for the initialization of the RL model can significantly reduce the sample complexity, and therefore, saving innumerous hours of exploration with the environment.

More recently, some authors have also used the aforementioned affordances to tackle the high-dimensional data issue of RL. Ahmed *et al.*, proposed an end-to-end AD system comprised of two major components: supervised network and a Deep RL agent (DDPG) [37]. The supervised network encodes RGB images into a set of affordances. Subsequently, the Deep RL transforms the affordances, vehicle measurements and a navigation command into control commands. A simplified version of the architecture of this solution is depicted in Figure 5.

Toromanoff *et al.* also used affordances in a RL pipeline [77]. The first component of the system is an encoder trained to predict affordances such as distance to centerline and traffic-lights. Then, the output features of the encoder are conveyed into the RL, instead of the affordances. The authors have named this approach *implicit affordances*, since it uses the information that predicted the affordances and not the affordances itself.

There are mainly two differences between the approaches described in the last paragraphs: how to encode the raw data from the sensors? and how to learn a driving policy based on the encoded data? Whether using feature vectors from encoder neural networks [14], [60], [64], [65] or affordances [58], [59], [77], the goal is to produce a low dimensional intermediate representation of the environment to simplify further processing. In this aspect, we believe that feature vectors are preferable to affordances. Using feature vectors, the relevant features are learned by the model, whilst in affordances, the relevant features to be learned are user-defined, which can introduce human bias into the system. Furthermore, it is questionable whether an approach based on affordances should be considered end-to-end or not, because it presents the same problems of modular approaches: human definition of affordances; diverse ways to integrate the affordances in a learning method; error propagation due to incorrect prediction of the affordances. Regarding the learning of the driving policy, there are two distinct approaches: IL and RL. Based on the ratio of IL/RL in recent approaches, it is inconclusive to assess what is the leading learning method in urban environments. As will be discussed in Section III, both approaches achieve satisfactory results and therefore further research is required to investigate which learning method is more suitable for AD in urban environments.

B. INPUT SENSOR MODALITIES

The majority of end-to-end systems rely only on vision [39], [44], [77], using only one camera to predict the control commands. However, in urban environments the single modality configuration is usually insufficient to produce a robust and reliable AD system [60]. Furthermore, AD in urban environments requires navigation from one point to another, and therefore, additional navigation inputs are often mandatory. Xiao *et al.*, in [27], performed a comparison between single and multimodal end-to-end AD systems. They used

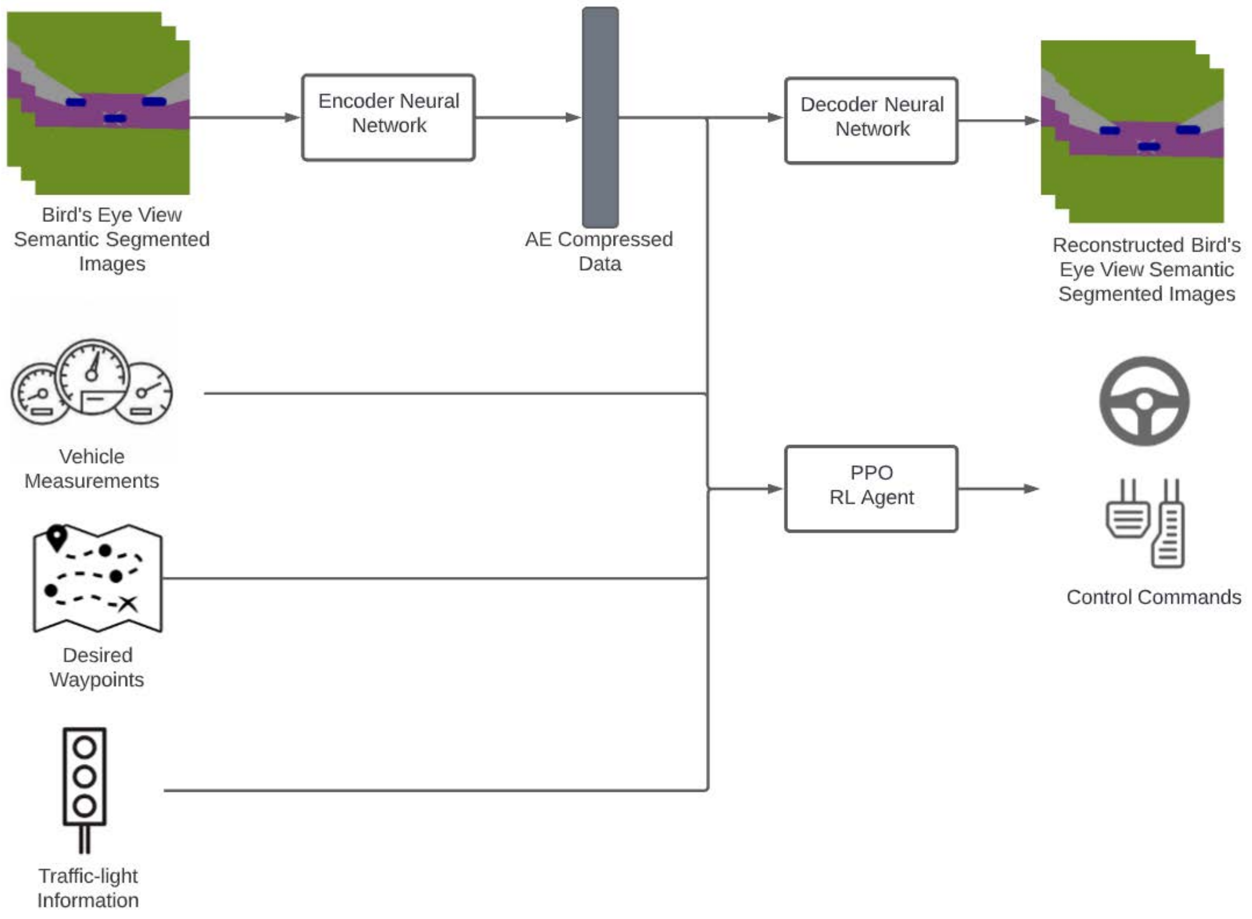


FIGURE 4. Simplified version of the system proposed by Agarwal *et al.* [14]. The system receives as inputs three bird's eye view semantic segmented images, vehicle measurements, navigation commands, desired waypoints and traffic-light information. Each input is provided directly by CARLA. The images are processed by an autoencoder (AE) [76] that produces a compressed version of the images (AE Compressed Data). The compressed data is combined with the remaining inputs to form the RL state representation. The RL agent it then responsible to produce the control commands. In addition to the control commands, the system also outputs reconstructed bird's eye view semantic segmented images through the decode of the compressed data.

RGB images and depth information as the sensor modalities and demonstrated that multimodality is beneficial to end-to-end systems, outperforming single modality configurations. Regarding the fusion scheme, the authors concluded that the early fusion, i.e. increasing the number of channels from three (RGB) to four (RGBD), was the one that achieved the best results.

Regarding the navigation inputs, Codevilla *et al.* performed a study about the incorporation of navigation commands into the AD system [39]. Navigation commands are referred to as an indication about the future action taken by the agent, such as "go right" or "turn left on the intersection." These commands can be generated by high-level route planners [79], [80] or by humans. Codevilla *et al.* implemented two different architectures: *command input* (see Figure 6) and *branched* (see Figure 7). In *command input*, the network takes the navigation commands as input, together with the raw images and some vehicle measurements. These three inputs are processed independently, and then the combination of the three results are delivered to a control model to produce the control commands. On the other hand, in the *branched* architecture only the image and the measurements are conveyed into the network as inputs. In this case, the control module is replaced by a set of branches. The role of the navigation command is to select which branch should be active, and therefore the navigation command can be seen as a switch. Results demonstrated that the *branched* architecture performed significantly better than the command input approach and other baseline approaches.

Huang *et al.*, in [66], proposed a multimodal system that receives RGB images and depth information as input. This information is encoded by a neural network and is processed by a conditional driving policy. The conditional driving policy is a branched fully connected network, and in addition to receiving the encoded data, also receives a navigation command. The navigation command, as in [39], activates the corresponding branch, and each branch is a neural network that produces the control commands.

In [58], Sauer *et al.* also used the concept of specialized neural networks, but instead of predicting conditional

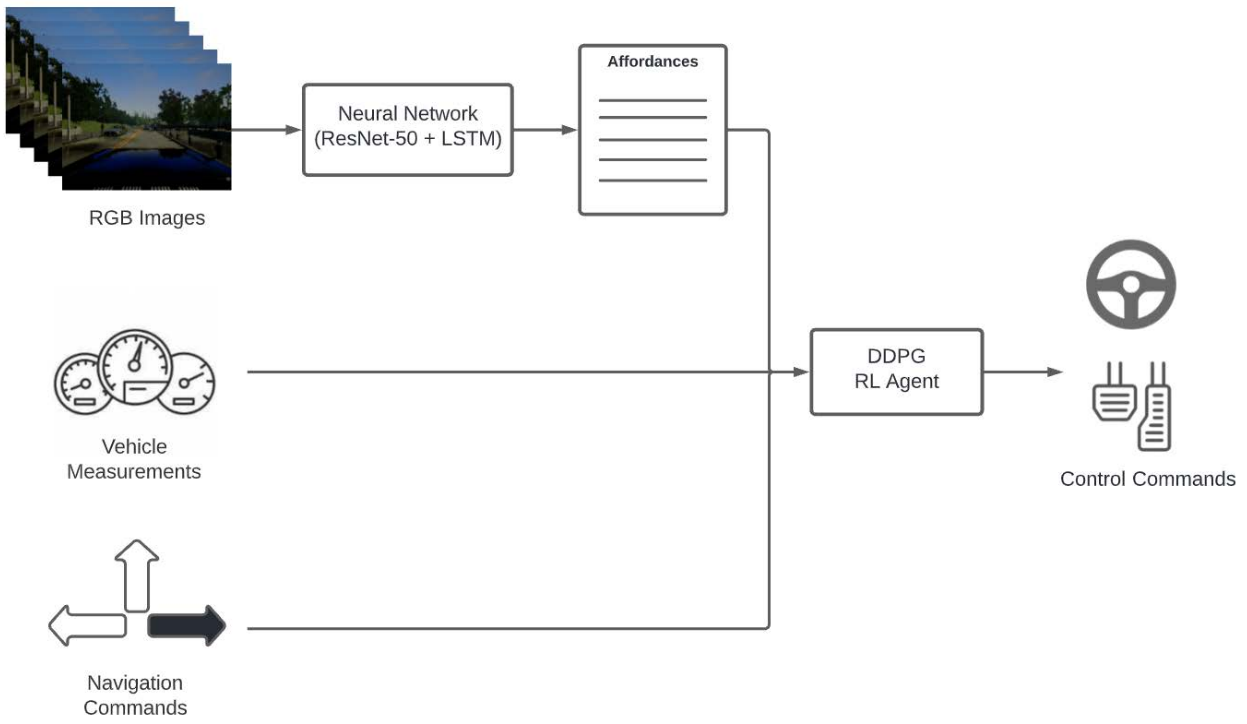


FIGURE 5. **Simplified version of the system proposed by Ahmed *et al.* [37]. The system receives as input a stack of RGB images, vehicle measurements and navigational commands. Firstly, a residual network (ResNet-50 [78]) is used to extract the features of the images, and then LSTM units are employed to model the dependencies between successive frames. At the end of this processing, the affordances are predicted. The DDPG RL agent receives a vector that comprises the affordances, vehicle measurements and a navigation command, and produces as output the control commands.**

control commands, the system predicts conditional affordances. Their architecture, Figure 2, receives a navigation command that selects a specific neural network to predict the affordances. The authors reported that training specialized neural networks for each navigation command leads to better performance than training neural networks that use navigation commands as inputs.

As an alternative to navigation commands, some authors have used the desired trajectory, provided by a global planner, as input [14], [60], [81]. Usually the trajectory is conveyed into the system under the form of waypoints. For instance, in [60], Prakash *et al.* used GPS coordinates provided by CARLA as input, to predict local waypoints (see Figure 3). The GPS coordinates provided by CARLA are relatively sparse and can be spaced hundreds of meters apart. Conversely, the waypoints predicted by the neural network refer to the trajectory that the agent should follow in subsequent timestamps. Cai *et al.*, in [81], instead of using the global planner from CARLA, implemented the A^* [82], [83] algorithm to plan the coarse route from the initial point to the destination point based on static maps. The waypoints provided by global planners do not consider dynamic objects nor information regarding traffic-lights. Its only purpose is to provide a global trajectory based on static elements of the environment.

In the last few years, several authors suggested that, for urban environments, the integration of RGB cameras and LiDAR is essential [60], [65], [81]. These modalities are often seen as complementary, where the RGB cameras provide information about the road and visual aspects of the scene, such as traffic-lights, and the LiDAR provides accurate spatial information in 360 degrees [65]. Prakash *et al.*, in [60], as discussed above, proposed the usage of the attention mechanism of transformers to integrate RGB images and LiDAR data. Authors argued that it is a robust and flexible way of integrate different modalities of sensors in general, and not only the RGB camera and LiDAR (see Figure 3). Chen *et al.* also used RGB images and LiDAR data as inputs. Instead of using the raw point clouds from the LiDAR as input, they performed a prepossessing step, where the raw point clouds are converted into a 2D LiDAR bird's eye image, which is then conveyed into the network [65].

Cai *et al.* in [81] explored even further the combination of multiple modalities, where they proposed an end-to-end AD system that receives RGB camera, LiDAR and RADAR data as input. This multimodal information is processed by uniform alignment and projection onto the image plane. In addition to cameras, LiDARs and RADARs, some authors also use HD maps as inputs. For example, Zeng *et al.* proposed a system that takes LiDAR data and HD maps as inputs of the network [25].

Multiple authors have also used high-level measurements about the state of the vehicle, such as current velocity or acceleration [37], [39], [44]. When the model only considers

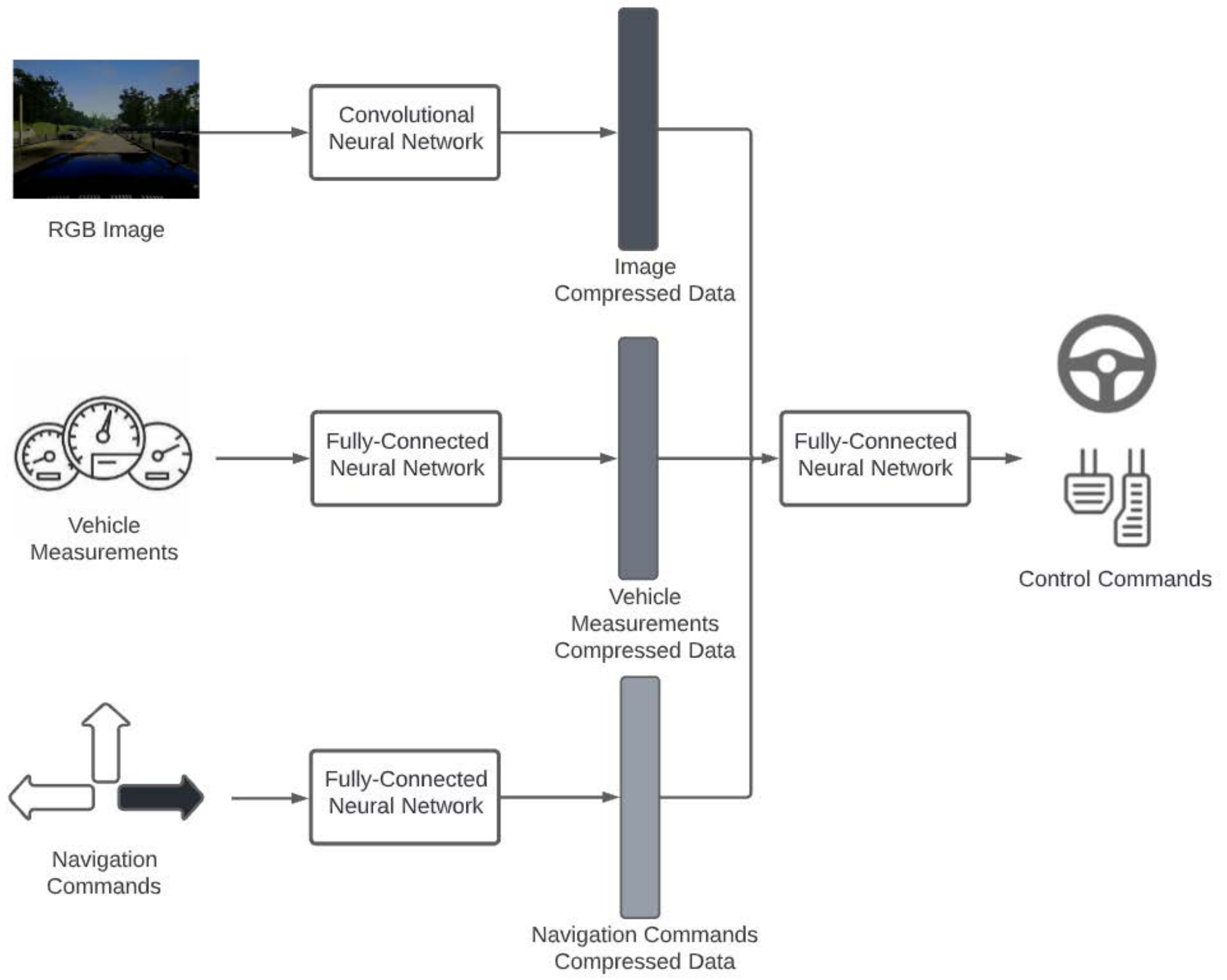

RGB Image
Convolutional Neural Network
Image Compressed Data
Vehicle Measurements
Fully-Connected Neural Network
Vehicle Measurements Compressed Data
Fully-Connected Neural Network
Control Commands
Navigation Commands
Fully-Connected Neural Network
Navigation Commands Compressed Data

FIGURE 6. Simplified version of the *command input* architecture proposed by Codevilla *et al.* [39]. The system takes a RGB image as an input, alongside with vehicle measurements and navigation commands. These inputs are processed independently by three neural networks: a convolutional neural network and two fully-connected neural networks, respectively. The outputs of these neural networks are then concatenated to form the input of the control module, which is a fully-connected neural network. At the end, the control module produces the control commands.

one frame to make a decision, the usage of the current velocity can be highly useful [24]. However, in IL approaches, if the model receives the current speed and predicts the speed for the next timestamp as one of the outputs, there is high changes of causing the inertia problem. In most cases, the current speed and the speed in the next timestamp are highly correlated, which can induce the model to only consider the current speed to predict the speed in the next timestamp. This can seriously hamper the effectiveness of the agent, because it can lead to agents that are reluctant to change the velocity.

Although most of end-to-end approaches use a RGB camera as the only input modality, several works reported that multimodality outperforms single modality configurations in urban environments [39], [60], [65], [81]. Based on latest end-to-end AD papers, it is evident that researchers are moving from single modality to multimodal configurations (see Table 1). From all the available sensors, LiDAR appears to be the one that can add more valuable information to the RGB camera. Furthermore, as AD in urban environments deals with the problem of navigating from one location to another, navigation commands or desired trajectories, are often used.

C. OUTPUT MODALITIES

The majority of the end-to-end AD systems produce the steering angle and the speed for the next timestamp as outputs [27], [58], [64], [77]. These properties are easily obtained from a vehicle and, therefore, can be used as labeled data for IL approaches. Usually, traditional PID controllers [84] are required to convert the steering angle and speed outputted by the network into acceleration/brake and steering torque of the vehicle [14], [64], [66]. The downside of these approaches is that it is very difficult to comprehend the decisions taken by the model. In other words, when the model produces an incorrect driving decision, it is not possible to understand the reason for that decision.

In recent years, some authors have explored the usage of trajectories (waypoints) as the output modality of the system [25], [60]. For instance, Prakash *et al.* proposed

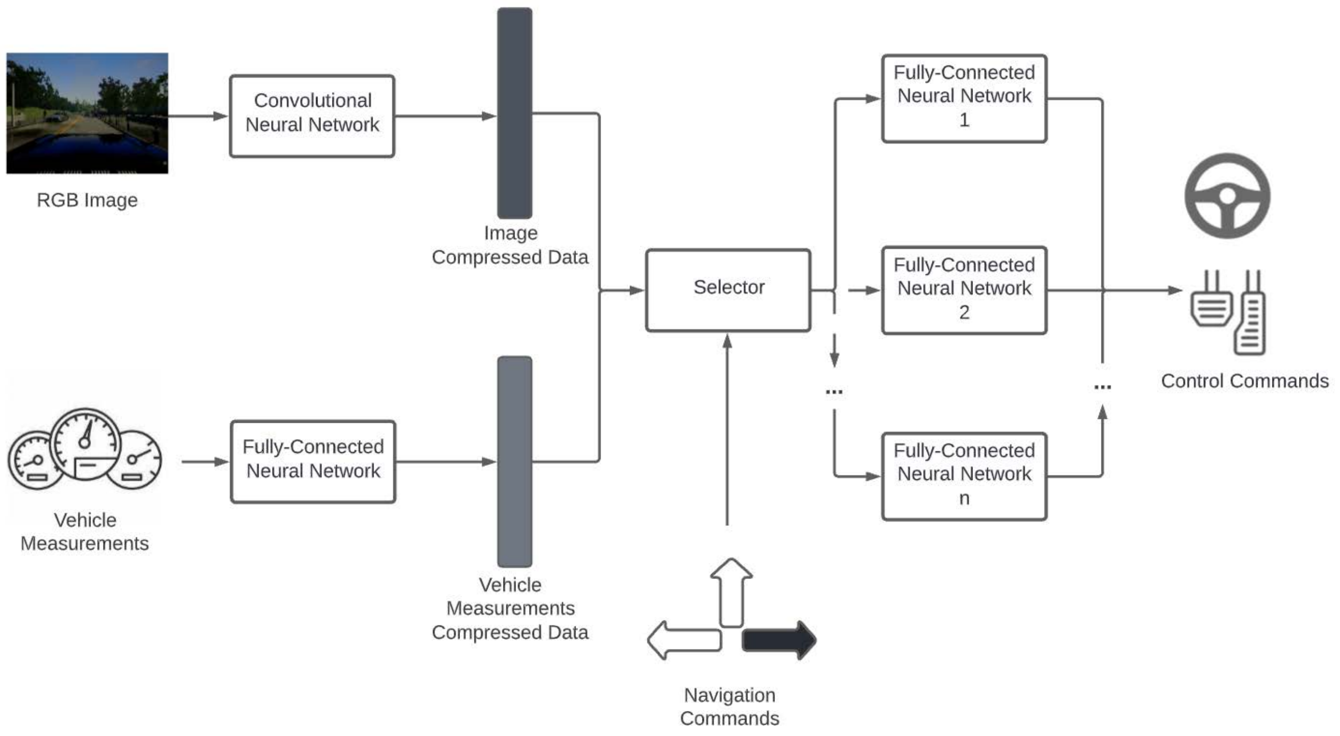

RGB Image
Convolutional Neural Network
Image Compressed Data
Vehicle Measurements
Fully-Connected Neural Network
Vehicle Measurements Compressed Data
Selector
Navigation Commands
Fully-Connected Neural Network 1
Fully-Connected Neural Network 2
...
Fully-Connected Neural Network n
Control Commands

FIGURE 7. **Simplified version of the *branched* architecture proposed by Codevilla *et al.* [39]. The system receives as inputs a RGB image, vehicle measurements and a navigation command. The RGB image and the vehicle measurements are processed independently by a convolutional neural network and a fully-connected neural network, respectively. The outputs of these neural networks are concatenated to form the input of the next stage. The navigation command is used as a switch that selects which fully-connected neural network should process the concatenated data and therefore produce the control commands.**

a system that predicts waypoints for the future 4 timestamps [60] (see Figure 3). Zeng *et al.*, in [25], proposed a system that produces 3D detections as well as their future trajectories, and then uses a planner to choose from the set of possible trajectories the one that minimizes a predefined cost. In this output modality, it is also necessary to implement a controller (usually a PID) that generates low-level steering and acceleration/braking commands to reach the desired trajectory [60], [72]. As an alternative to PID controllers, Gutiérrez *et al.* proposed a modular and scalable waypoint tracking controller, fully integrated in ROS [85]. One advantage of outputting waypoints, instead of the steering angle and velocity commands, is that the model is required to plan the action for the future timestamps, and not only for the next one. This long-term planning increases the robustness of the driving policy because it converts a reactive agent into a planning agent. Another advantage concerns interpretability: it is much easier to interpret and analyze waypoints rather than momentary steering and velocity commands. Using waypoints, it is possible to convey the intentions of the system.

In order to further increase the interpretability of the systems, some authors have also used additional outputs. For example, in [65], as discussed above, the system predicts a semantic mask of the scene, and in [66] the system reconstructs semantic images using a scene understanding decoder. Usually, these additional outputs are computed by a separate branch of the network based on an intermediate layer of the network [24]. These outputs are not directly related to the main output of the network, but they provide information about the internal representation of the network, which is immensely useful to comprehend the driving decisions and to explain the failures. The authors, in [59], reported that the joint learning of the main and additional outputs leads to more robust and effective driving policies.

Although the majority of end-to-end AD produces steering angle, throttle, and braking as final output of the network, the most promising output modality is waypoints. The long-term planning and interpretability make waypoints more suitable for AD, especially for urban environments. To further increase the level of interpretability, the usage of additional outputs, learned in a joint learning mechanism, is highly recommended.

III. EVALUATION

This section focuses on quantitatively evaluating the results from the approaches described throughout the document. However, to additionally provide a comparison between end-to-end approaches and modular approaches, we have also included a modular approach, proposed by Dosovitskiy *et al.* [54], in this section. This approach divides the driving task into three modules: perception, planning, and continuous control. The perception module uses semantic segmentation to estimate lanes, dynamic objects, and other hazards. The planning module consists of a rule-based state machine that implements a driving policy specially designed for urban environments. Finally, the continuous control module is a PID controller that produces the steering, throttle, and brake commands.

Table 1 depicts the summary of the aforementioned approaches divided by the three points of discussion: architecture, inputs and outputs. This table follows a chronological order to facilitate the extraction of possible trends. Regarding the architectures, there is not a clear trend. Thus, it is not possible to claim which approach, IL or RL, is the most suitable for urban environments. Concerning the inputs, in 2018, none of the approaches used LiDAR as data source, while in 2021, four out of seven used LiDAR, which clearly shows the applicability and usefulness of LiDAR in urban environments. Finally, regarding the outputs, in 2018, the only output modality was steering angle, throttle and braking, while in 2021, some authors have used future waypoints as the output of the models. As stated before, around 65% of the approaches studied are from 2020 or 2021, which indicates that, more than ever, researchers are focused on applying end-to-end AD system in urban environments.

Given that this paper focuses on evaluating end-to-end AD systems tested on CARLA, we used two benchmarks (*CoRL2017* and *NoCrash*) of the simulator to accurately compare the performance of the approaches. An additional advantage of comparing all approaches within the same benchmarks is that all approaches use the same sensors, and therefore share the technical specifications. For example, for all approaches, the camera sensor provides images with 800 $\times$ 600 pixels and has a horizontal field of view of 90 degrees. The LiDAR sensor is a Velodyne 64, with a range of 10 meters, and covers a horizontal field of view of 360 degrees. Lastly, the RaDAR sensor has a range of 100 meters and a horizontal field of view of 30 degrees.

Our goal was to compare all algorithms listed in Table 1; however, some works were evaluated in different benchmarks, reason why they are not considered in the evaluations. Notwithstanding, the majority of the approaches were evaluated in the aforementioned benchmarks. In both benchmarks, there are two towns: Town 01 and Town 02.

Town 01 (see Figure 8) consists of 2.9 Km of road with 11 intersections, and it is used to train the models. We will refer to this as the training conditions. Town 02 (see Figure 9) consists of 1.4 Km of road with 8 intersections, and it is used to test the models under different weather conditions when comparing with the ones used in the training conditions. We will refer to this as the test conditions. It is worth noting that Town 01 and Town 02 are simplified versions of urban environments. Their layout consists of several T-junctions with traffic lights. There are more realistic scenarios in CARLA, as is the case of Town 3, which contains 5-lane junctions, a roundabout, and a tunnel. However, only a few approaches have tested their algorithms under such conditions and therefore it is unsuitable for comparisons, at least for now.

The *CoRL2017* benchmark is the original CARLA benchmark, and the goal is to navigate from a starting point to a destination point using a route planner [54]. There are four driving tasks, with increasing difficulty levels: Straight, One Turn, Navigation and Navigation with Dynamic Obstacles (see Figure 8). In the Straight task, the destination is straight ahead of the starting point, and there are no dynamic obstacles in the environment. In the One Turn task, the destination is one turn away from the starting point and the environment contains no dynamic obstacles. In the Navigation task there is no restriction on the location of the destination and starting point, and once again, there are no dynamic obstacles. Finally, in the Navigation With Dynamic Obstacles, the scheme is the same as the previous task, but with dynamic obstacles (cars and pedestrians) introduced in the scene. For each task, for an episode to be considered successful the agent must reach the destination within a time limit, defined as the time required to reach the destination along the optimal path at a speed of 10 km/h [54]. Driving infractions, such as collisions or driving on the sidewalk do not lead to the termination of the episode, which means that the primary objective of this benchmark is to evaluate skills such as lane following and performing 90 degrees turns [55].

Table 2 depicts, for the *CoRL2017* benchmark, the percentage of successfully completed episodes out of 25, for each task in training conditions, using 10 approaches. In general, the results of each approach are worsening with the increase in difficulty of the task. However, both Chen *et al.* [36] and Toromanoff *et al.* [77] achieved the maximum score in all tasks. Huang *et al.* [66] and Agarwal *et al.* [14] also achieved excellent results. As expected, the overall results are very satisfactory, because the agents are being tested under the same conditions in which they were trained (same town and same weather).

Table 3 has the same structure as Table 2, but refer to the testing conditions. Chen *et al.* [36] kept the maximum score, while Toromanoff *et al.* [77] suffered a slightly decreased in the score. Huang *et al.* [66] also achieve very satisfactory results. As expected, the scores in training conditions are much better than in testing conditions. Nevertheless, the overall results under testing conditions are surprisingly good, suggesting that the models were able to generalize the driving policy to unknown scenarios.

The main difference between the system proposed by Chen *et al.* and the others is that the system uses a teacher that has access to privileged information to train a vision-based agent. Based on the results described above, there are strong indications to conclude that this learning method is immensely effective considering the evaluation metrics of the *CoRL2017* benchmark.

Based on Table 2 and Table 3, it is also possible to unmask some of the limitations of modular approaches, namely poor generalization, and error propagation. The testing results were considerably worse than the training conditions because the perception module fails systematically under complex and unseen conditions. When the perception module fails,

TABLE 1. Contributions of AD systems in urban environments, described in terms of: architecture, inputs and outputs. The table follows a chronological order of the papers.

Approach	Architecture	Inputs	Outputs
Dosovitskiy et al. 2017 [54]	modular architecture: perception, planning, and continuous control	RGB image navigation command	steering angle throttle brake
Sauer et al. 2018 [58]	conditional affordances prediction followed by a controller	RGB image navigation command	steering angle throttle brake
Mehta et al. 2018 [59]	IL with affordances prediction	RGB images navigation command	steering angle throttle brake
Liang et al. 2018 [44]	IL followed by RL	RGB image vehicle measurements navigation command	steering angle throttle brake
Codevilla et al. 2018 [39]	IL with navigational command as input (command input)	RGB image vehicle measurements navigation command	steering acceleration
Codevilla et al. 2018 [39]	conditional IL (branched)	RGB image vehicle measurements navigation command	steering acceleration
Chen et al. 2019 [36]	privileged IL	RGB image vehicle measurements navigation command	steering angle throttle brake
Toromanoff et al. 2020 [77]	RL with implicit affordances	RGB images navigation command	steering angle throttle
Xiao et al. 2020 [27]	multimodal conditional IL	RGB image depth information/LiDAR vehicle measurements navigation command	steering angle throttle brake
Cai et al. 2020 [81]	probabilistic motion planning using multimodal information	RGB image LiDAR RADAR vehicle measurements desired trajectory	steering angle throttle brake
Chen et al. 2020 [64]	conditional DQN	RGB image desired trajectory	steering angle acceleration
Chen et al. 2021 [65]	latent RL	RGB image LiDAR desired trajectory	steering angle throttle brake
Zeng et al. 2021 [25]	interpretable IL	LiDAR HD map	waypoints
Huang et al. 2021 [66]	IL with scene understanding	RGB image depth information/LiDAR navigation command	steering angle speed
Ahmed et al. 2021 [37]	affordances prediction followed by RL	RGB image vehicle measurements navigation command	steering angle acceleration brake
Agarwal et al. 2021 [14]	combination of modular and RL	segmented bird's eye view traffic-light information vehicle measurements desired trajectory	steering angle speed
Prakash et al. 2021 [60]	IL with multimodal fusion transformer	RGB image LiDAR vehicle measurements desired trajectory	waypoints
C. Huang et al. 2021 [72]	deductive RL	RGB image navigation command	steering angle acceleration brake

the planning module is not able to produce a reliable path and therefore the continuous control module is unable to produce accurate control commands. However, we want to stress that these results do not allow us to conclude that current end-to-end approaches are superior to modular approaches. The modular approach considered is from 2017, and it is expected that novel and better ones have been developed in the meantime. Furthermore, such bold claim would require an extensive comparison between modular and end-to-end approaches, which is out of the scope of this paper.

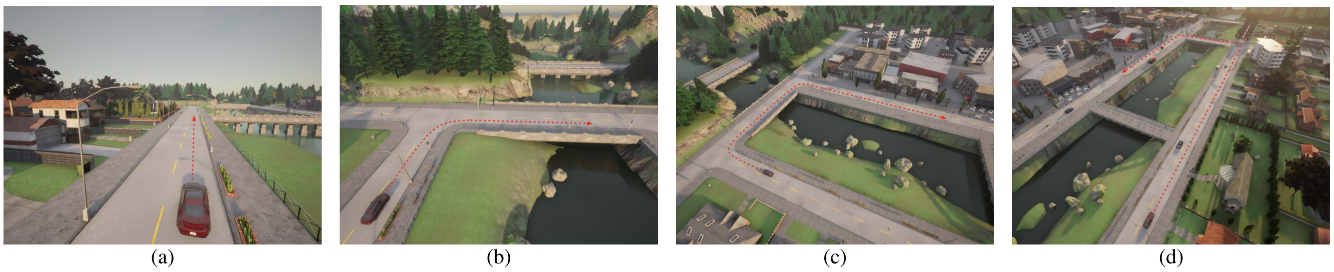


FIGURE 8. Illustration of the four driving tasks of *CoRL2017* benchmark in Town 01. (a), (b), (c), and (d) represents the Straight task, One Turn task, Navigation task, and Navigation with Dynamic Obstacles task, respectively.

TABLE 2. Results of *CoRL2017* benchmark. Each value corresponds to the percentage of successfully completed episodes, for each task in training conditions. Each column corresponds to an approach. Best scores are highlighted in bold.

Task	Dosovitskiy et al. 2017	Liang et al. 2018	Sauer et al. 2018	Chen et al. 2019	Toromanoff et al. 2020	Xiao et al. 2020	Huang et al. 2021	Ahmed et al. 2021	Agarwal et al. 2021	C. Huang et al. 2021
Straight	98	98	**100**	**100**	**100**	98	**100**	**100**	99	**100**
One Turn	82	97	97	**100**	**100**	99	**100**	98	**100**	98
Navigation	80	93	92	**100**	**100**	93	**100**	93	**100**	93
Nav. dynamic	77	82	83	**100**	**100**	89	98	94	**100**	75

TABLE 3. Results of *CoRL2017* benchmark. Each value corresponds to the percentage of successfully completed episodes for each task in testing conditions. Each column corresponds to an approach. Best scores are highlighted in bold.

Task	Dosovitskiy et al. 2017	Liang et al. 2018	Sauer et al. 2018	Chen et al. 2019	Toromanoff et al. 2020	Xiao et al. 2020	Huang et al. 2021	Ahmed et al. 2021	Agarwal et al. 2021	C. Huang et al. 2021
Straight	50	98	94	**100**	**100**	97	**100**	97	**100**	**100**
One Turn	50	82	72	**100**	**100**	83	**100**	95	98	82
Navigation	47	68	68	**100**	**100**	93	**100**	92	**100**	68
Nav. dynamic	44	62	64	**100**	98	94	94	91	99	60

TABLE 4. Results of *NoCrash* benchmark. Each value corresponds to the percentage of successfully completed episodes for each task in training conditions. Each column corresponds to an approach. Best scores are highlighted in bold.

Task	Chen et al. 2019	Toromanoff et al. 2020	Huang et al. 2021	Ahmed et al. 2021	Agarwal et al. 2021
Empty	**100**	**100**	**100**	**100**	**100**
Regular	**99**	96	91	97	90
Dense	**95**	70	61	70	87

The *NoCrash* benchmark comprises three tasks with distinct levels of difficulties: Empty, Regular, and Dense (see Figure 9) [55]. The Empty task corresponds to an uninhibited town with no dynamic obstacles. The Regular task consists of a town with a moderate number of vehicles and pedestrians. Finally, the Dense task corresponds to a town with many vehicles and pedestrians. This benchmark is more recent than *CoRL2017*, so only a few approaches have assessed their algorithms in these tasks. The core idea of this benchmark is to introduce a new aspect in the evaluation of the agent: the response to dynamic objects. Measuring an agent solely based on whether it navigates to the destination point without considering what happened in the meantime is a limited judgement of its driving capabilities. As such, in the *NoCrash* benchmark, for an episode to be considered successful, the agent must reach the destination under the time limit with the additional constraint of not colliding with any object. This benchmark is a more complete assessment of the driving performance, although there are several other aspects of urban driving which are yet to be considered. Some examples are respecting traffic lights and abiding to speed limits.

Table 4 contains the percentage of successfully completed episodes out of 25, for each task in training conditions, for 5 approaches. The overall scores, in this table, clearly indicate that this benchmark is more difficult than the *CoRL2017* benchmark. Once again, Chen *et al.* [36] achieved the best results in all tasks. Without considering Chen *et al.* [36], all other approaches presented relatively low scores in the Dense task, even though the conditions were exactly the same as in training.

Table 5 contains the results for the testing conditions, and here, Ahmed *et al.* [37] achieved the best results, surpassing Chen *et al.* [36] in two tasks. Agarwal *et al.* [14] also achieved excellent scores in these conditions, achieving the same score as Ahmed *et al.* [37] in the Regular and Dense tasks. In the Empty and Regular tasks, all approaches have achieved good results, always above 80%. The poor results showed in the Dense task in training conditions were amplified in test conditions. Toromanoff *et al.* [77] and Huang *et al.* [66] achieved a score of less than 50% successful episodes, which clearly proves that current end-to-end AD systems have a considerable difficulty in dealing with dense urban environments.

Results from the quantitative evaluation are inconclusive in what concerns the best architecture for urban environments. For the *CoRL2017* benchmark, the most effective approach was IL-based architecture (Chen *et al.* [36]), while for the *NoCrash* benchmark the most effective approach

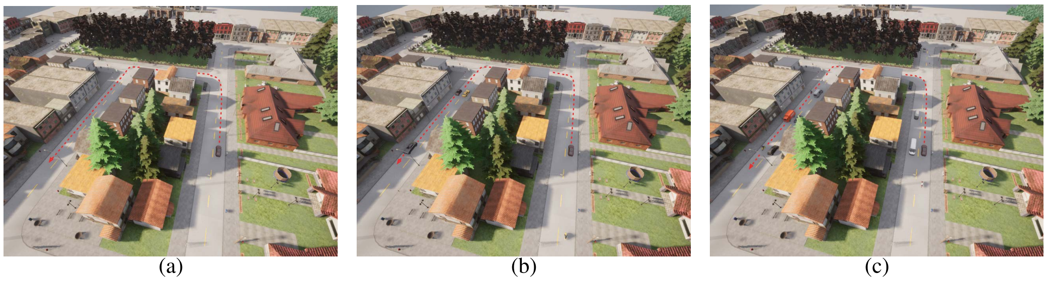


FIGURE 9. Illustration of the three driving tasks of *NoCrash* benchmark in Town 02. (a), (b), and (c) represents the Empty task, Regular task, and Dense task, respectively.

TABLE 5. Results of *NoCrash* benchmark. Each value corresponds to the percentage of successfully completed episodes for each task in testing conditions. Each column corresponds to an approach. Best scores are highlighted in bold.

Task	Chen et al. 2019	Toromanoff et al. 2020	Huang et al. 2021	Ahmed et al. 2021	Agarwal et al. 2021
Empty	**100**	99	**100**	**100**	**100**
Regular	94	87	82	**96**	**96**
Dense	85	42	43	**87**	82

was RL-based architecture (Ahmed *et al.* [37]). Table 2 and Table 3 suggest that current end-to-end AD systems are already very effective at tackling simple driving tasks, such as Straight and One Turn. These approaches also appear to be relatively well prepared for Navigation tasks. Conversely, Table 4 and Table 5 suggest that current end-to-end AD systems are not yet prepared to cope with dense traffic situations, characteristic of many real-world cities. The intricate problem of dealing with multiple agents and their unpredictability appears to be most challenging problem for end-to-end AD systems in urban environments. These results clearly indicate that further research is required in this area to tackle the dense urban environments.

IV. CONCLUSION

This paper is the first evaluation of end-to-end AD systems in urban environments. We performed a detailed analysis of 17 approaches, based on three key points: architecture, inputs and outputs. Two CARLA benchmarks were used to quantitatively compare the approaches: *CoRL2017* and *NoCrash*. For the *CoRL2017* benchmark, we compared 10 approaches both in training and testing conditions, where we show that the solution proposed by Chen *et al.* [36] achieved an excellent score of 100% in all tasks considered. Furthermore, results also suggest that modular approaches suffer from poor generalization and error propagation. For the *NoCrash* benchmark, we compared 5 approaches both in training and testing conditions. In the training conditions Chen *et al.* [36], once again, achieved the best results, while in the testing conditions the approach proposed by Ahmed *et al.* [37] achieved the best scores. From the analyses of Table 4 and Table 5, we can conclude that the current end-to-end AD systems are not prepared to deal with dense traffic, suggesting that additional research is required.

The use of simulators, such as CARLA, clearly plays a key role in the training of AD systems. Learning to drive in the simulation domain presents innumerous advantages: avoiding human casualties and expensive crashes, changing lightning and weather conditions, and reshaping structural elements of the scenes. It is also possible to reconstruct rare and dangerous scenarios that foster the learning of a robust and safer driving policy [86]. Furthermore, in simulators, we can exploit privileged information, such as the pose of the vehicle and semantic information, that would otherwise not be possible to have. However, the carry over from simulation to reality poses significant problems, mainly due to the simulation-reality gap [87]. The process of transferring a model trained in simulation to the real world is referred to as transfer learning and, in the past years, several approaches have been proposed to tackle this issue [86]–[89]. Due to the novelty of end-to-end AD systems, none of the works addressed in this paper have been validated in real-sized vehicles in the real world. This is an important area to address in the near future.

The complexity of driving situations that can be encountered makes it difficult to ensure that current end-to-end systems can be safely deployed on public roads. The lack of interpretability, common to most end-to-end systems, is also a characteristic that can delay the implementation of these systems in the real world. For autonomous driving systems to be considered safe and viable, they must not only have fewer accidents when compared to humans, but must also convey the reasoning behind their driving decisions. The feeling of safety in a self-driving car that is completely opaque in terms of reasoning can only be achieved once we learn to fully trust autonomous systems, which is something that comes with years and years of safe driving behaviour.

The next paragraphs of this section offer a critical analysis of end-to-end AD, focused on the three points considered in this paper: architecture, inputs and outputs.

IL is the most dominant strategy for end-to-end AD systems. However, there are some fundamental limitations that should be highlighted. First, IL is limited to the average of the training data, i.e., the model will learn the most repeated features in the data, ignoring the rare cases. In driving, a rare case might be a child running towards a ball in the middle of the road, and that is not a case that we are willing to ignore. Second, and from our point of view, the most limiting factor of IL concerns the limitation of the teachers. Since the goal of AD is to obtain systems that can drive better than humans, we cannot be limited by demonstrations from humans, otherwise, the best we can hope to achieve is the same driving performance of humans, and as we have noted, it is not enough. For those reasons, we believe that the most promising architecture is **RL-based**. In recent years, agents trained with RL techniques have already achieved superhuman performance, as in the case of game playing [90] and robotics [91].

The most dominant strategy for input sensor modalities, in end-to-end AD, is to use RGB cameras. Most proponents of those configurations claim that it is the most affordable way to deploy AD systems. However, based on the works described above, **multimodality** appears to be much more effective than the single modality RGB cameras. Furthermore, as demonstrated by Chen *et al.* [65], RGB images and LiDAR information are complementary, providing a more detailed understanding of the scene. Concerning navigation information, we believe that momentary indications, such as ''turn right'', can induce ambiguity in a scene where multiple roads may lead to the right. Based on that, it seems that the model should receive the **desired trajectory**, provided by a global planner, as input. As it is widely used, **vehicle measurements** should also be taken into consideration.

The most common outputs in end-to-end AD systems are steering angle, throttle and braking. At first glance, this is the logic approach since we are dealing with end-to-end systems. However, we believe that it is not the most promising approach. Momentary control commands are very difficult to interpret, and to explain the decisions taken by the model. On the other hand, as used in [25], [60], **waypoints** are better to convey the intentions of the system. Furthermore, predicting waypoints forces the model to plan the long-term trajectory instead of reacting to momentary inputs. When using waypoints, both the lateral and longitudinal movement are explained: the lateral movement is explicit in the waypoints and the longitudinal movement can be easily explained by the distance between successive points. To further increase the visualization of the intentions of the model, and to increase the effectiveness of the driving policy, **additional outputs**, learned in a joint mechanism, are highly recommended.

REFERENCES

[1] J. Laconte, A. Kasmi, R. Aufrère, M. Vaidis, and R. Chapuis, ''A survey of localization methods for autonomous vehicles in highway scenarios,'' *Sensors*, vol. 22, no. 1, p. 247, Dec. 2021.

[2] E. Horváth, C. Pozna, and M. Unger, ''Real-time LIDAR-based urban road and sidewalk detection for autonomous vehicles,'' *Sensors*, vol. 22, no. 1, p. 194, Dec. 2021.

[3] Y.-B. Chang, C. Tsai, C.-H. Lin, and P. Chen, ''Real-time semantic segmentation with dual encoder and self-attention mechanism for autonomous driving,'' *Sensors*, vol. 21, no. 23, p. 8072, Dec. 2021.

[4] L. Fridman, E. D. Brown, M. Glazer, W. Angell, S. Dodd, B. Jenik, J. Terwilliger, A. Patsekin, J. Kindelsberger, L. Ding, S. Seaman, A. Mehler, A. Sipperley, A. Pettinato, B. Seppelt, L. Angell, B. Mehler, and B. Reimer, ''MIT advanced vehicle technology study: Large-scale naturalistic driving study of driver behavior and interaction with automation,'' *IEEE Access*, vol. 7, pp. 102021–102038, 2019.

[5] Y. Zhang, H. Chen, L. Steven Waslander, J. Gong, G. Xiong, T. Yang, and K. Liu, ''Hybrid trajectory planning for autonomous driving in highly constrained environments,'' *IEEE Access*, vol. 6, pp. 32800–32819, 2018.

[6] C. Sun, X. Zhang, Q. Zhou, and Y. Tian, ''A model predictive controller with switched tracking error for autonomous vehicle path tracking,'' *IEEE Access*, vol. 7, pp. 53103–53114, 2019.

[7] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah, ''Learning to drive in a day,'' in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 8248–8254.

[8] M. Bansal, A. Krizhevsky, and A. Ogale, ''ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst,'' 2018, *arXiv:1812.03079*.

[9] E. Santana and G. Hotz, ''Learning a driving simulator,'' 2016, *arXiv:1608.01230*.

[10] A. Hu, Z. Murez, N. Mohan, S. Dudas, J. Hawke, V. Badrinarayanan, R. Cipolla, and A. Kendall, ''FIERY: Future instance prediction in bird's-eye view from surround monocular cameras,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15273–15282.

[11] T. Arakawa, ''Trends and future prospects of the drowsiness detection and estimation technology,'' *Sensors*, vol. 21, no. 23, p. 7921, Nov. 2021.

[12] S. Singh, *Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey*, document DOT HS 812 506, Traffic Safety Facts Crash.Stats. Report No National Highway Traffic Safety Administration, Washington, DC, USA, Mar. 2018.

[13] P. Andersson and P. Ivehammar, ''Benefits and costs of autonomous trucks and cars,'' *J. Transp. Technol.*, vol. 9, no. 2, pp. 121–145, 2019.

[14] T. Agarwal, H. Arora, and J. Schneider, ''Learning urban driving policies using deep reinforcement learning,'' in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, Sep. 2021, pp. 607–614.

[15] S. Arshad, M. Sualeh, D. Kim, D. V. Nam, and G.-W. Kim, ''Clothoid: An integrated hierarchical framework for autonomous driving in a dynamic urban environment,'' *Sensors*, vol. 20, no. 18, p. 5053, Sep. 2020.

[16] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*, 1st ed. Cham, Switzerland: Springer, 2009.

[17] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, ''A survey of autonomous driving: Common practices and emerging technologies,'' *IEEE Access*, vol. 8, pp. 58443–58469, 2020.

[18] C. Gómez-Huélamo, J. D. Egido, L. M. Bergasa, R. Barea, E. López-Guillén, F. Arango, J. Araluce, and J. López, ''Train here, drive there: ROS based end-to-end autonomous-driving pipeline validation in CARLA simulator using the NHTSA typology,'' *Multimedia Tools Appl.*, vol. 81, no. 3, pp. 4213–4240, Jan. 2022.

[19] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, ''Towards fully autonomous driving: Systems and algorithms,'' in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2011, pp. 163–168.

[20] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, ''A survey of motion planning and control techniques for self-driving urban vehicles,'' *IEEE Trans. Intell. Vehicles*, vol. 1, no. 1, pp. 33–55, Jun. 2016.

[21] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, and M. Gittleman, ''Autonomous driving in urban environments: Boss and the urban challenge,'' *J. Field Robot.*, vol. 25, pp. 425–466, 2009.

[22] S. Yang, X. Mao, S. Yang, Z. Liu, G. Chen, S. Wang, J. Xue, and Z. Xu, ''Towards a robust software architecture for autonomous robot software,'' in *Proc. 7th Int. Workshop Comput. Sci. Eng., (WCSE)*, Apr. 2017, pp. 1197–1207.

[23] M. Quigley, ''ROS: An open-source robot operating system,'' in *Proc. ICRA*, 2009, pp. 1–6.

[24] A. Tampuu, T. Matiisen, M. Semikin, D. Fishman, and N. Muhammad, ''A survey of end-to-end driving: Architectures and training methods,'' *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 4, pp. 1364–1384, Apr. 2022.

[25] W. Zeng, W. Luo, S. Suo, A. Sadat, B. Yang, S. Casas, and R. Urtasun, ''End-to-end interpretable neural motion planner,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8652–8661.

[26] B. Peng, Q. Sun, S. E. Li, D. Kum, Y. Yin, J. Wei, and T. Gu, "End-to-end autonomous driving through dueling double deep Q-network," *Automot. Innov.*, vol. 4, no. 3, pp. 328–337, Aug. 2021.

[27] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. Lopez, "Multimodal end-to-end autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 537–547, Jan. 2020.

[28] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[29] X.-W. Chen and X. Lin, "Big data deep learning: Challenges and perspectives," *IEEE Access*, vol. 2, pp. 514–525, 2014.

[30] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.

[31] R. Li, Z. Zhao, Q. Sun, C.-L. I, C. Yang, X. Chen, M. Zhao, and H. Zhang, "Deep reinforcement learning for resource management in network slicing," *IEEE Access*, vol. 6, pp. 74429–74441, 2018.

[32] Y. Wang, H. Liu, W. Zheng, Y. Xia, Y. Li, P. Chen, K. Guo, and H. Xie, "Multi-objective workflow scheduling with deep-q-network-based multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 39974–39982, 2019.

[33] X. Li, J. Fang, W. Cheng, H. Duan, Z. Chen, and H. Li, "Intelligent power control for spectrum sharing in cognitive radios: A deep reinforcement learning approach," *IEEE Access*, vol. 6, pp. 25463–25473, 2018.

[34] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, J. A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell, "Learning to navigate in complex environments," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR) Conf. Track*, 2017, pp. 1–16.

[35] O.-E. L. Team, A. Stooke, A. Mahajan, C. Barros, C. Deck, J. Bauer, J. Sygnowski, M. Trebacz, M. Jaderberg, M. Mathieu, N. McAleese, N. Bradley-Schmieg, N. Wong, N. Porcel, R. Raileanu, S. Hughes-Fitt, V. Dalibard, and W. M. Czarnecki, "Open-ended learning leads to generally capable agents," 2021.

[36] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Proc. Conf. Robot Learn. (CoRL)*, 2019, pp. 66–75.

[37] M. Ahmed, A. Abobakr, C. P. Lim, and S. Nahavandi, "Policy-based reinforcement learning for training autonomous driving agents in urban areas with affordance learning," *IEEE Trans. Intell. Transp. Syst.*, early access, Nov. 2, 2021, doi: 10.1109/TITS.2021.3115235.

[38] S. Liu, J. Tang, Z. Zhang, and J.-L. Gaudiot, "Computer architectures for autonomous driving," *Computer*, vol. 50, no. 8, pp. 18–25, Aug. 2017.

[39] F. Codevilla, M. Müller, A. Lopez, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4693–4700.

[40] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," 2016, *arXiv:160407316*.

[41] J. Hua, L. Zeng, G. Li, and Z. Ju, "Learning for a robot: Deep reinforcement learning, imitation learning, transfer learning," *Sensors*, vol. 21, no. 4, p. 1278, Feb. 2021.

[42] W. Zu, H. Yang, R. Liu, and Y. Ji, "A multi-dimensional goal aircraft guidance approach based on reinforcement learning with a reward shaping algorithm," *Sensors*, vol. 21, no. 16, p. 5643, Aug. 2021.

[43] H. Hu, Z. Lu, Q. Wang, and C. Zheng, "End-to-end automated lane-change maneuvering considering driving style using a deep deterministic policy gradient algorithm," *Sensors*, vol. 20, no. 18, pp. 1–22, 2020.

[44] X. Liang, T. Wang, L. Yang, and E. Xing, "CIRL: Controllable imitative reinforcement learning for vision-based self-driving," in *Proc. Eur. Conf. Comput. Vis.* in Lecture Notes in Computer Science, vol. 11211, 2018, pp. 604–620.

[45] H. Yi, E. Park, and S. Kim, "Multi-agent deep reinforcement learning for autonomous driving," *KIISE Trans. Comput. Practices*, vol. 24, no. 12, pp. 670–674, Dec. 2018.

[46] M. Bojarski, P. Yeres, A. Choromanska, K. Choromanski, B. Firner, L. Jackel, and U. Muller, "Explaining how a deep neural network trained with end-to-end learning steers a car," 2017, *arXiv:1704.07911*.

[47] V. Rausch, A. Hansen, E. Solowjow, C. Liu, E. Kreuzer, and J. K. Hedrick, "Learning a deep neural net policy for end-to-end control of autonomous vehicles," in *Proc. Amer. Control Conf. (ACC)*, May 2017, pp. 4914–4919.

[48] Z. Chen and X. Huang, "End-to-end learning for lane keeping of self-driving cars," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2017, pp. 1856–1860.

[49] M. Hesham Eraqi, N. Mohamed Moustafa, and J. Honer, "End-to-end deep learning for steering autonomous vehicles considering temporal dependencies," in *Proc. Conf., 31st Conf. Neural Inf. Process. Syst. (NIPS), MLITS Workshop*, Dec. 2017, pp. 1–8.

[50] S. K. Kwon, J. H. Seo, J.-W. Lee, and K.-D. Kim, "An approach for reliable end-to-end autonomous driving based on the simplex architecture," in *Proc. 15th Int. Conf. Control, Autom., Robot. Vis. (ICARCV)*, Nov. 2018, pp. 1851–1856.

[51] Y. Wang, D. Liu, H. Jeon, Z. Chu, and E. Matson, "End-to-end learning approach for autonomous driving: A convolutional neural network model," in *Proc. 11th Int. Conf. Agents Artif. Intell.*, 2019, pp. 833–839.

[52] M. Campbell, M. Egerstedt, J. P. How, and R. Murray, "Autonomous driving in urban environments: Approaches, lessons and challenges," *Philos. Trans. Roy. Soc. A, Math., Phys. Eng. Sci.*, vol. 368, no. 1928, pp. 4649–4672, Oct. 2010.

[53] M. Aria, "A survey of self-driving urban vehicles development," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 662, Nov. 2019, Art. no. 042006.

[54] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. CoRL*, 2017, pp. 1–16.

[55] F. Codevilla, E. Santana, A. Lopez, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9328–9337.

[56] P. Kaur, S. Taghavi, Z. Tian, and W. Shi, "A survey on simulators for testing self-driving cars," in *Proc. 4th Int. Conf. Connected Auto. Driving (MetroCAD)*, Apr. 2021, pp. 62–70.

[57] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "DeepDriving: Learning affordance for direct perception in autonomous driving," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2722–2730.

[58] A. Sauer, N. Savinov, and A. Geiger, "Conditional affordance learning for driving in urban environments," in *Proc. Conf. Robot Learn. (CoRL)*, Jun. 2018, pp. 237–252.

[59] A. Mehta, A. Subramanian, and A. Subramanian, "Learning end-to-end autonomous driving using guided auxiliary supervision," in *Proc. 11th Indian Conf. Comput. Vis., Graph. Image Process.*, Dec. 2018, pp. 1–8.

[60] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 7073–7083.

[61] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, N. A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2017, pp. 5999–6009.

[62] Y. Gao and D. Glowacka, "Deep gate recurrent neural network," *J. Mach. Learn. Res.*, vol. 63, pp. 350–365, Nov. 2016.

[63] G. Dai, C. Ma, and X. Xu, "Short-term traffic flow prediction method for urban road sections based on space-time analysis and GRU," *IEEE Access*, vol. 7, pp. 143025–143035, 2019.

[64] L. Chen, X. Hu, B. Tang, and Y. Cheng, "Conditional DQN-based motion planning with fuzzy logic for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 4, pp. 2966–2977, Apr. 2022.

[65] J. Chen, S. E. Li, and M. Tomizuka, "Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 5068–5078, Jun. 2022.

[66] Z. Huang, C. Lv, Y. Xing, and J. Wu, "Multi-modal sensor fusion-based deep neural network for end-to-end autonomous driving with scene understanding," *IEEE Sensors J.*, vol. 21, no. 10, pp. 11781–11790, May 2021.

[67] R. F. J. Dossa, S. Huang, S. Ontanon, and T. Matsubara, "An empirical investigation of early stopping optimizations in proximal policy optimization," *IEEE Access*, vol. 9, pp. 117981–117992, 2021.

[68] S. H. Silva, A. Alaeddini, and P. Najafirad, "Temporal graph traversals using reinforcement learning with proximal policy optimization," *IEEE Access*, vol. 8, pp. 63910–63922, 2020.

[69] U. M. Gidado, H. Chiroma, N. Aljojo, S. Abubakar, S. I. Popoola, and M. A. Al-Garadi, "A survey on deep learning for steering angle prediction in autonomous vehicles," *IEEE Access*, vol. 8, pp. 163797–163817, 2020.

[70] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[71] E. Jo, M. Sunwoo, and M. Lee, "Vehicle trajectory prediction using hierarchical graph neural network for considering interaction among multimodal maneuvers," *Sensors*, vol. 21, no. 16, p. 5354, Aug. 2021.

[72] C. Huang, R. Zhang, M. Ouyang, P. Wei, J. Lin, J. Su, and L. Lin, "Deductive reinforcement learning for visual autonomous urban driving navigation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5379–5391, Dec. 2021.

[73] C. E. Van Uden, S. A. Nastase, A. C. Connolly, M. Feilong, I. Hansen, M. I. Gobbini, and J. V. Haxby, "Modeling semantic encoding in a common neural representational space," *Frontiers Neurosci.*, vol. 12, p. 437, 2018.

[74] S. Zheng and H. Liu, "Improved multi-agent deep deterministic policy gradient for path planning-based crowd simulation," *IEEE Access*, vol. 7, pp. 147755–147770, 2019.

[75] Y. H. Xu, C. C. Yang, M. Hua, and W. Zhou, "Deep deterministic policy gradient (DDPG)-based resource allocation scheme for NOMA vehicular communications," *IEEE Access*, vol. 8, pp. 18797–18807, 2020.

[76] Y. Qi, C. Shen, D. Wang, J. Shi, X. Jiang, and Z. Zhu, "Stacked sparse autoencoder-based deep network for fault diagnosis of rotating machinery," *IEEE Access*, vol. 5, pp. 15066–15079, 2017.

[77] M. Toromanoff, E. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for urban driving using implicit affordances," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7151–7160.

[78] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[79] J. R. Sánchez-Ibáñez, C. J. Pérez-del-Pulgar, and A. García-Cerezo, "Path planning for autonomous mobile robots: A review," *Sensors*, vol. 21, no. 23, p. 7898, Nov. 2021.

[80] M. Alharbi and H. A. Karimi, "A global path planner for safe navigation of autonomous vehicles in uncertain environments," *Sensors*, vol. 20, no. 21, p. 6103, Oct. 2020.

[81] P. Cai, S. Wang, Y. Sun, and M. Liu, "Probabilistic end-to-end vehicle navigation in complex dynamic environments with multimodal sensor fusion," *IEEE Robot. Autom. Lett.*, vol. 5, no. 3, pp. 4218–4224, Jul. 2020.

[82] A. K. Guruji, H. Agarwal, and D. K. Parsediya, "Time-efficient A* algorithm for robot path planning," *Proc. Technol.*, vol. 23, no. 1, pp. 144–149, 2016.

[83] S. Erke, D. Bin, N. Yiming, Z. Qi, X. Liang, and Z. Dawei, "An improved A-Star based path planning algorithm for autonomous land vehicles," *Int. J. Adv. Robotic Syst.*, vol. 17, no. 5, Sep. 2020, Art. no. 172988142096226.

[84] B. Hekimoğlu, "Optimal tuning of fractional order PID controller for DC motor speed control via chaotic atom search optimization algorithm," *IEEE Access*, vol. 7, pp. 38100–38114, 2019.

[85] R. Gutiérrez, E. López-Guillén, L. M. Bergasa, R. Barea, Ó. Pérez, C. Gómez-Huélamo, F. Arango, J. del Egido, and J. López-Fernández, "A waypoint tracking controller for autonomous road vehicles using ROS framework," *Sensors*, vol. 20, no. 14, p. 4062, Jul. 2020.

[86] A. Bewley, J. Rigley, Y. Liu, J. Hawke, R. Shen, V.-D. Lam, and A. Kendall, "Learning to drive from simulation without real world labels," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 4818–4824.

[87] M. Müller, A. Dosovitskiy, B. Ghanem, and V. Koltun, "Driving policy transfer via modularity and abstraction," 2018, *arXiv:1804.09364*.

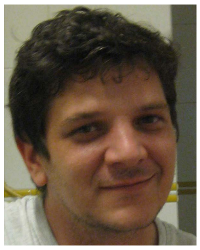
[88] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 23–30.

[89] X. Pan, Y. You, Z. Wang, and C. Lu, "Virtual to real reinforcement learning for autonomous driving," in *Proc. Brit. Mach. Vis. Conf.*, 2017, pp. 1–13.

[90] D. Silver, A. Huang, C. Maddison, A. Guez, L. Sifre, G. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[91] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "QT-opt: Scalable deep reinforcement learning for vision-based robotic manipulation," 2018, *arXiv:1806.10293*.

DANIEL COELHO received the bachelor's and M.Sc. degrees from the University of Aveiro, Aveiro, Portugal, in 2019 and 2021, respectively, where he is currently pursuing the Ph.D. degree in mechanical engineering (specialization in robotics). His research interests include artificial intelligence, autonomous driving, localization, and robotics.



MIGUEL OLIVEIRA received the bachelor's and M.Sc. degrees from the University of Aveiro, Aveiro, Portugal, in 2004 and 2007, respectively, and the Ph.D. degree in mechanical engineering (specialization in robotics, on the topic of autonomous driving systems), in 2013. From 2013 to 2017, he was an Active Researcher with the Institute of Electronics and Telematics Engineering of Aveiro, Aveiro, and with the Institute for Systems and Computer Engineering, Technology and Science, Porto, Portugal, where he participated in several EU-funded projects, as well as national projects. He is currently an Assistant Professor with the Department of Mechanical Engineering, University of Aveiro, where he is also the Director of the Masters in Automation Engineering. He has supervised more than 20 M.Sc. students. He authored over 20 journal publications, from 2015 to 2020. His research interests include autonomous driving, visual object recognition in open-ended domains, multimodal sensor fusion, computer vision, and the calibration of robotic systems.