Addressing Limitations of State-Aware Imitation Learning for Autonomous Driving

Luca Cultrera, Federico Becattini, Lorenzo Seidenari, Pietro Pala, and Alberto Del Bimbo, *Senior Member, IEEE*

***Abstract*—Conditional Imitation learning is a common and effective approach to train autonomous driving agents. However, two issues limit the full potential of this approach: (i) the inertia problem, a special case of causal confusion where the agent mistakenly correlates low speed with no acceleration, and (ii) low correlation between offline and online performance due to the accumulation of small errors that brings the agent in a previously unseen state. Both issues are critical for state-aware models, yet informing the driving agent of its internal state as well as the state of the environment is of crucial importance. In this article we propose a multi-task learning agent based on a multi-stage vision transformer with state token propagation. We feed the state of the vehicle along with the representation of the environment as a special token of the transformer and propagate it throughout the network. This allows us to tackle the aforementioned issues from different angles: guiding the driving policy with learned stop/go information, performing data augmentation directly on the state of the vehicle and visually explaining the model's decisions. We report a drastic decrease in inertia and a high correlation between offline and online metrics.**

***Index Terms*—Autonomous driving, imitation learning, inertia problem.**

I. Introduction

Autonomous driving is becoming a reality. To make this possible, several problems have to be solved, such as perception [1], planning [2], and forecasting [3]. A recent trend that has obtained remarkable results is to directly train driving agents from raw observations with Imitation Learning (IL) [4], [5], i.e. learning to mimic demonstrations from expert human drivers. In this way, the autonomous driving problem is tackled holistically, without having to rely on different heterogeneous modules.

Imitation learning, however, has some limitations. Since the driving capabilities are learned by behavioral cloning, IL models usually lack explicit causal understanding. Rather than rules, relations between patterns are learned, thus making the agent vulnerable to spurious correlations in the data. This phenomenon is known in the literature as *causal confusion* [6]. In particular, when training IL agents for automotive, there is evidence of a special case of causal confusion referred to as the *inertia problem* [5], [7], [8]. The inertia problem stems from a spurious correlation between low speed and no acceleration in the training data, making the driving agent likely to get stuck in a stationary state. As a consequence, when a state-aware agent halts (e.g. at a traffic light or in a traffic jam), it may not move again when it should. For state-awareness here we refer to any source of information that can inform the agent about its halted state, such as a state variable, either explicitly modeled or implicitly inferred, that encodes velocity.

A second issue that limits the applicability of IL is the gap between offline and online driving capabilities [9], [10]. Codevilla et al. [10] showed that there is a low correlation between offline evaluation metrics (e.g. frame-wise Mean Squared Error in steer angle prediction) and the success rate in online driving benchmarks. In online driving, the output of the model influences future inputs, violating the i.i.d. assumption made by the learning framework [11]. Accumulation of small errors thus brings the vehicle into new states, never observed at training time [12]. Similarly to the inertia problem, this issue manifests itself the most in state-aware models: the more variables are observed by the model, such as ego-velocity or previous driving commands, the sparser the coverage of the training data gets, making it more likely to end up in under-represented configurations at driving time.

To summarize, IL agents suffer from ill-distributed training data that presents spurious correlations and domain shift compared to the test set. These issues make it particularly hard to train state-aware agents: using multiple input sources increases the chances of discovering unwanted correlations in the data or of observing under-represented inputs at inference time, for which the agent does not know how to act confidently [11], [12], [13]. In this article, we address these difficulties in training state-aware IL models.

In literature, some attempts to identify and solve these issues have been done. The inertia problem has been addressed by regularizing training through vehicle speed prediction [5], whereas [10] demonstrated the usefulness of two data augmentation approaches to improve offline and online driving capability correlation: first, augmenting the training set with lateral cameras, thus simulating a vehicle with an unusual trajectory, and second, perturbing the driving policy to record samples where

Manuscript received 3 October 2023; revised 1 November 2023; accepted 7 November 2023. Date of publication 23 November 2023; date of current version 23 February 2024. This work was supported by the European Commission under European Horizon 2020 Programme, under Grant 951911–AI4Media. This research is part of the activities of the Joint Laboratory UNIFI-Meccanica42. *(Corresponding author: Luca Cultrera.)*

Luca Cultrera, Lorenzo Seidenari, Pietro Pala, and Alberto Del Bimbo are with the University of Florence, 50121 Firenze, Italy (e-mail: luca.cultrera@unifi.it; lorenzo.seidenari@unifi.it; pietro.pala@unifi.it; alberto.delbimbo@unifi.it).

Federico Becattini is with the University of Siena, 53100 Siena, Italy (e-mail: federico.becattini@unisi.it).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TIV.2023.3336063.

Digital Object Identifier 10.1109/TIV.2023.3336063

the vehicle recovers from anomalous states. In this article, we build on these ideas without collecting additional data. We propose an IL agent that propagates the state of the vehicle through the model and uses it as the core of a multi-task architecture. On the one hand, this allows us to explicitly train the model to avoid issues such as the inertia problem. On the other hand, this allows us to perform data augmentation on all the observed data, reducing the distribution shift between training samples and what the agent may see at driving time.

Our IL agent is designed as a hierarchical transformer model with state token propagation. The vehicle's state is encoded in a special token of a vision transformer [14] and is enriched with new information at each stage of the architecture. At first, we predict whether the vehicle must stop or go, directly tackling *inertia*. This information is passed to the next stage which predicts the driving commands (namely steer, throttle, and brake). Finally, the model leverages a differentiable Command Coherency Module (CCM), encouraging the model to correctly bring the vehicle to the desired future state by generating non-conflicting controls. Such command is used only at training time and acts as a regularizer. Since our architecture is based on a transformer encoder [15], it heavily relies on attention. We leverage such attention to gain insights about what the model is focusing on to make its decisions (e.g., the vehicle's state or visual patterns), following the recent trend of designing explainable driving models [16], [17], [18].

Interestingly, the ability to explain the model's decisions provides us with a better understanding of the inertia problem. Inertia makes an IL model halt and stay still whenever the speed of the vehicle is close to zero. However, it is hard to discriminate this phenomenon from other kinds of failures that make the vehicle stop indefinitely. For instance, if part of the environment is mistakenly interpreted as a crossing pedestrian or a red traffic light, the vehicle will wait indefinitely for the state of the surrounding environment to change. Whenever this happens, a different solution must be sought in order to enforce the visual backbone of the model, rather than its causal inference capabilities. By combining the model's attention with a retrieval-based explainability method, we are able to highlight these differences and isolate instances of inertia from backbone failures.

The main contributions of our article are the following:

- We propose a state-aware conditional imitation learning model for autonomous driving. The model is multi-stage and exploits state propagation through different transformer layers breaking down the generation of driving commands into coarse to fine tasks.
- We specifically address issues in state-aware imitation learning such as the inertia problem and the offline/online performance gap. Inertia is drastically limited by state token propagation and multi-stage learning, whereas the correlation between online success rate and offline metrics is enforced via data augmentation on the vehicle's state.
- We propose a combination of the transformer's self attention with an ex-post semantic explainability method that we use for inspecting model failures. This points out interesting "hallucinations" of the visual backbone that cause behaviors mistakenly confused with inertia.

II. Related Works

Imitation Learning (IL) is based on the idea that, to learn a complex task, a model can observe the demonstrations of an expert performing it [19], [20]. This paradigm has been successfully applied to autonomous driving. One of the first approaches based on IL predicted steering commands for lane following and obstacle-avoiding tasks [21]. The task soon evolved into the so-called Conditional Imitation Learning (CIL), in which predictions are conditioned on high-level commands such as *turn* or *go straight*. Several works followed this approach [4], [17], [22], [23], [24], also combining it with reinforcement learning [25], [26], [27].

To obtain better driving capabilities, several sensors and additional synthetic data are often used [4], [28], [29], [30], [31]. Large use of environmental information is done by prior work in the form of semantic segmentations [26], [32], [33], [34], top-view maps [30], or both [27], [35], [36], [37], [38]. Similarly, other methods leverage depth information [23], [38], LiDAR data [24], [24], [32], [38] or cues such as traffic light states [26], [33], lane position [26], and intersection presence [26], [33]. Such data has been also used in the form of *affordances*, low-dimensional representations of environmental attributes [22], [26], [39]. Differently from all the aforementioned methods, we rely on a purely RGB-based approach. Whereas these methods have access to environmental data, either as inputs or as additional sources of supervision, we assume to have access only to the RGB stream and the state of the vehicle (i.e., current speed, steer, acceleration, and brake), which is a direct consequence of the driving policy. A similar assumption is done in recent works such as [5], [40], [41].

Training a state-aware imitation learning agent hides some challenges [13]. Despite its simplicity and effectiveness, it breaks the i.i.d assumption made by any statistical supervised learning framework since current decisions influence future inputs [11], [12]. The main difficulty that needs to be addressed is trying to keep the model in a state close to what has been observed at training time [13]. When this does not happen, online errors tend to accumulate over time, generating less accurate behaviors [11], [42]. The effect is to have online capabilities that do not correlate with offline error metrics measured on a validation set [10], which makes the agent difficult to train. A solution to bridge this gap is to perform data augmentation. Codevilla et al. [10] showed that collecting data from three different cameras while adding noise to the driving policy helps in recovering from unexpected scenarios. This however requires collecting hours of additional data. Image-level data augmentations such as changes in contrast, brightness and tone also have beneficial effects, especially for generalizing to similar scenarios with different conditions (e.g. weather) [5], [10]. Nonetheless, augmenting the pixel space has a limited effect on state-aware models, where predicted quantities are provided as input. Differently from prior work, we perform augmentation on the vehicle's state, injecting it into the model as a special

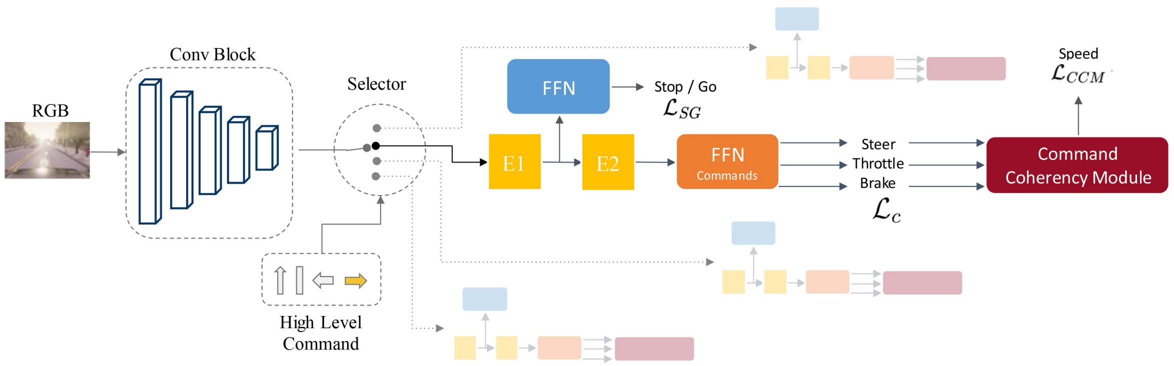

Fig. 1. Convolutional backbone extracts a feature map, which is fed to a multi-stage transformer architecture. The first stage (E1) takes the feature and a state token, which is propagated across the network. The output of E1 corresponding to the state token is decoded into a stop/go prediction with a Feed Forward Network (FFN). The second stage (E2) uses the propagated state to predict driving commands. Finally, the Command Coherency Module is used as a loss regularizer to ensure consistency between driving commands.

token of a transformer [15]. Augmenting the state leads to a better coverage of the state space during training.

The presence of the state token allows us to address another well-known issue with imitation learning in automotive: the inertia problem [5], [7]. This has been addressed in literature by predicting the current speed of the vehicle [5] or via causal imitative learning [8], also based on speed prediction. A memory-based approach for retrieving previously observed scenarios has also been exploited recently [43]. The common speed-prediction solution proposed in [5] suffers from a high collision rate, likely due to overcompensation of inertia. Instead of making the network predict its current velocity, we leverage a multi-stage architecture, where a stop/go loss based on the actual causes for stopping (presence of pedestrians, traffic lights, other vehicles) conditions the command generation. In this way, we inform the model about external elements that should be taken into account while driving. We find this solution to almost eradicate inertia entirely.

III. Overview

Imitation Learning (IL) trains an agent by observing a set of expert demonstrations to learn a policy [20]. In the simplest scenario, IL is a direct mapping from observations to actions [19]. In automotive, the expert is a driver, the policy is *"safe driving"* and the demonstrations are a set of *(frame, driving-controls)* pairs. In this article, we address Conditional Imitation Learning (CIL), a declination of imitation learning where the policy must reflect a given high-level command, such as *turn right* or *follow lane*. As in prior work (e.g. [4], [17], [22]), we divide our architecture into multiple branches, with separate heads learning command-specific policies. However, differently from prior work, we structure our model as a hierarchy of stages, each of which is dedicated to addressing different aspects of driving, as depicted in Fig. 1.

The proposed model is state-aware, in the sense that it takes as input the speed and the steer, acceleration and brake values predicted at the previous timestep. In principle, informing the model of the current state of the vehicle could ensure temporal smoothness and coherency in the driving policy (i.e., the predicted driving controls). In practice, this makes the model vulnerable to spurious correlations in the data, bringing out the *inertia problem*. To address this issue, we propose a multi-stage transformer model with state token propagation. We feed the vehicle state to the model as a special token of a vision transformer (ViT) [14]. Operatively speaking, the state token fulfills the same role as the *CLS* token in standard ViTs. However, by enclosing vehicle measurements we can inject information into the model and let it correlate to relevant spatial features via self-attention. After each layer, the state token is enriched with spatial information and is decoded into coarse-to-fine driving commands, depending on the stage. The coarser of such commands is a decision on whether the vehicle should stop or go, thus explicitly addressing inertia. Injecting the state token into the model has the additional benefit of enabling data augmentation on the state values itself, addressing what is arguably the biggest limitation of imitation learning, i.e., the inability to perform well in previously unseen states [9] that is also responsible for the gap of accuracy between offline and online driving. We also introduce a regularizer that ensures coherency in the generated driving commands. This is different from similar solutions adopted in prior works, where speed is predicted to reduce inertia [5], but here we use it to reduce online-offline evaluation gap.

IV. Model

A. State Token Propagation

Our model exploits a multi-stage transformer encoder architecture. The hierarchy of layers reflects a coarse-to-fine learning where each stage generates a different output. The rationale is that the $i-th$ stage can inform stage $i+1$ by taking the output of the encoder corresponding to the state token and propagating it as the new state token. To enrich the token with increasingly complex semantics, at each stage we decode it into a different output with a Feed Forward Network (FFN), specific for separate tasks.

We define our multi-task hierarchy as follows. The first stage predicts whether the agent should halt the car or keep it going. This is specifically thought to address the inertia problem. This stage does not produce any driving control and is expected to focus on traffic lights and other agents. The second stage generates the actual driving commands: throttle, brake, and steer. This second stage of the model should instead learn and understand road topology and ego-motion patterns. Thanks to the propagated state token, the generation of the driving commands is conditioned on the stop/go decision of the previous stage. The third and final stage is the command coherency module that acts as a regularizer, thus we use it only at training time. The initial state token is the embedding of steer, throttle, brake and speed at time $t-1$.

To cope with the non-uniform distribution of vehicle states in the train set (see Section VII-A), we introduce a data augmentation strategy based on noise injection to perturb the state token. We inject a zero mean Gaussian noise with $\sigma = 0.1$ for driving controls, since they are all in [0,1]. For the speed, that takes values in [0,10], we use $\sigma = 1$ instead.

B. Pixel-State Attention

Every stage of the model performs token-to-token attention, thanks to the transformer's self-attention. The advantages are twofold: on the one hand, prior work has shown that explicitly modeling attention improves driving capabilities [17], [40]; on the other hand, it provides a built-in interpretability mechanism that can be used to visually explain decisions.

In our model, the attention involves not only visual patches as in [17], [40] but also the state of the vehicle. First, the output of the convolutional backbone, i.e. a feature map f of size $H_f \times W_f \times C$, is flattened into $N = H_f \cdot W_f$ separate C-dimensional tokens, corresponding to 1×1 spatial patches in the feature map. Patches are then linearly projected into a D-dimensional space to adapt them to the input size of the transformer. The four scalar quantities that compose the state of the vehicle (*speed*, *steer*, *acceleration* and *brake*) are lifted to a dimension of $D/4$ and concatenated into the D-dimensional state token, which we refer to as x_{state}. As in [14], a learnable positional embedding E_{pos} is added to all the $N+1$ tokens. To summarize, the set of $N+1$ tokens fed to the encoder is composed as follows:

$$z = [x_{state}; f^1 P_v; \ldots; f^N P_v] + E_{pos} \tag{1}$$

where $P_v \in \mathbb{R}^{C \times D}$ is the feature projection matrix, $E_{pos} \in \mathbb{R}^{(N+1) \times D}$ and $x_{state} \in \mathbb{R}^{1 \times D}$.

The self-attention carried out in every layer of the transformer is thus a pixel-state attention, where every pixel of the feature map can attend to each other plus the state token. This allows us to inspect at each stage which information is privileged by the model: when the state token carries relevant information from the previous stage (e.g., if the vehicle must stop), the model will give it high importance; vice-versa, if the image carries meaningful cues (e.g., an intersection) the model will focus on the interested pixels.

C. Command Coherency Module

A possible cause for low correlation between off-line error and on-line driving performance [10] can be found in throttle, brake and steer being predicted independently. What is missing is the optimization of a common goal that brings the vehicle from one initial state to a desired one, considering all three quantities. Furthermore, individual biases may interfere with the quality of the overall policy.

To generate the appropriate driving behavior, the predicted commands must be compatible with each other. To this end, we introduce the Command Coherency Module (CCM). The CCM takes as input steer, throttle, brake and speed at time t and predicts the future speed at time $t+1$. We first train the command coherency module on training measurements to learn how such quantities affect the speed of the vehicle. Once the module is trained, we freeze it and use it as a regularizer while training the driving agent. To implement the CCM, we use a lightweight multi-layer perceptron with three layers and ELU activations.

Our CCM shares some traits with the speed prediction module of [5]. Here, the authors feed a frame-based estimate of the speed to the model. Instead of feeding the predicted speed as an additional input, we optimize it to regularize the outputs and conciliate the driving commands.

D. Architecture and Training

The proposed model is composed of a shared convolutional backbone plus four parallel branches, one for each high-level command. The shared backbone consists of 5 convolutional layers with ELU activations. The first three layers have respectively 24, 36, and 48 5×5 kernels with stride 2, followed by two other layers with 64 3×3 filters with stride 1. Input images are resized to a 200×88 px, yielding a $4 \times 18 \times 64$ feature map. After flattening we obtain $N = 72$ visual tokens. Each branch is a multi-stage transformer encoder with input size $D = 64$. We use 3 heads with a depth of 4 for each encoder stage.

The first stage of the transformer takes the state token x_{state} along with the N visual tokens. The stage outputs $N+1$ transformed tokens, among which the enriched state token is used to predict whether the vehicle should stop or go. To optimize the stop/go prediction we use an L1 loss:

$$\mathcal{L}_{SG} = \frac{|S_{TL} - \bar{S}_{TL}| + |S_P - \bar{S}_P| + |S_o - \bar{S}_o|}{3} \tag{2}$$

where S_{TL}, S_P, S_o represent intention signals in $[0,1]$ [5], respectively for traffic light stop, pedestrian stop, and stop due to other vehicles.

The second stage is in charge of generating driving commands. Similarly to the first stage, the propagated state token taken from the output of the stage is fed to a feed-forward regressor to predict steer, throttle and brake. We use an L1 loss for driving command prediction:

$$\mathcal{L}_c = |\alpha(s - \bar{s})| + |\beta(t - \bar{t})| + |\gamma(b - \bar{b})| \tag{3}$$

where $s \in [0,1]$, $t \in [0,1]$ and $b \in [0,1]$ are respectively the predicted steer, throttle and brake values, $\bar{s}$, $\bar{t}$ and $\bar{b}$ the corresponding ground truth values and α, β, and γ are weights with values $0.5, 0.45$, and 0.05, as in [23]. For the Command Coherency Module, we also use an L1 loss. The CCM loss $\mathcal{L}_{CCM}$, and the stop/go loss, denoted as $\mathcal{L}_{SG}$, contribute to the total loss according to: $\mathcal{L}_{Total} = \lambda\mathcal{L}_c + \kappa\mathcal{L}_{CCM} + \tau\mathcal{L}_{SG}$ where $\lambda = 0.8$, $\kappa = 0.1$ and $\tau = 0.1$. We train our model end to end with the Adam optimizer for 100 epochs with a batch size of 64 and a learning rate of 0.0001.

V. Model Explainability

The self-attention of the transformer stages in our model allows us to inspect the behavior of the model, thus providing explanations for the predictions. We refer to this as *built-in explainability*. Since we have dedicated each stage of the model to different tasks, we can leverage such information to gain insights about what is important for different aspects of the learned policy. We combine the built-in explainability with *ex-post explainability*, i.e. an approach specifically designed to provide an additional interpretation of the model's behavior at inference time.

A. Built-in Explainability

In both stages of the transformer model, we can obtain visual explanations in the form of attention maps. The maps are obtained by considering the attention between the state token and the image patches. The first stage provides information on what the model looks at for stop/go prediction, whereas the second identifies relevant image regions for a correct navigation.

B. Ex-Post Semantic Explainability

Built-in explainability only explains which regions are taken into account. However, it does not provide information about how these regions are interpreted by the model. We propose an ex-post semantic explainability that combines visual attention with k-NN search of image features.

We gather offline a set of m feature maps from the training set and collect the D-dimensional descriptors of each spatial location. In this way, we obtain a total of $M = m * N$ feature vectors, N being the number of image spatial patches. We denote the i-th feature in the set as y_i. At inference time, we extract the feature map f of the input image and, for any spatial location of interest (e.g., the most attended ones by built-in attention), we perform a k-NN search with FAISS [44] using the $L2$ distance:

$$L = k - \underset{i=1:M}{argmin} \|f_p - y_i\|_2 \tag{4}$$

where f_p is the p-th feature vector of the input image ($p \in \{1, \ldots, N\}$).

For each k-NN we reproject the feature back onto the original image and take the semantic segmentation of the corresponding region[1]. This allows us to inspect what the model is hallucinating by finding the dominant semantic category in the neighbors and allows us to interpret failures.

VI. Experimental Results

A. Dataset

For training and evaluating our model, we use the *Corl2017* [45] and NoCrash [5] datasets, both based on the Carla simulator [45]. The *Corl2017* dataset has expert demonstrations driving across the same town with a set of different weather conditions. Testing is performed by driving in different conditions: same town and weather as training; same town and new weather; new town; new town new weather. Testing also includes 4 tasks: go straight, one turn, navigation, navigation dynamic. The navigation tasks require driving from two distant waypoints and the dynamic scenario includes other vehicles and pedestrians. *NoCrash* has been designed to evaluate advanced driving skills such as stopping at traffic lights, avoiding collisions and driving in dense traffic environments. The evaluation involves 25 episodes on three navigation tasks, spanning from an empty town scenario to a dense traffic one. *Corl2017* has 657.601 frames and *NoCrash* instead 1.279.738 frames, divided into frontal and two lateral cameras ($-30°, +30°$). For both datasets, the agent must comply with a given high-level command among *go straight*, *turn right*, *turn left* and *follow lane*. As in [5] we train on a subsample of 10% of the data, comprising 10 hours out of a total 100 hours of driving. Both datasets provide meta-data including the current state of the autonomous vehicle and environmental information such as driving commands, high-level commands and position.

[1]Ground truth segmentations are available in the *NoCrash* dataset [5]

TABLE I
Success Rate on *Corl2017*

Model	Training Conditions	New Weather	New Town	New Town and Weather	Overall
MP§ [45]	84.2	94.5	51.2	47.7	69.4
MT$^{\S\blacklozenge}$ [32]	86.7	89.0	76.5	79.5	83.0
CAL† [22]	93.0	92.0	77.2	74.5	84.2
EF$^{\blacklozenge}$ [23]	94.7	92.0	88.5	91.0	91.5
CEF$^{\blacklozenge}$ [24]	94.5	87.2	87.2	91.0	90.0
MTL$^{\S\blacklozenge}$ [46]	99.7	98.2	95.2	96.2	97.3
CILRS* [5]	93.7	96.0	78.7	92.5	90.2
LBC* [30]	100	99.0	99.7	100	99.6
RL [45]	86.0	26.5	22.7	24.5	40.0
CIL [4]	88.2	88.5	58.5	53.5	72.2
EF-RGB [23]	90.5	75.5	67.7	65.0	74.7
CIRL [25]	92.5	90.0	66.2	**80.0**	82.2
LVA [17]	93.7	96.0	67.7	77.7	83.8
TRP [40]	96.2	97.0	72.5	73.5	84.8
Ours	**96.2**	**97.2**	**78.0**	78.0	**87.4**

Superscripts identify additional data sources: $\blacklozenge$ (depth), § (semantic segmentation), † (temporal modeling), $\star$ (Different training data).

B. Results

We report in Table I the results on the *Corl2017* dataset. For a fair analysis, we compare our method directly against other RGB-based methods. We also report methods that leverage additional sources of supervision such as depth and semantic segmentation or additional data to train the model. The results show that our method obtains better or on-par results when compared to other RGB-based models. Per-task success rates are in the supplementary material.

Compared to *Corl2017*, where traffic light violations and collisions are not considered, the *NoCrash* benchmark is extremely more challenging since environmental cues must be taken into account. We report results in Table II. Our approach outperforms RGB methods, with the only exception of CILRS [5], which performs slightly better in some empty scenarios. In the more challenging scenarios with regular and dense traffic, our approach performs better than the competitors, highlighting the

TABLE II
SUCCESS RATE ON *NOCRASH*

	Training conditions			New Weather		
Model	Empty	Regular	Dense	Empty	Regular	Dense
LBC [30]	89	87	75	60	60	54
FASNet [47]	96	90	44	98	80	38
MoDE [48]	98	93	45	98	84	46
WOR [35]	98	100	96	90	90	84
CADRE [49]	95	92	82	94	86	76
GRIAD [33]	98	98	93	83	86	82
CIL [4]	79	60	21	83	55	13
CAL [22]	81	73	42	85	68	33
MT [32]	84	54	13	58	40	7
HDRM [7]	80	70	26	92	76	44
CILRS [5]	97	83	42	**96**	77	47
Ours	**98**	**90**	**50**	90	**88**	**52**

	New Town			New Weather & New Town		
Model	Empty	Regular	Dense	Empty	Regular	Dense
LBC [30]	86	79	53	36	36	12
FASNet [47]	95	77	37	92	66	32
MoDE [48]	93	80	37	94	68	34
WOR [35]	94	89	74	78	82	66
CADRE [49]	92	78	61	78	72	52
GRIAD [33]	94	93	77	68	63	52
CIL [4]	48	27	10	24	13	2
CAL [22]	36	26	9	25	14	10
MT [32]	41	22	7	57	32	14
HDRM [7]	27	16	6	8	4	2
CILRS [5]	66	49	23	**90**	56	24
Ours	**72**	**54**	**32**	84	**58**	**28**

Methods above the line have access to privileged information not used by other RGB-based methods: top-view maps (LBC, WOR), semantic segmentations (LBC, WOR) and additional data (FASNet, MoDE, CADRE, GRIAD).

TABLE III
TRAFFIC LIGHT VIOLATIONS ON *NOCRASH* BENCHMARK (*EMPTY*)

	Traffic Light Violations	
Task	Train Conditions	New Town & Weather
CIL [4]	83%	82%
CILRS [5]	27%	64%
Single Stage	38%	75%
Ours	**14%**	**52%**

capacity of the model to interpret patterns relative to traffic lights and other agents.

In Table III we show the percentage of traffic light violations committed by our model. These results are computed on the task *Empty* both for *Training Conditions* and for *New Weather & New Town*. As a baseline, we also report the results for a Single Stage model, i.e. a simplified version of our approach without the first stage. This model is state-aware as the full model, but does not exploit the stop/go loss which we designed to prevent inertia. Interestingly, our model outperforms the single stage baseline by a large margin, showing the usefulness of the stop/go loss to correctly focus on traffic lights. At the same time, we significantly lower the traffic light violations compared to CILRS, despite it obtained a higher success rate in the empty tasks for *New Weather & New Town* (see Table II). We attribute this difference to two factors: (i) CILRS' strong ResNet vision backbone yields better generalization across weather conditions; (ii) higher capacity of our model to focus on traffic lights thanks to attention and stop/go loss.

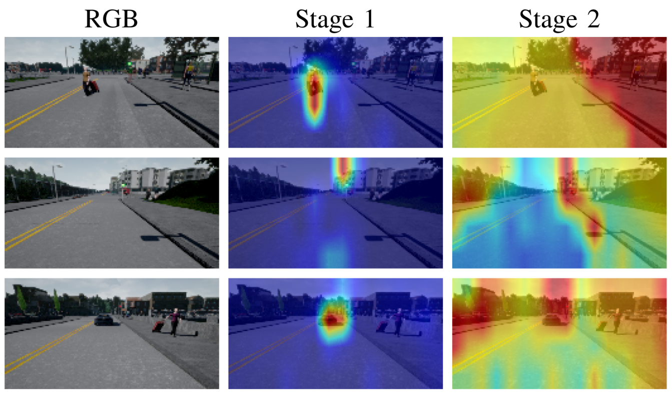

Fig. 2. Each row shows visual attention for the two stages of the model w.r.t an input image. The two stages reflect important cues for the stop/go and command generation losses respectively.

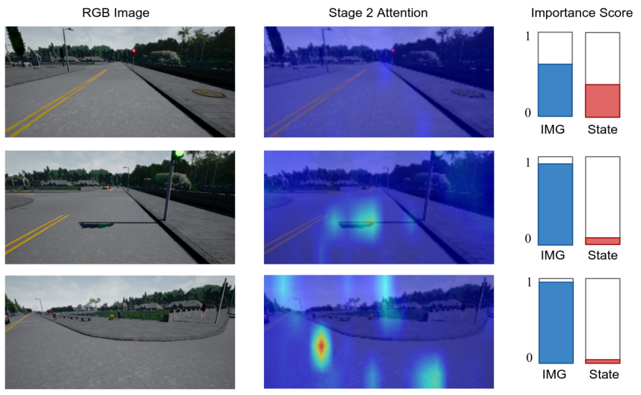

Fig. 3. Importance of state token vs image tokens. The presence of a red traffic light is detected by the first transformer stage and encoded in the state token propagated to the second stage. In this case, the second transformed stage assigns a high attention value (red bar) to the state token. When restarting at the green light and turning, the image tokens (blue bar) gain importance.

Attention plays an important role in identifying relevant cues. Since we employ transformer encoders in every stage of the model, we can visually inspect self-attention for every stage. We create heatmaps by reprojecting on the image the attention value relative to the state token against every visual token (Fig. 2). The heatmaps for the two stages reflect the tasks that are addressed at the corresponding levels: stop/go decision and driving command generation. The first stage focuses on small scene details such as traffic lights or pedestrians (additional qualitative examples for the first stage of our model are shown in Fig. 9), while the second stage attention is scattered and attends regions that are important for correct navigation such as intersections and roadsides.

VII. STATE TOKEN AND INERTIA PROBLEM

We inspect the relative importance of the state token and the image image content. The state token emitted by the first stage is used to predict a stop/go decision with a dedicated loss. This makes the token carry useful information to the second stage, which is in charge of generating the actual driving commands. In presence of a halt cue (e.g. red traffic light) encoded in the state token propagated to the second stage, the attention scheme of the

TABLE IV
ABLATION STUDY SWITCHING OFF MODEL COMPONENTS ON *NOCRASH*

	Training conditions						New Weather					
Task	Single Stage	w/o NI	w/o ST	w/o S/G loss	w/o CCM	Ours	Single Stage	w/o NI	w/o ST	w/o S/G loss	w/o CCM	Ours
Empty	73	94	82	80	84	**98**	72	90	62	84	80	**90**
Regular	60	88	50	56	60	**90**	56	72	46	46	52	**88**
Dense	30	48	20	28	28	**50**	32	36	16	24	28	**52**
	New Town						New weather and new Town					
Task	Single Stage	w/o NI	w/o ST	w/o S/G loss	w/o CCM	Ours	Single Stage	w/o NI	w/o ST	w/o S/G loss	w/o CCM	Ours
Empty	60	70	60	58	64	**72**	62	72	66	64	70	**84**
Regular	36	48	24	28	32	**54**	32	40	16	24	36	**58**
Dense	16	28	12	16	16	**32**	12	16	8	12	12	**28**

We remove: noise injection function (w/o NI), command coherency module (w/o CCM), state token (w/o ST) and stop/go loss (w/o S/G loss). We also show results for the single stage baseline.

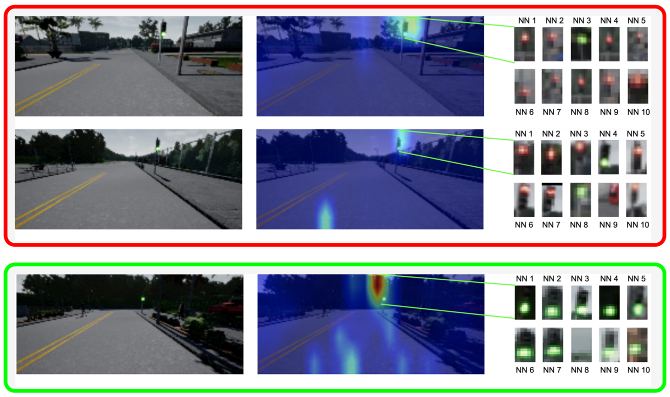

Fig. 4. Top 10 nighbors for the highest scoring attention after a traffic light turns green. We show examples of both successful crossing of the traffic light (framed in green) and failed due to red light "hallucination" (framed in red).

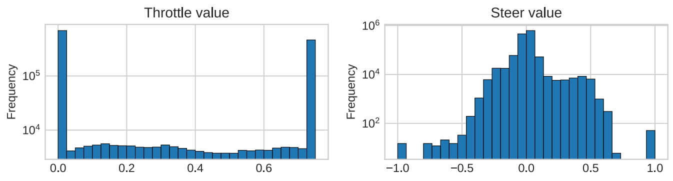

Fig. 5. *NoCrash* (left) Throttle distribution; (right) Steer distribution.

second stage focuses on the state token rather than on the image patches. When the state token indicates that the vehicle can advance, the attention focuses instead on the image patches to generate appropriate driving commands. Fig. 3 shows examples of stage 2 attention, with values of the state token and of the image, accumulated for each visual token.

The stop/go loss has a great impact on driving performance. In Table IV we show the effect of removing such loss on the *NoCrash* benchmark. In densely trafficked environments, the success rate is almost halved when removing the loss. Similar results are obtained with the single stage baseline. We also test a model trained using a random vector as state token (w/o ST), yet keeping the stop/go loss: success rate heavily drops, especially in crowded environments. By feeding the state of the vehicle, the agent becomes aware of its speed and momentum, e.g. indicating whether and how a turn is taking place. This is hard to deduct from a single image.

TABLE V
FAILURE RATE DUE TO INERTIA PROBLEM IN TOWN01 - NEW WEATHER OF THE *NOCRASH* BENCHMARK

	Train conditions		New weather	
Task	Single Stage	Ours	Single Stage	Ours
Empty	17%	**1%**	40%	**8%**
Regular	9%	**1%**	20%	**2%**
Dense	16%	**6%**	22%	**4%**

Furthermore, the use of the state token and the stop/go loss, have a direct effect on addressing the inertia problem since the first stage is explicitly trained to predict movement. Table V shows the failure rate due to inertia. As in [5] we identify the inertia problem when an agent is still for 8 seconds before time out. Most failures of the single-stage baseline can be traced back to inertia and these are almost completely eliminated with the multi-stage model. Surprisingly, the NewWeather-Empty configuration in the NoCrash Benchmark exhibits the highest failure rate, attributed to inertia (as indicated in Table V). In this context, when the vehicle comes to a halt, it becomes trapped in a stationary state due to inertia. Notably, in an empty scenario, the sole discernible visual cue is the traffic light. Conversely, in Regular or Dense scenarios, the dynamic nature of the environment allows the autonomous vehicle to break free from its stationary state by observing other vehicle behaviors and the dynamic surroundings, prompting a reevaluation of its decisions. In simple terms, the distance from a vehicle ahead or the dynamic behaviors of other agents can act as a trigger to escape from stall states caused by the inertia problem.

A more general analysis of the causes of failure is also provided in Table VI. The multi-stage model considerably reduces collisions with pedestrians and vehicles, compared to the single-stage baseline. Interestingly failures due to time out (which include inertia) are almost eliminated. Tables V and VI indicate that, despite addressing in a very effective way the inertia problem, the model still suffers from a few inertia failures. We exploit the Ex-post Semantic Explainability approach presented in Section V to inspect 50 episodes of the *NoCrash* benchmark where the inertia problem still occurs at traffic lights. In 56% of the cases where the vehicle is stuck at a green light, the k most similar features to the attended one contain a red traffic light, in 18% a pedestrian crossing, and in 3% a vehicle (Table VII).

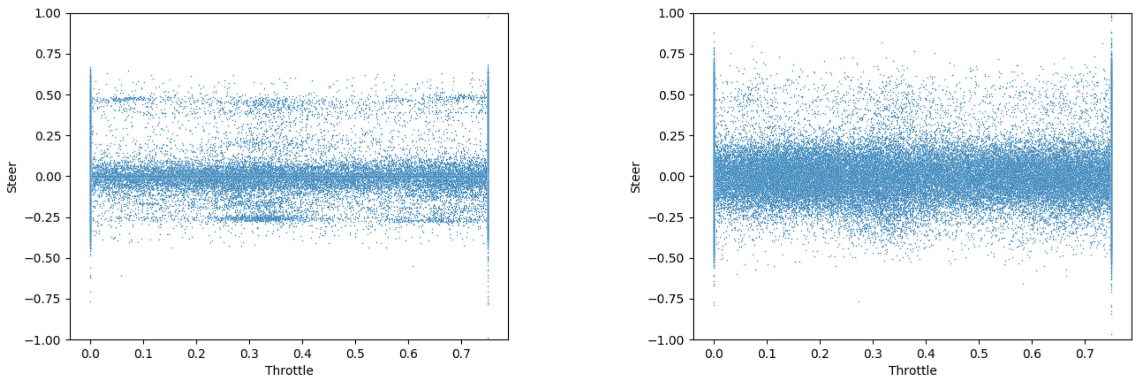

Fig. 6. Distribution of Steer-Throttle without (left) and with (right) Noise Injection on *NoCrash*. We have a better coverage of the space with Noise Injection.

TABLE VI
SUCCESS RATE AND NUMBER OF FAILED EPISODES TERMINATION CAUSE ON NOCRASH

	Train Conditions									New Weather								
	Empty			Regular			Dense			Empty			Regular			Dense		
	CILRS [5]	Single Stage	Ours	CILRS [5]	Single Stage	Ours	CILRS [5]	Single Stage	Ours	CILRS [5]	Single Stage	Ours	CILRS [5]	Single Stage	Ours	CILRS [5]	Single Stage	Ours
Success ↑	97	73	98	83	60	90	42	30	50	98	72	90	77	56	88	47	32	52
Ped. Col. ↓	**0**	**0**	**0**	4	7	**2**	24	**16**	18	**0**	**0**	**0**	**2**	4	**2**	**12**	16	20
Veh. Col. ↓	**0**	**0**	**0**	8	9	**2**	18	33	**11**	**0**	**0**	**0**	17	18	**3**	**26**	34	30
Other Col. ↓	2	10	**1**	**5**	12	**5**	13	**5**	15	**0**	8	6	**3**	12	6	11	7	**6**
Time out ↓	**1**	17	**1**	**0**	12	1	**3**	16	6	**2**	20	4	**1**	10	**1**	4	11	**2**

We compare with the baseline and CILRS in all tasks and weather conditions in TOWN01. For success rate we underline the method with best performances (higher is better ↑), for failure cases we highlight (bold) the method with the lowest number of occurrences for that type of failure (lower is better ↓).

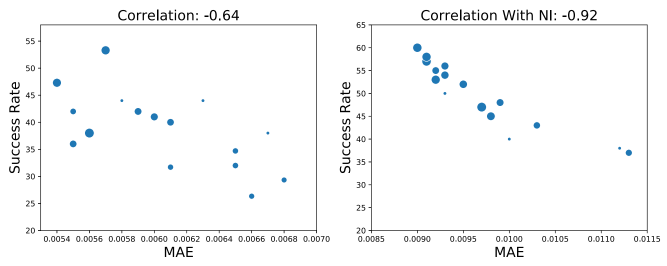

Fig. 7. Pearson correlation between online success rate and offline MAE obtained by training the model multiple times without (left) and with (right) data augmentation on the state token. Dot size corresponds to training epoch.

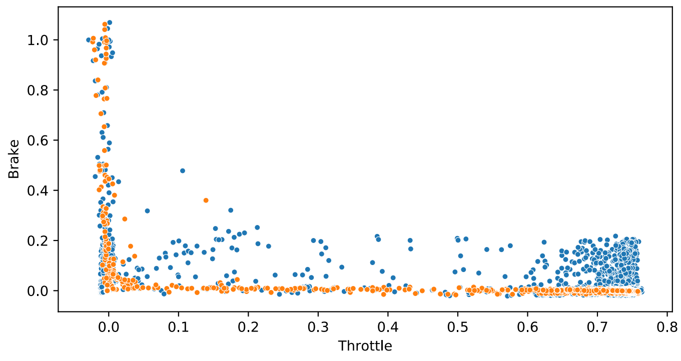

Fig. 8. Distribution of Brake-Throttle values with the Command Coherency Module disabled (blue) and enabled (orange).

In Fig. 4 we show the top 10 nearest samples of the image region with the highest attention value (first transformer stage). The first two rows show failure cases: the model correctly focuses on the traffic light but although it is green, the model maps it in a region of the latent space densely populated by red traffic lights. We also show a sample of correct driving, where the vehicle accelerates as soon as the light turns green: retrieved images all depict green lights. This suggests that what may appear as inertia might instead be confused with a failure of the backbone that mistakenly "hallucinates" halt cues.

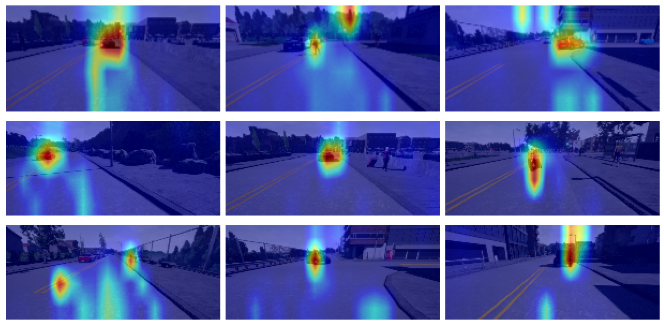
Fig. 9. Examples of Stage 1 attention. This layer is dedicated to understanding whether the vehicle should halt its motion or not and therefore focuses on relevant cues in the scene such as vehicles, pedestrians and traffic lights.

TABLE VII
% OF DETECTED ENTITIES IN FEATURES WHEN THE VEHICLE IS STOPPED AT GREEN TRAFFIC LIGHT ON NOCRASH

Cause of failure	Percentage of detection
Red Traffic light	56%
Pedestrian crossing	18%
Vehicle obstruction	3%
Tot	**77%**

A. *Online/Offline Evaluation and Noise Injection*

To address the offline/online shift, exhaustive coverage at training time of possible input configurations (observed environment + internal state) could be a solution, yet it is difficult to achieve. For instance, the *NoCrash* dataset is unbalanced and throttle and steer values are extremely biased (Fig. 5). This limits the possibility of effectively inputting the vehicle state into the model at driving time. Our data augmentation strategy that injects noise on the state token (Section IV-A) is intended to address this limitation. We introduce a zero mean Gaussian noise with $\sigma = 0.1$ on the driving controls (which are in [0,1]) and with $\sigma = 1$ for speed. This has the effect of letting the model see at training time different combinations of state values. In Fig. 6 we show the joint distribution of steer and throttle values with and without noise injection. Two modes for throttle can be observed corresponding to the over-represent stationary and full-throttle scenarios. At the same time, steer has a Gaussian distribution centered in zero (indicating no steer). With noise injection, we get a more uniform distribution in the low-steer interval $[-0.25, 0.25]$ across all throttle values. Also, higher steer values obtain a more uniform coverage.

In Fig. 7 we quantify the correlation between online success rate and offline validation MAE using the sample Pearson correlation coefficient, as done in [10]. We plot the results without using data augmentation via noise injection (corr: -0.64) and with (corr: -0.92). Despite not having a huge impact on the results in training conditions, as shown in Table IV, in generalization conditions noise injection brings noticeable benefits. From the plots in Fig. 7 it can be seen that without using data augmentation there are huge differences for similar MAE values (e.g., 20% success rate gap with a small difference of 0.0001 in MAE). Another component to help the agent act as the expert demonstrator is the command consistency module (see IV-C). It acts as a regularizer, encouraging the model to generate driving commands that are not in conflict with each other, and thus preventing unwanted behaviors at driving time. The necessity of CCM also stems from the fact that maneuvers (e.g. a right turn) could be performed in different ways (e.g. slow and narrow or fast and wide turn). Results in Table IV confirm the usefulness of CCM.

B. On the Command Coherency Module

In the proposed architecture, the CCM module is responsible for the generation of non-conflicting throttle and brake commands at training time. The impact of this module on the coherency of throttle and brake pairs and ultimately on the accuracy of driving at test time is demonstrated in the experiments reported in Table IV that highlight a severe performance drop when training is conducted with the CCM disabled. To delve deeper into the effect of the CCM module, we log pairs of throttle and brake command values during a driving session executed twice, with CCM respectively enabled and disabled. For this experiment, we use an episode of the *NoCrash* benchmark consisting of about 3000 frames. Values of pairs (throttle, brake) are shown in the scatter plots of Fig. 8. When the CCM is disabled, it can be noticed that a relevant portion of the outputs is characterized by both throttle and brake values greater than zero, meaning that the vehicle is both trying to accelerate and decelerate at the same time. By using the CCM, almost all these non-coherent configurations disappear and only one of the two commands at a time can take values significantly greater than zero.

VIII. Conclusion and Future Works

We addressed two major issues in training a state-aware model with imitation learning. First, the inertia problem has been dealt with using a multi-stage architecture with state token propagation, where the first stage learns to inform the next one about stop/go decisions. We report extremely low rates of inertia. Second, the offline/online gap has been bridged by performing data augmentation on the state token, significantly increasing the correlation between success rate and validation error. In addition, we also exploited built-in visual attention with a retrieval-based ex-post explainability to characterize failures. We found that what may appear as inertia might indeed be caused by a completely different problem, such as backbone hallucinations. In future works, we intend to exploit the hierarchical structure of the model to create an inspection chain to debug the model: whenever the explainability allows us to discover an issue, we can add a new level in the hierarchy to enrich the state token with new information and condition the generation of the driving commands. An interesting approach in this direction would be to study a model with different modules to be deployed in parallel rather than stacking them hierarchically. Furthermore, the proposed model could be easily improved to include additional sources of data such as depth and segmentation, e.g. concatenating them to the input image.

References

[1] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, “Full-resolution residual networks for semantic segmentation in street scenes,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4151–4160.

[2] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, “A review of motion planning for highway autonomous driving,” *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1826–1848, May 2020.

[3] F. Marchetti, F. Becattini, L. Seidenari, and A. D. Bimbo, “Multiple trajectory prediction of moving agents with memory augmented networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 6688–6702, Jun. 2023.

[4] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, “End-to-end driving via conditional imitation learning,” in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1–9.

[5] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, “Exploring the limitations of behavior cloning for autonomous driving,” in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9329–9338.

[6] P. De Haan, D. Jayaraman, and S. Levine, “Causal confusion in imitation learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, Art. no. 1049.

[7] A. Greco, L. Rundo, A. Saggese, M. Vento, and A. Vicinanza, “Imitation learning for autonomous vehicle driving: How does the representation matter?,” in *Proc. Int. Conf. Image Anal. Process.*, 2022, pp. 15–26.

[8] M. R. Samsami, M. Bahari, S. Salehkaleybar, and A. Alahi, “Causal imitative model for autonomous driving,” 2021, *arXiv:2112.03908*.

[9] L. Le Mero, D. Yi, M. Dianati, and A. Mouzakitis, “A survey on imitation learning techniques for end-to-end autonomous vehicles,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 9, pp. 14128–14147, Sep. 2022.

[10] F. Codevilla, A. M. Lopez, V. Koltun, and A. Dosovitskiy, “On offline evaluation of vision-based driving models,” in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 236–251.

[11] S. Ross and D. Bagnell, “Efficient reductions for imitation learning,” in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 661–668.

[12] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 627–635.

[13] Y. Schroecker and C. L. Isbell, “State aware imitation learning,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2915–2924.

[14] A. Dosovitskiy et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” 2020, *arXiv:2010.11929*.

[15] A. Vaswani et al., “Attention is all you need,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.

[16] D. Omeiza, H. Webb, M. Jirotka, and L. Kunze, "Explanations in autonomous driving: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 10142–10162, Aug. 2022.

[17] L. Cultrera, L. Seidenari, F. Becattini, P. Pala, and A. Del Bimbo, "Explaining autonomous driving by learning end-to-end visual attention," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 340–341.

[18] É. Zablocki, H. Ben-Younes, P. Pérez, and M. Cord, "Explainability of vision-based autonomous driving systems: Review and challenges," *Int. J. Comput. Vis.*, vol. 130, no. 10, pp. 2425–2452, 2022.

[19] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[20] A. Attia and S. Dayan, "Global overview of imitation learning," 2018, *arXiv 1801.06503v1*.

[21] M. Bojarski et al., "End to end learning for self-driving cars," 2016, *arXiv:1604.07316*.

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

[23] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. López, "Multimodal end-to-end autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 537–547, Jan. 2022.

[24] M. Haris and A. Glowacz, "Navigating an automated driving vehicle via the early fusion of multi-modality," *Sensors*, vol. 22, no. 4, 2022, Art. no. 1425.

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

[26] 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.*, 2020, pp. 7153–7162.

[27] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. Van Gool, "End-to-end urban driving by imitating a reinforcement learning coach," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15222–15232.

[28] D. Lee, S. Liu, J. Gu, M.-Y. Liu, M.-H. Yang, and J. Kautz, "Context-aware synthesis and placement of object instances," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 10414–10424.

[29] L. Berlincioni, F. Becattini, L. Seidenari, and A. Del Bimbo, "Multiple future prediction leveraging synthetic trajectories," in *Proc. IEEE 25th Int. Conf. Pattern Recognit.*, 2021, pp. 6081–6088.

[30] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Proc. Conf. Robot Learn.*, 2020, pp. 66–75.

[31] Z. Yang, Y. Zhang, J. Yu, J. Cai, and J. Luo, "End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perceptions," in *Proc. 24th Int. Conf. Pattern Recognit.*, 2018, pp. 2289–2294.

[32] Z. Li, T. Motoyoshi, K. Sasaki, T. Ogata, and S. Sugano, "Rethinking self-driving: Multi-task knowledge for better generalization and accident explanation ability," 2018, *arXiv:1809.11100*.

[33] R. Chekroun, M. Toromanoff, S. Hornauer, and F. Moutarde, "GRI: General reinforced imitation and its application to vision-based autonomous driving," *Robot.*, vol. 12, no. 5, 2023, Art. no. 127.

[34] D. Chen and P. Krähenbühl, "Learning from all vehicles," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 17222–17231.

[35] D. Chen, V. Koltun, and P. Krähenbühl, "Learning to drive from a world on rails," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15590–15599.

[36] K. Chitta, A. Prakash, and A. Geiger, "NEAT: Neural attention fields for end-to-end autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 15793–15803.

[37] S. Hu, L. Chen, P. Wu, H. Li, J. Yan, and D. Tao, "ST-P3: End-to-end vision-based autonomous driving via spatial-temporal feature learning," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 533–549.

[38] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger, "TransFuser: Imitation with transformer-based sensor fusion for autonomous driving," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.

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

[40] L. Cultrera, F. Becattini, L. Seidenari, P. Pala, and A. Del Bimbo, "Explaining autonomous driving with visual attention and end-to-end trainable region proposals," *J. Ambient Intell. Humanized Comput.*, pp. 1–13, 2023.

[41] Y. Xiao, F. Codevilla, D. P. Bustamante, and A. M. Lopez, "Scaling self-supervised end-to-end driving with multi-view attention learning," 2023, *arXiv:2302.03198*.

[42] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 1988, pp. 305–313.

[43] C.-C. Chuang, D. Yang, C. Wen, and Y. Gao, "Resolving copycat problems in visual imitation learning via residual action prediction," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 392–409.

[44] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, Jul. 2021.

[45] A. Dosovitskiy, G. Ros, F. Codevilla, and A. López, "CARLA: An open urban driving simulator," in *Proc. Conf. Robot Learn.*, 2017, pp. 1–16.

[46] K. Ishihara, A. Kanervisto, J. Miura, and V. Hautamaki, "Multi-task learning with attention for end-to-end autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2902–2911.

[47] I. Kim, H. Lee, J. Lee, E. Lee, and D. Kim, "Multi-task learning with future states for vision-based autonomous driving," in *Proc. Asian Conf. Comput. Vis.*, 2020, pp. 654–669.

[48] I. Kim, J. Lee, and D. Kim, "Learning mixture of domain-specific experts via disentangled factors for autonomous driving," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 1148–1156.

[49] Y. Zhao et al., "CADRE: A cascade deep reinforcement learning framework for vision-based autonomous urban driving," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 3481–3489.

Luca Cultrera received the Ph.D. degree in information engineering from the University of Florence, Florence, Italy, in 2023. He is currently a Postdoctoral Fellow with Media Integration and Communication Center (MICC), University of Florence.

Federico Becattini is currently a Tenure-Track Assistant Professor with the University of Siena, Siena, Italy. He has organized tutorials and workshops at ICPR2020, ICIAP2020, ACMMM2022, ICPR2022, ECCV2022. He has coauthored more than 40 papers. His research interests include computer vision and memory-based learning. He is an Associate Editor for the *International Journal of Multimedia Information Retrieval* (IJMIR).

Lorenzo Seidenari is currently an Associate Professor with the University of Florence, Florence, Italy. He is an ELLIS Scholar. He was a Visiting Scholar with the University of Michigan, Ann Arbor, MI, USA, in 2013. He gave a tutorial at ICPR 2012 on image categorization. He is author of 17 journal papers and more than 50 peer-reviewed conference papers.

Pietro Pala is currently a Full Professor of computer science and engineering with the University of Florence, Florence, Italy. He is author of more than 40 journal papers and more than 140 peer-reviewed conference papers. His research interests include 3D data processing with deep learning models for biometrics and action recognition. He is an Associate Editor for the *ACM Transactions on Multimedia Computing, Communications, and Applications*, and Springer *Multimedia Systems*.

Alberto Del Bimbo (Senior Member, IEEE) is currently an Emeritus Professor with the University of Florence, Florence, Italy. He was the Director of the University of Florence. His research focuses on multimedia vision. He was the Co-Chair of ACMMM2010, ECCV2012, ICPR2020, ACM nominated him Distinguished Scientist. He was the recipient of the SIGMM Technical Achievement Award for Outstanding Technical Contributions to Multimedia Computing, Communications and Applications.