Multi-Task Adaptive Gating Network for Trajectory Distilled Control Prediction

Shoaib Azam, *Member, IEEE* and Ville Kyrki, *Senior Member, IEEE*

***Abstract*—End-to-end autonomous driving is often categorized based on output into trajectory prediction or control prediction. Each type of approach provides benefits in different contexts, resulting in recent studies on how to combine them. However, the current proposals are based on heuristic choices that only partially capture the complexities of varying driving conditions. How to best fuse these sources of information remains an open research question. To address this, we introduce MAGNet, a Multi-Task Adaptive Gating Network for Trajectory Distilled Control Prediction. This framework employs a multi-task learning strategy to combine trajectory and direct control prediction. Our key insight is to design a gating network that learns how to optimally combine the outputs of trajectory and control predictions in each situation. Using the CARLA simulator, we evaluate MAGNet in closed-loop settings with challenging scenarios. Results show that MAGNet outperforms the state-of-the-art on two publicly available CARLA benchmarks, Town05 Long and Longest6.**

***Index Terms*—Intelligent Transportation Systems, Autonomous Agents, Imitation Learning, Gating Network, End-to-End Autonomous Driving**

I. Introduction

LEARNING effective driving policies is pivotal for the development of end-to-end autonomous driving solutions. Typically, these driving policies are distinguished based on their outputs, falling into either trajectory prediction [1]–[4] or direct control prediction categories [5]–[7]. Trajectory prediction aims to forecast the vehicle's motion in the future over a specified horizon and uses separate controllers, for instance, PID or model predictive controllers (MPC), to translate the planned trajectories to the vehicle actuators. Conversely, control-based methods optimize the control signal directly. Both trajectory and direct control prediction have merits and demerits. In particular, trajectory prediction outcomes can be integrated with other tasks, like semantics and occupancy prediction methods [8], or multi-agent interactions [9], enhancing safety and refining the planned trajectory. However, since trajectory prediction relies on controllers to convert planned trajectories into control signals, the type of controller used may constrain its performance. On the other hand, control-prediction methods often result in discontinuous and unstable behavior because they make independent predictions at different steps. However, a clear consensus on which paradigm is superior remains elusive.

The underlying research question rarely studied in the literature is how to combine the trajectory and control prediction based on observed situations. The pioneering work in this direction is Trajectory-guided control prediction (TCP) [10], which has developed a multi-task learning framework for combining both prediction methods by heuristically determining a situation-fusing parameter. However, this heuristic parameter cannot fully capture the situation-dependency of the optimal combination of trajectory and control predictions.

To fill this gap, we introduce **MAGNet** (Multi-Task Adaptive Gating Network for Trajectory Distilled Control Prediction) by designing a gating network that learns the situation-fusing parameter based on the perception of the environment. MAGNET employs a multi-task learning strategy to perform trajectory and control prediction simultaneously. MAGNet incorporates the self-attention mechanism to distill the control prediction branch with the trajectory guidance to address the limitations of direct control methods that predominantly focus on immediate low-level actions, often not fully capturing the complexities of end-to-end autonomous driving. Moreover, our method dynamically learns the situation-fusing parameter, adapting to the environmental input representation, for fusing the trajectory and control prediction outputs. By doing so, we achieve a more dynamic integration of trajectory and control predictions, enhancing the vehicle's situational awareness.

The main contributions of this paper can be summarized as follows:

1) We developed MAGNet with a novel gating network that dynamically fuses control and trajectory predictions, distinguishing it from traditional methods that typically depend on static or heuristic-based approaches for integration. This methodological advancement empowers the model to adapt its integration strategy to suit each unique driving scenario. This flexibility is anticipated to enhance the robustness and accuracy of driving policies.
2) We have integrated a self-attention mechanism into the control prediction branch of MAGNet, primarily due to its ability to enhance the model's focus on the most pertinent features derived from trajectory prediction. The use of self-attention in this context is novel because it allows MAGNet to selectively emphasize critical aspects of the input data, which is crucial for making precise

Manuscript received: October, 14, 2023; Revised December, 26, 2023; Accepted March, 13, 2024.

This paper was recommended for publication by Editor Aleksandra Faust upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the Research Council of Finland Flagship programme: Finnish Center for Artificial Intelligence FCAI. (Corresponding author: Shoaib Azam)

Shoaib Azam and Ville Kyrki are with the Department of Electrical Engineering and Automation, Aalto University, Finland (shoaib.azam, ville.kyrki)@aalto.fi

This letter has supplementary downloadable video material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier (DOI): see top of this page.

and efficient control decisions in dynamic and complex driving environments.

3) Our evaluations and ablation studies demonstrate that MAGNet's situation-based fusing parameter outperforms heuristic methods, with experimental results on CARLA benchmarks confirming its efficacy over state-of-the-art models in closed-loop settings.

II. Related Work

A. End-to-end Autonomous Driving

End-to-end autonomous driving methods, classified into trajectory and direct control prediction approaches, learn to map sensor data to actions via imitation learning (IL) or reinforcement learning (RL) [11]. RL, particularly model-free reinforcement learning, is effective in autonomous driving, adapting well to data shifts and proven successful in vehicle control [12]. Furthermore, model-based methods learn the world model using pre-recorded trajectories and compute action-value functions, which, with sensor inputs, train a policy for error-correct navigation [13]. Some studies separate perception from the RL process in driving policy learning [14]–[16].

In literature, end-to-end driving policies are often learned through imitation learning, particularly behavior cloning. This involves feature representation steps like mapping BEV semantics to waypoint prediction [2], or incorporating global and temporal reasoning [17]. Some studies also focus on a unified framework that integrates perception, prediction, and planning using intermediate representations [8], [18]. Sensor fusion techniques are increasingly used in driving policy learning, such as combining Lidar and image data with self-attention and GRU-based decoders for trajectory prediction [3], [19]. Additionally, some methods learn policies from both ego and other vehicles' perspectives using viewpoint-invariant representations [9], and also improve the decoder for trajectory learning [20].

Unlike the conventional trajectory prediction, direct control prediction is another approach for learning the driving policy [21]–[24]. Some studies have incorporated the perception network and learned controller for post-trajectory prediction for learning the control policy [5], [25], [26].

B. Multi-task learning and Knowledge Distillation

Multi-task learning trains networks on related tasks to boost performance and generalization, a technique increasingly applied in end-to-end autonomous driving systems [27]–[29]. FASNet, within a multi-task learning framework, forecasts future states and actions using deep-predictive coding and vehicle kinematics, with control signals produced from a weighted average of predicted actions [30]. Similar to our work, Trajectory-guided control prediction (TCP) follows a multi-task learning framework for trajectory and control prediction and then adopts a heuristic approach to fuse them [10]. Unlike TCP's heuristic integration of trajectory and control predictions, our method employs a learned fusion strategy via a situation-aware gating network, adjusting fusion coefficients for contextual precision. We also enhance branch interaction with a self-attention mechanism, optimizing knowledge distillation by prioritizing salient feature integration.

Knowledge distillation has been used in autonomous driving, training a privileged agent with extensive data and then using it to train a sensorimotor agent with limited data [1]. Some studies include an an alignment module as enhancement, to better transfer knowledge from teacher to student, optimizing learning through a coaching approach [31].

III. Method

A. Problem Setting

In end-to-end autonomous driving, the objective is to translate an input representation $\mathbf{x}$ into a corresponding control action $\mathbf{u}$. In this paper, we consider the input representation which encompasses sensor signal s_i, vehicle speed υ, a high-level navigation command ρ, a goal point (x, y). This goal point (x, y) provides a target location for the vehicle's navigation, integral to the driving task. The resulting control action constitute of longitudinal control signals: [*throttle* $\in [0, 1]$, *brake* $\in [0, 1]$], and the lateral control signal: [*steer* $\in [-1, 1]$].

In our research, we explore methods to contextually and adaptively merge the outputs of trajectory and control prediction in a learnable manner. For the trajectory prediction, a point-to-point navigation approach is adopted by learning a driving policy π that imitates the behavior of an expert policy π^* in a supervised manner with the loss function, $\mathcal{L}$:

$$\arg\min_{\theta} \mathbb{E}_{(\mathbf{x},\mathcal{W})\sim D}[\mathcal{L}(\mathcal{W}, \pi_\theta(\mathbf{x}))] \tag{1}$$

where $\mathcal{W}$ are the ground-truth waypoints and $\pi(\mathbf{x})$ is the learned policy for predicting the waypoint over the horizon $\mathcal{T}$. Similarly, the control branch is trained in a manner consistent with behavior cloning in imitation learning, where expert-provided control signals directly supervise the model's current control predictions, and it can be formulated as:

$$\arg\min_{\theta} \mathbb{E}_{(\mathbf{x},\mathbf{u})\sim D}[\mathcal{L}(\mathbf{u}, \pi_\theta(\mathbf{x}))] \tag{2}$$

where D corresponds to the dataset. The dataset D is collected by rolling the expert policy π^* that interacts with the simulated world. Each trajectory $\tau = (\mathbf{x_0}, \mathbf{u_0^*}, \mathbf{x_1}, \mathbf{u_1^*}, ..., \mathbf{x_T})$ comprises of state-action $(\mathbf{x}, \mathbf{u}^*)_{i=0}^{T}$ pairs, where $\mathbf{u}^*$ includes the controls signals and waypoints information, along with the goal point data.

B. Architecture

Fig.1 provides an overview of the MAGNet architecture, which consists of four main components: an encoding stage for feature extraction, trajectory prediction and control prediction branches, and a situation-based gating network for fusing the outputs of these branches. The encoding stage is further divided into two encoders. The image encoder ($\mathbf{E_{I_c}}$), built on a ResNet [32] architecture, is responsible for extracting feature embeddings ($I_{C_{emb}}$) and feature vector ($I_{C_{feat}}$) from the input RGB image. $I_{C_{feat}}$ is the feature vector from the last block of ResNet module used in image encoder ($\mathbf{E_{I_c}}$) for the feature representation. Additionally, a measurement

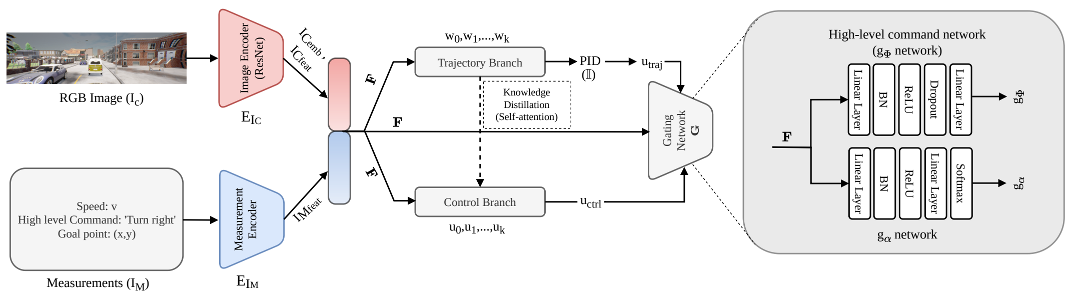


Fig. 1: **Overview of architecture.**The architecture comprises of three modules: trajectory prediction branch, control prediction branch and gating network. The encoded features are shared by all the three modules. The gating network receive both outputs from the trajectory and trajectory-distilled control prediction branch, and fuse them by learning the situation-based fusing parameter.

encoder ($\mathbf{E_{I_m}}$) is employed to generate measurement features ($I_{M_{feat}}$). The image feature embedding ($I_{C_{emb}}$) is averaged and concatenated with measurement features ($I_{M_{feat}}$) to form combined feature vector $\mathbf{F} \in \mathbb{R}^{d_c+d_m}$, where d_c and d_m are the dimensions of ($I_{C_{emb}}$) and ($I_{M_{feat}}$), respectively. The feature vector **F** is propagated to the subsequent two branches and the gating network. The following sections detail the trajectory branch, control branch, and situation-based gating network.

*1) **Trajectory Prediction Branch**:* Unlike the control prediction that directly predicts the control action, the trajectory prediction branch, as illustrated in Fig.2, predicts the planned trajectory over the horizon $\mathcal{K}$, which are then processed by low-level controllers $\mathbf{u_{traj}} = \mathbb{I}(\mathcal{W})$, where $\mathbb{I}$ corresponds to the low-level controller, $\mathcal{W}$ corresponds to the waypoints. In the proposed method, the trajectory prediction branch inputs the combined feature vector **F**, down-sampled to a feature vector of $f = 256$ by passing through a series of linear layers. For predicting the next waypoints, we have employed the auto-regressive GRU [33] model and initiated the hidden states of the GRU model with the feature vector f. The auto-regressive model, built on a GRU architecture, utilizes the current position and goal location as inputs. This design enables the network to concentrate on pertinent contextual information within its hidden states, thereby enhancing its ability to predict subsequent waypoints. Finally, a linear layer followed by GRU layers is used to predict the next waypoints $(w_0, w_1, ..., w_{\mathcal{K}})$ over horizon $\mathcal{K} = 4$. Two PID controllers, one for longitudinal and another for lateral control, process the predicted waypoints for generating control actions in the form of throttle, brake, and steer, respectively.

*2) **Control Prediction Branch**:* As illustrated in Fig.3, we designed the control prediction branch to predict the multi-step control actions in the future by distilling the information from the trajectory branch. Since the traditional control prediction methods follow the behavioral cloning approach, which relies on independent and identically distribution, it does not hold in the case of closed-loop settings. To address this limitation, we employ self-attention to design a trajectory distilled control prediction branch.

The control prediction branch comprises two branch networks: value and policy head. An initial feature vector **F**

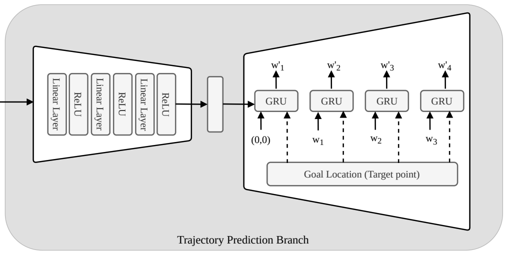


Fig. 2: **Trajectory prediction branch.** The architecture receives the encoded features **F**, down-sampled and passed to GRU based decoder for predicting the next waypoints.

undergoes processing through a series of linear layers to produce a down-sampled feature vector **x**, which is then utilized by both the value and policy heads. In the trajectory distilled control prediction, the self-attention is used initially to compute the attention matrix $A \in \mathbb{R}^{m \times n}$, as shown in Eq.3, where the **Q**,**K** and **V** matrices are derived from the measurement features ($I_{M_{feat}}$). The rationale behind employing the self-attention mechanism in our model lies in its capability to independently evaluate and integrate input features, both measurement and image data. This approach ensures contextually informed and temporally coherent feature integration, which is critical for making accurate decisions in dynamic driving environments. It is to be noted here that the self-attention employed in our trajectory distilled control prediction branch is different from the TCP. TCP uses trajectory-guided attention to focus on specific regions of the sensor input, creating an attention map that aggregates 2D image features for control prediction. However, MAGNet employs a self-attention mechanism that merges measurement features with image features, enhancing the model's capability to focus on the most relevant aspects of the input for control prediction. This approach is more dynamic and context-aware, allowing for integrating different types of input features.

$$A(\mathbf{Q},\mathbf{K},\mathbf{V}) = softmax(\frac{\mathbf{Q}\boldsymbol{K}^T}{\sqrt{d}})\mathbf{V} \tag{3}$$

This initial attention matrix is used to compute the feature embedding for the control prediction branch by taking the dot product with image feature vector ($I_{C_{feat}}$). The core logic

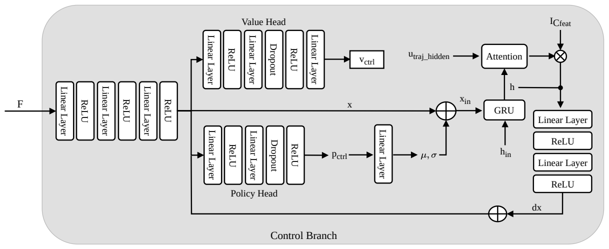


Fig. 3: **Control prediction branch.** The architecture for predicting the multi-step control prediction through trajectory branch supervision. GRU for multi-step control prediction and self-attention for knowledge distillation between trajectory and control.

unfolds within the temporal loop implemented as GRU with a prediction horizon $\mathcal{K}$. For each iteration $[0, \mathcal{K}-1]$, the model ingest a concatenated vector $\mathbf{x_{in}} \in \mathcal{R}^{n+2p}$, where n and p denote the dimensions of the current control state $\mathbf{x}$ and the parameters μ and σ, respectively. The hidden state $h \in \mathcal{R}^q$ is updated using the GRU decoder as illustrated in the Eq.4,

$$\mathbf{h_t} = GRU(\mathbf{x_{in}}, \mathbf{h_{t-1}}) \tag{4}$$

where h_{t-1}, which serves as $h_{in}$is the input hidden state for the GRU at the current time step t.

The hidden state h and a trajectory-guided hidden state $u_{traj_{hidden}}$ are subsequently used to compute a waypoint-based attention map wp_A using another self-attention mechanism. This attention map is applied to $I_{C_{feat}}$ to produce a new feature embedding, which is then combined with h to obtain the merged feature. This merged feature updates $\mathbf{x_{in}}$ as shown in Eq.5:

$$\mathbf{x_{in}} = x + dx + \mu + \sigma \tag{5}$$

Throughout the loop, the model refines these variables iteratively, generating a sequence of multi-step control predictions $(u_0, u_1, \ldots, u_{\mathcal{K}})$ that are dynamically and temporally coherent. Incorporating self-attention mechanisms into the architecture significantly enhances the model's capacity for sequential decision-making.

*3) **Gating Network:*** The gating network, as illustrated in Fig.1, serves as high-level decision-making in fusing the trajectory and control prediction outputs to yield an optimized and context-aware command to the vehicle actuators. The primary objective in designing the gating network is to fuse it with situational awareness capabilities. It aims to dynamically evaluate and choose between trajectory-based controls $\mathbf{u_{traj}} = \mathcal{I}(\mathcal{W})$ and direct control signals $\mathbf{u_{ctrl}} = (u_0, u_1, \ldots, u_{\mathcal{K}})$ from the control prediction branch. This enables the gating network to make context-sensitive decisions in various driving scenarios, such as navigating intersections, executing turns, or overtaking other vehicles.

To this end, the gating network generates two outputs: a high-level command $\mathbf{g_\Phi}$ and a situation-fusing parameter $\mathbf{g}_\alpha$, respectively, by receiving the combined feature vector $\mathbf{F}$ as input. The high-level command g_Φ encompasses a set of commands including 'straight', 'left turn', 'right turn', 'lane-following', 'change lane to the left', and 'change lane to the right'. The g_Φ is an auxiliary information that is predicted from the proposed MAGNet framework. Mathematically, let $\mathbf{F}$ represent the situation context derived from the sensor information (e.g., image and measurements); the gating network can be expressed as in Eq.6:

$$(\mathbf{g}_\Phi, \mathbf{g}_\alpha) = \mathbf{G}(\mathbf{F}, \mathbf{u_{traj}}, \mathbf{u_{ctrl}}) \tag{6}$$

$\mathbf{g}_\alpha$ is a function of $\mathbf{F}$ and the outputs $\mathbf{u_{traj}}$, $\mathbf{u_{ctrl}}$ from the trajectory and control branches as illustrated in Eq.7:

$$\mathbf{g}_\alpha = softmax\left(W_{\mathbf{g}_\alpha} \cdot [\mathbf{F}; \mathbf{u_{traj}}; \mathbf{u_{ctrl}}] + b_{\mathbf{g}_\alpha}\right) \tag{7}$$

where $W_{\mathbf{g}_\alpha}$ and $b_{\mathbf{g}_\alpha}$ are learnable parameters. The $softmax$ function ensures $\mathbf{g}_\alpha$ is a probabilistic weighting factor in the $[0, 1]$ range. The high-level command g_Φ network outputs the discrete high-level commands and is expressed as in Eq.8

$$\mathbf{g}_\Phi = W_{out} \cdot ReLU(W_{hidden} \cdot BN(W_{in} \cdot \mathcal{F} + b_{in}) + b_{hidden}) + b_{out} \tag{8}$$

Finally, the output control action $\mathcal{P}$ is a weighted sum of $\mathbf{u_{traj}} = \mathcal{I}(\mathcal{W})$ and $\mathbf{u_{ctrl}} = (u_0, u_1, \ldots, u_{\mathcal{K}})$, modulated by $\mathbf{g}_\alpha$ is given by Eq.9 as:

$$\mathcal{P} = \mathbf{g}_\alpha \cdot \mathbf{u_{traj}} + (1 - \mathbf{g}_\alpha) \cdot (u_0, u_1, ..., u_{\mathcal{K}}) \tag{9}$$

The model can thus adaptively balance long-term planning and immediate reactive behaviors, making it highly robust and adaptive to a variety of dynamically changing environments.

*4) **Loss Design:*** The MAGNet framework includes trajectory planning loss $\mathcal{L}_{traj}$, control prediction loss $\mathcal{L}_{ctrl}$, auxlilary loss $\mathcal{L}_{aux}$ and the gating loss $\mathcal{L}_{\mathbf{G}}$. Since the MAGNet focuses on incorporating the trajectory and control prediction in a unified framework with situation-based fusion, the proposed method was trained in two phases. In phase one, the trajectory and control prediction branches are trained end-to-end without a gating network and then frozen for training the gating network in the second phase.

The trajectory loss $\mathcal{L}_{traj}$ can be expressed as shown in Eq.10

$$\mathcal{L}_{traj} = \sum_{i=1}^{\mathcal{K}} \|\mathbf{w_i} - \hat{\mathbf{w}}_\mathbf{i}\|_1 + \lambda_F \cdot \mathcal{L}_\mathcal{F}\left(f_{traj}^{(0)}, f_{Expert}^{(0)}\right) \tag{10}$$

where $\mathbf{w_i}$, and $\hat{\mathbf{w}}_\mathbf{i}$ signify the predicted and ground-truth waypoints at time i, respectively. λ_F serves as a tunable weight for the feature loss $\mathcal{L}_\mathcal{F}$, which computes the L_2 distance between $f_{traj}^{(0)}$ and $f_{Expert}^{(0)}$ at the current time step, thereby acting as an auxiliary supervisory signal. $f_{traj}^{(0)}$ is the feature representation of the predicted trajectory at the initial state and $f_{Expert}^{(0)}$ represent the feature representation from the expert demonstration. For the control prediction, the $\mathcal{L}_{ctrl}$ loss is expressed in Eq.11

$$\begin{aligned}
\mathcal{L}_{ctrl} = {} & KL(\text{Beta}(\mathbf{a}_0) \parallel \text{Beta}(\hat{\mathbf{a}}_0)) \\
& + \frac{1}{\mathcal{K}} \sum_{i=1}^{\mathcal{K}} \text{KL}(\text{Beta}(\mathbf{a}_i) \parallel \text{Beta}(\hat{\mathbf{a}}_i)) \\
& + \lambda_F \cdot \mathcal{L}_F(f_{ctrl}^0, f_{Expert}^0) \\
& + \frac{1}{\mathcal{K}} \sum_{i=1}^{\mathcal{K}} \mathcal{L}_F(f_{ctrl}^i, f_{Expert}^i)
\end{aligned} \tag{11}$$

The loss function $\mathcal{L}_{ctrl}$ comprises four terms. It uses Kullback-Leibler (KL) divergence to measure the difference between predicted and ground-truth Beta distributions, initially and over future time steps i. A feature loss, weighted by λ_F, enhances the model's learning at each time step. The aggregated loss $\mathcal{L}$ for phase one is:

$$\mathcal{L} = \lambda_{traj} \cdot \mathcal{L}_{traj} + \lambda_{ctrl} \cdot \mathcal{L}_{ctrl} + \lambda_{aux} \cdot \mathcal{L}_{aux} \tag{12}$$

where $\mathcal{L}_{aux}$ is the weighted sum of L_1 loss for speed prediction and L_2 loss for the value prediction, respectively.

After training, the trajectory and control prediction branches are fixed, and the gating network $\mathbf{G}$ is trained end-to-end. The loss function $\mathcal{L}_{\mathcal{G}}$ is expressed in Eq.13.

$$\begin{aligned}\mathcal{L}_{\mathbf{G}} = \sum_{i\in\{\text{steer, throttle, brake}\}} & \lambda_i \cdot \left((o_{\text{combined},i} - o_{\text{traj},i})^2\right. \\ & \left.+(o_{\text{combined},i} - o_{\text{ctrl},i})^2\right) \\ & + \lambda_{command} \cdot \mathcal{L}_{command} \\ & + \lambda_{\text{L1}} \cdot \mathcal{L}_1 \end{aligned} \tag{13}$$

Here, λ_i, $\lambda_{command}$ and λ_{L1} are the hyper-parameters. $\mathcal{L}_{command}$ is the loss between predicted high-level command and ground-truth as expressed by Eq.14, whereas $\mathcal{L}_1$ corresponds to regularization term given by Eq.15

$$\mathcal{L}_{command} = -\sum_{c=1}^{6} y_c \log(p_c) \tag{14}$$

$$\mathcal{L}_1 = \sum_j |\theta_j| \tag{15}$$

The combined output $o_{combined,i}$ for each control signal $i \in steer, throttle, brake$ is computed as a weighted average of the outputs from the trajectory and control prediction branches, denoted as $o_{traj,i}$, and $o_{ctrl,i}$ respectively. The weights $\alpha_{traj,i}$ and $\alpha_{ctrl,i}$ modulate these contributions as expressed in Eq.16.

$$o_{combined,i} = \frac{\tanh(\alpha_{traj,i} \cdot o_{traj,i}) + \tanh(\alpha_{ctrl,i} \cdot o_{ctrl,i})}{2} \tag{16}$$

IV. Experiments

A. Benchmark

In this work, CARLA simulator used for the closed-loop evaluation of the proposed method [34]. We have used two widely used benchmarks, **Town05 Long** and **Longest6** [3], where the Longest6 benchmark uses the six longest routes of each town (Town01-Town06) comprising 36 routes. In each benchmark, the routes are defined by a sequence of navigation points together with sensor and high-level command data (turn right/left, lane changing and following, straight). The task in closed-loop driving is to drive the autonomous agent to the desired destination by simulating the traffic situation and also include challenging scenarios, for instance, obstacle avoidance, crossing unprotected intersections, and sudden control loss.

B. Data Collection

In our experiments, we choose Roach [16] for the supervision as an expert model. Roach is an RL-trained model incorporating privileged information, including roads, routes, lanes, vehicles, pedestrians, and traffic elements, rendered into a 2D bird-eye-view (BEV) image. This learning-based expert offers advantages over rule-based experts by providing a richer set of information beyond just direct supervision signals.

For data generation, we adhere to the protocol outlined in [10], rolling out an expert policy with privileged information to gather the dataset using the CARLA simulator. Our data collection settings utilize a monocular camera (front-facing), IMU, GPS, and speedometer. We have collected data in Town01, Town03, Town04, and Town06, under various environmental conditions, resulting in 189K data points for training.

C. Evaluation Metrics

Our model's performance is assessed using CARLA Leaderboard metrics, focusing on **Route Completion (RC)** for measuring route success, **Infraction Score (IS)** for traffic rule adherence, and **Driving Score (DS)** as the primary metric combining RC and IS for a holistic performance evaluation [34] [3] [9].

D. Training Details

The training of the MAGNet is done in two phases. In the first phase, the trajectory and control prediction branches are trained end-to-end. For this, the image encoder adopts ResNet architecture trained on ImageNet [35]. The size of the input RGB image is 900×256, with the FOV of the camera set to $100\,\text{deg}$. In the trajectory and control branch, the $\mathcal{T} = 4$ corresponds to the next four future steps at $2HZ$. For the PID settings, we follow the same settings as proposed in [3], where the values of $K_p = 5.0$, $K_d = 1.0$ and $K_i = 0.5$ are for the longitudinal control, and the values of PID controllers are $K_p = 0.75$, $K_d = 0.3$ and $K_i = 0.75$ for lateral control. The hyper-parameters used in the training for phase one are as follows: $\lambda_{traj} = 1$, $\lambda_{ctrl} = 1$, $\lambda_F = 0.05$ and $\lambda_{aux} = 0.05$. For the training of the gating network $(\mathcal{G})$, the hyper-parameters are set as follows: $\lambda_{steer} = 1$, $\lambda_{throttle} = 1$, $\lambda_{brake} = 1$, $\lambda_{command} = 1$, and $\lambda_{L_1} = 0.5$. The training

TABLE I: Comparison of MAGNet with state-of-the-art methods on Town05 Long benchmark in terms of driving score (DS), route completion (RC) and infraction score (IS).†shows MAGNet results without attention.

Methods	**Sensors**		**Metrics**		
	RGB	**Lidar**	**DS ↑**	**RC ↑**	**IS ↑**
CILRS [25] [20]	✓	✗	7.80 ± 0.30	10.30 ± 0.00	0.75 ± 0.05
LBC [1] [20]	✓	✗	12.30 ± 2.00	31.90 ± 2.20	0.66 ± 0.02
Transfuser [3] [20]	✓	✓	31.00 ± 3.60	47.50 ± 5.30	0.77 ± 0.04
Roach [16] [20]	✓	✗	41.60 ± 1.80	96.40 ± 2.10	0.43 ± 0.03
LAV [9] [20]	✓	✓	46.50 ± 2.30	69.80 ± 2.30	0.73 ± 0.02
TCP [10] [20]	✓	✗	57.20 ± 1.50	80.40 ± 1.50	0.73 ± 0.02
Think Twice [20]	✓	✓	65.00 ± 1.70	95.50 ± 2.00	0.69 ± 0.05
Ours w/o attn†	✓	✗	69.74 ± 1.81	96.25 ± 2.30	0.71 ± 0.03
Ours	✓	✗	73.30 ± 3.90	98.50 ± 1.18	0.79 ± 0.05

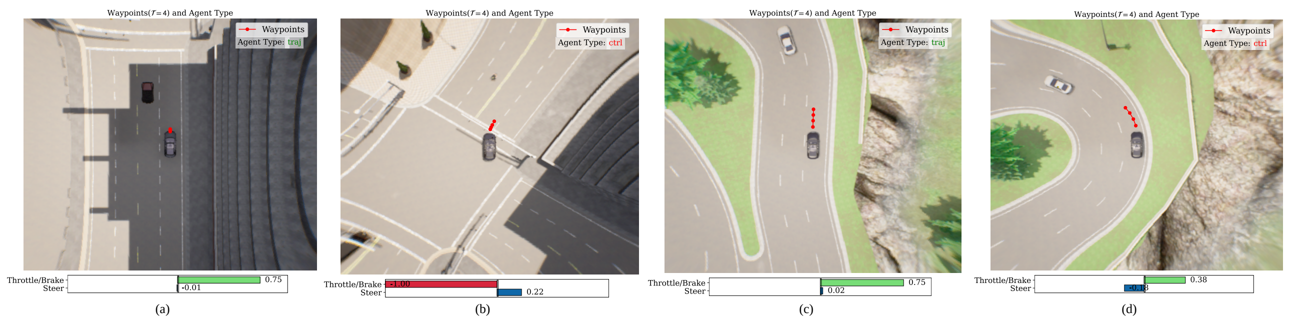


Fig. 4: **Visualization of MAGNet's decision-making on CARLA benchmarks: (a,b) show urban driving in Town05 Long, (c,d) in Longest6, highlighting 'traj' and 'ctrl' modes and the gating network's effectiveness.**

for both phases is done on 2 Nvidia V100 GPUs, having a memory of 32GB each. The Adam optimizer [36] is used for each training phase with a learning rate of 5×10^{-4} and weight decay of 1×10^{-7}. In both training phases, the models are trained for 60 epochs having a batch size of 64.

E. Results

We compare the proposed method MAGNet with other state-of-the-art methods on two publicly available benchmarks, **Town05 Long** and **Longest6**, in closed-loop settings. Table-I illustrates the quantitative results of MAGNet with the state-of-the-art methods on **Town05 Long** benchmark. In our quantitative evaluation, the proposed method is equally compared to the camera and Lidar-based state-of-the-art methods. As the MAGNet employs a monocular camera for predicting the diving policies, it obtains better driving, route completion, and infraction scores when compared with camera-based driving agents. Specifically, MAGNet achieves a driving score of 73.3 ± 3.9, 98.5 ± 1.18 of route completion, and an infraction score of 0.69 ± 0.05, outperforming the ThinkTwice [20] (a camera-based driving agent) by 12.8% in driving, 3.1% in route completion and 14.5% in infraction scores, respectively. Similarly, MAGNet also performs better when compared with Lidar-based methods; for instance, MAGNet outperforms LAV(a Lidar-based driving agent) [9] by a margin of 57.6% in driving score, 41.1% in route completion and 8.2% in infraction score respectively. Since MAGNet follows a multi-task learning framework, we compared our method to the baseline method TCP [10], which also follows the multi-task learning framework. Upon evaluation, the MAGNet outperforms the TCP [10] baseline method by 28.2% in driving score, 22.5% in route completion, and 8.2% in infraction score, respectively.

As for the **Longest6** benchmark, MAGNet has also shown better performance when compared with state-of-the-art methods as illustrated in Table-II. For instance, MAGNet achieves the driving score of 71.43 ± 2.3, route completion score of 84.54 ± 1.5, and infraction score of 0.87 ± 0.05, as compared to TCP [10], where it achieves the driving score of 42.86 ± 0.63, route completion score of 61.83 ± 4.19 and 0.71 ± 0.04 of infraction score. Thus, MAGNet outperforms TCP [10] by a margin of 66.7% in driving score, 36.7% in route completion, and 22.5% in infraction score on **Longest6** benchmark, respectively. Similarly, when the proposed MAGNet is compared with camera and Lidar-based methods, it performs better in the driving, route completion, and infraction scores, as illustrated in Table-II on **Longest6** benchmark.

TABLE II: Comparison of MAGNet with state-of-the-art methods on Longest6 benchmark in terms of driving score (DS), route completion (RC) and infraction score (IS).†shows MAGNet results without attention.

Methods	**Sensors**		**Metrics**		
	RGB	**Lidar**	**DS ↑**	**RC ↑**	**IS ↑**
LAV [9] [31]	✓	✓	48.41 ± 3.40	80.71 ± 0.84	0.60 ± 0.04
Transfuser [3] [31]	✓	✓	46.20 ± 2.57	83.61 ± 1.16	0.57 ± 0.00
WOR [13] [31]	✓	✗	17.36 ± 2.95	43.46 ± 2.99	0.54 ± 0.06
NEAT [2] [31]	✓	✗	24.08 ± 3.30	59.94 ± 0.50	0.49 ± 0.02
TCP [10] [31]	✓	✗	42.86 ± 0.63	61.83 ± 4.19	0.71 ± 0.04
CAT [31]	✓	✗	58.36 ± 2.24	78.79 ± 1.50	0.77 ± 0.02
Think Twice [20]	✓	✓	66.70	77.20	0.84
Ours w/o attn†	✓	✗	68.32 ± 2.50	79.17 ± 3.87	0.82 ± 0.02
Ours	✓	✗	71.43 ± 2.30	84.54 ± 1.50	0.87 ± 0.05

The efficacy of MAGNet is illustrated in Fig.4, showcasing adaptability in various driving scenarios. The qualitative findings align well with quantitative benchmarks, substantiating its comparative effectiveness against state-of-the-art methods.

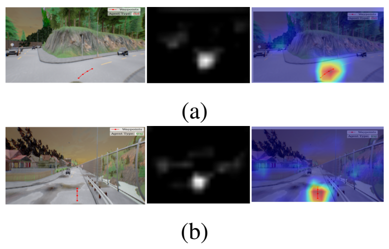


Fig. 5: **Attention map visualizations for MAGNet:** (a) showing 'traj' mode selection highlighted by focused attention regions, and (b) illustrating 'ctrl' mode selection where attention disperses relevant to control adjustments. These attention maps illustrate that the model is learning the representations.

F. Ablation Study

This section presents a quantitative analysis of control-only, trajectory-only, and our proposed method, using a uniform feature extraction process with a ResNet-based image encoder and a measurement encoder. The control-only model uses only

TABLE III: Ablation Study between ours Control-only, Trajectory-only, TCP, ours (heuristic) method, and ours (MAGNet).

Methods	**Sensors**		**Metrics**		
	RGB	**Lidar**	**DS** ↑	**RC** ↑	**IS** ↑
Control-only	✓	✗	45.20 ± 1.54	71.56 ± 1.50	0.47 ± 0.07
Trajectory-only	✓	✗	39.50 ± 1.96	63.87 ± 2.23	0.55 ± 0.05
TCP [10] [20]	✓	✗	57.20 ± 1.50	80.40 ± 1.50	0.73 ± 0.02
Ours (heuristic)	✓	✗	65.70 ± 1.64	89.34 ± 1.50	0.72 ± 0.03
Ours (MAGNet)	✓	✗	73.30 ± 3.90	98.50 ± 1.18	0.79 ± 0.05

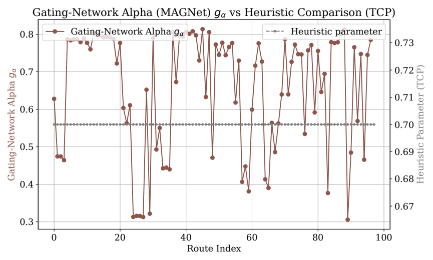
the control branch, while the trajectory-only model uses only the trajectory branch. Control-only predictions use the feature vector $\mathbf{F}$ and trajectory-only predictions down-sample $\mathbf{F}$ for the GRU decoder to forecast future waypoints. As shown in Table-III, control-only exhibits higher reactivity but more infractions, and trajectory-only shows lower route completion, both under-performing compared to our proposed method, which combines both approaches with a situational gating network, leading to superior performance metrics. Additional we have extended our ablation study to include a heuristic-based combination of control-only and trajectory-only module. We have adopted the same heuristic-based approach used in TCP for fair comparative analysis. While the heuristic approach improved over the individual control-only and trajectory-only models, it still did not achieve the performance level of our integrated MAGNet approach as illustrated in Table-III.

We have conducted a statistical analysis to evaluate the effectiveness of our MAGNet model. Our study assesses MAGNet's efficacy, focusing on the g_α parameter within its gating network and comparing it to TCP's heuristics approach. We investigated the impact of these parameters on throttle, brake, and steer controls. Moreover, we demonstrated MAGNet's adaptability to environmental changes through attention maps, as shown in Fig.5. Table-I—II shows a quantitative comparison between MAGNet without attention and the proposed MAGNet with attention.

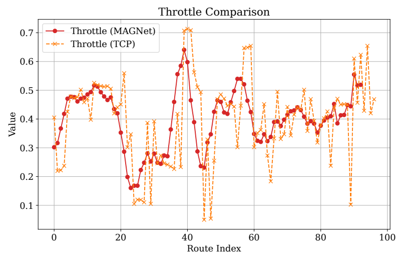
Fig.6 (a—d) details the results of this comprehensive analysis, comparing MAGNet with TCP across different routes and driving conditions. It highlights instances where the agent alternates between 'traj' (trajectory) and 'ctrl' (control) modes in response to varying situations. Notably, we found that MAGNet's throttle, brake, and steering profiles are significantly smoother than those of TCP, demonstrating the efficacy of our model. Additionally, the analysis reveals the adaptive behavior of the g_α parameter in MAGNet, which dynamically adjusts based on the driving context. We also present a distribution of 'traj' and 'ctrl' modes across various routes in Fig.7. This distribution reveals that 'ctrl' mode is favored at lower alpha values and 'traj' mode at higher ones, indicating their respective suitability for different driving scenarios.

V. Conclusion

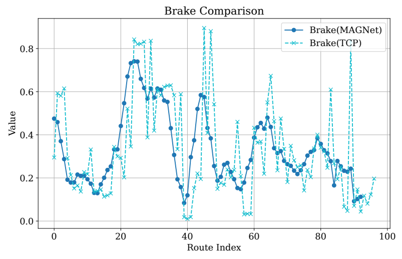
In this work, we present MAGNet, a framework designed to learn situational fusion strategies that integrate trajectory and direct control predictions. We also develop a trajectory-distilled control prediction technique that leverages



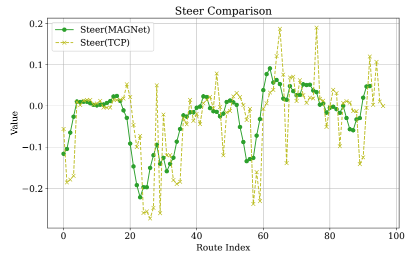
(a)



(b)



(c)



(d)

Fig. 6: **Statistical comparison of MAGNet and TCP:** (a) shows MAGNet's (g_α) and TCP's heuristic parameters; (b-d) display their impacts on throttle, brake, and steering controls.

self-attention for multi-step control output predictions. Our findings indicate that the situational fusion parameter can be effectively learned without resorting to heuristic methods for merging trajectory and control predictions. Notably, our proposed approach surpasses the leading TCP method in a closed-loop setting across two widely recognized benchmarks. Furthermore, compared to state-of-the-art methods, including those using camera and Lidar-based agents, MAGNet performs better in driving score, route completion, and infraction score metrics.

The challenge of effectively fusing situation-based parameters in autonomous driving remains an open issue. While our proposed work takes a significant step forward by adaptively

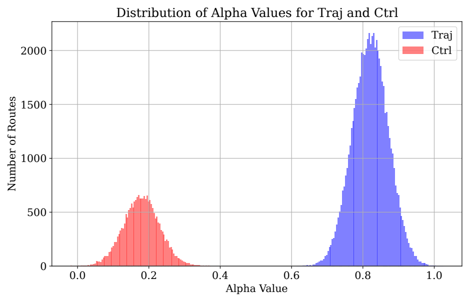


Fig. 7: **Alpha-Driven Mode Distribution:** 'ctrl' mode is common at lower alphas, 'traj' at higher, indicating strategic mode selection.

learning the situation-based fusing parameter, it still needs to incorporate rules-based methods. Specifically, combining signal-temporal-logic (STL) with adaptive learning introduces complexities in harmonizing these adaptive approaches with established rules. The key challenge lies in ensuring their cohesive operation to improve system safety and efficiency, presenting a promising avenue for future research.

References

[1] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Conference on Robot Learning*. PMLR, 2020, pp. 66–75.

[2] K. Chitta, A. Prakash, and A. Geiger, "Neat: Neural attention fields for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 793–15 803.

[3] K. Chitta, A. Prakash, B. Jaeger, Z. Yu, K. Renz, and A. Geiger, "Transfuser: Imitation with transformer-based sensor fusion for autonomous driving," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[4] Y. Hu, J. Yang, L. Chen, K. Li, C. Sima, X. Zhu, S. Chai, S. Du, T. Lin, W. Wang *et al.*, "Planning-oriented autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 853–17 862.

[5] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 4693–4700.

[6] X. Liang, T. Wang, L. Yang, and E. Xing, "Cirl: Controllable imitative reinforcement learning for vision-based self-driving," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 584–599.

[7] E. Ohn-Bar, A. Prakash, A. Behl, K. Chitta, and A. Geiger, "Learning situational driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11 296–11 305.

[8] S. Casas, A. Sadat, and R. Urtasun, "Mp3: A unified model to map, perceive, predict and plan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 403–14 412.

[9] D. Chen and P. Krähenbühl, "Learning from all vehicles," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 222–17 231.

[10] P. Wu, X. Jia, L. Chen, J. Yan, H. Li, and Y. Qiao, "Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6119–6132, 2022.

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

[12] 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 *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8248–8254.

[13] D. Chen, V. Koltun, and P. Krähenbühl, "Learning to drive from a world on rails," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 15 590–15 599.

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

[15] M. Toromanoff, E. Wirbel, and F. Moutarde, "End-to-end model-free reinforcement learning for urban driving using implicit affordances," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7153–7162.

[16] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. Van Gool, "End-to-end urban driving by imitating a reinforcement learning coach," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 15 222–15 232.

[17] H. Shao, L. Wang, R. Chen, S. L. Waslander, H. Li, and Y. Liu, "Reasonnet: End-to-end driving with temporal and global reasoning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 723–13 733.

[18] 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 *European Conference on Computer Vision*. Springer, 2022, pp. 533–549.

[19] B. Jaeger, K. Chitta, and A. Geiger, "Hidden biases of end-to-end driving models," *arXiv preprint arXiv:2306.07957*, 2023.

[20] X. Jia, P. Wu, L. Chen, J. Xie, C. He, J. Yan, and H. Li, "Think twice before driving: Towards scalable decoders for end-to-end autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 21 983–21 994.

[21] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, vol. 1, 1988.

[22] U. Muller, J. Ben, E. Cosatto, B. Flepp, and Y. Cun, "Off-road obstacle avoidance through end-to-end learning," *Advances in neural information processing systems*, vol. 18, 2005.

[23] H. Xu, Y. Gao, F. Yu, and T. Darrell, "End-to-end learning of driving models from large-scale video datasets," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2174–2182.

[24] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," *arXiv preprint arXiv:1812.03079*, 2018.

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

[26] Z. Huang, H. Liu, J. Wu, and C. Lv, "Differentiable integrated motion prediction and planning with learnable cost function for autonomous driving," *IEEE transactions on neural networks and learning systems*, 2023.

[27] X. Liang, Y. Wu, J. Han, H. Xu, C. Xu, and X. Liang, "Effective adaptation in multi-task co-training for unified autonomous driving," *Advances in Neural Information Processing Systems*, vol. 35, pp. 19 645–19 658, 2022.

[28] Y. Zhang, Z. Zhu, W. Zheng, J. Huang, G. Huang, J. Zhou, and J. Lu, "Beverse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving," *arXiv preprint arXiv:2205.09743*, 2022.

[29] X. Jia, Y. Gao, L. Chen, J. Yan, P. L. Liu, and H. Li, "Driveadapter: Breaking the coupling barrier of perception and planning in end-to-end autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 7953–7963.

[30] I. Kim, H. Lee, J. Lee, E. Lee, and D. Kim, "Multi-task learning with future states for vision-based autonomous driving," in *Proceedings of the Asian Conference on Computer Vision*, 2020.

[31] J. Zhang, Z. Huang, and E. Ohn-Bar, "Coaching a teachable student," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7805–7815.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[33] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[34] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.