Think Twice before Driving: Towards Scalable Decoders for End-to-End Autonomous Driving

Xiaosong Jia[1,2], Penghao Wu[2,3], Li Chen[2], Jiangwei Xie[2], Conghui He[2], Junchi Yan[1,2†], Hongyang Li[2,1†]

[1] Shanghai Jiao Tong University [2] Shanghai AI Laboratory
[3] University of California at San Diego
†Correspondence authors

https://github.com/OpenDriveLab/ThinkTwice

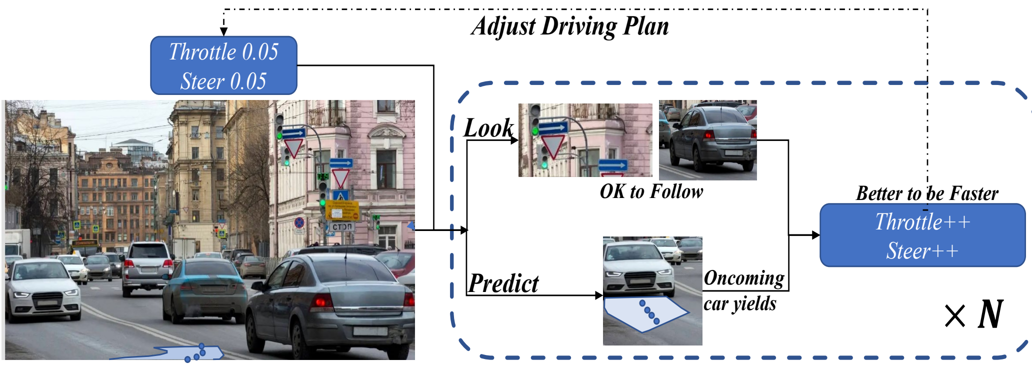

Figure 1. We propose **ThinkTwice**, a scalable decoder paradigm that generates the future trajectory and action of the ego vehicle sfor end-to-end autonomous driving. Conditioned on the coarse action/trajectory, we propose the *Look Module* to retrieve information from critical regions and the *Prediction Module* to anticipate the outcome of the ego behavior. Taking features from the two modules as input, the coarse prediction is refined by predicting its offset from ground-truth. We could stack multiple such layers and scale up the capacity of the decoder with dense supervision and spatial-temporal prior.

Abstract

End-to-end autonomous driving has made impressive progress in recent years. Existing methods usually adopt the decoupled encoder-decoder paradigm, where the encoder extracts hidden features from raw sensor data, and the decoder outputs the ego-vehicle's future trajectories or actions. Under such a paradigm, the encoder does not have access to the intended behavior of the ego agent, leaving the burden of finding out safety-critical regions from the massive receptive field and inferring about future situations to the decoder. Even worse, the decoder is usually composed of several simple multi-layer perceptrons (MLP) or GRUs while the encoder is delicately designed (e.g., a combination of heavy ResNets or Transformer). Such an imbalanced resource-task division hampers the learning process.

In this work, we aim to alleviate the aforementioned problem by two principles: (1) fully utilizing the capacity of the encoder; (2) increasing the capacity of the decoder. Concretely, we first predict a coarse-grained future position and action based on the encoder features. Then, conditioned on the position and action, the future scene is imagined to check the ramification if we drive accordingly. We also retrieve the encoder features around the predicted coordinate to obtain fine-grained information about the safety-critical region. Finally, based on the predicted future and the retrieved salient feature, we refine the coarse-grained position and action by predicting its offset from ground-truth. The above refinement module could be stacked in a cascaded fashion, which extends the capacity of the decoder with spatial-temporal prior knowledge about the conditioned future. We conduct experiments on the CARLA simulator and achieve state-of-the-art performance in closed-loop benchmarks. Extensive ablation studies demonstrate the effectiveness of each proposed module.

1. Introduction

With the advance in deep learning, autonomous driving has attracted attention from both academia and industry. End-to-end autonomous driving [46, 48] aims to build a fully differentiable learning system that is able to map the raw sensor input directly to a control signal or a future trajectory. Due to its efficiency and ability to avoid cumulative errors, impressive progress has been achieved in recent years [3, 10, 12, 15, 16, 56]. State-of-the-art works [9,27,49,57,67,68] all adopt the encoder-decoder paradigm. The encoder module extracts information from raw sensor data (camera, LiDAR, Radar, *etc.*) and generates a representation feature. Taking the feature as input, the decoder directly predicts way-points or control signals.

Under such a paradigm, the encoder does not have access to the intended behavior of the ego agent, which leaves the burden of finding out the safety-critical regions from the large perceptive field of massive sensor inputs and inferring about the future situations to the decoder. For example, when the ego vehicle is at the intersection, if it decides to go straight, it should check the traffic light across the road, which might consist of only several pixels. If it decides to go right, then it should check whether there are any agents on its potential route and think about how they would react to the ego vehicle's action. Even worse, the decoder is usually several simple multi-layer perceptrons (MLP) or GRUs while the encoder is a delicately designed combination of the heavy ResNet or Transformer. Such unmatched resource-task division hampers the overall learning process.

To address the aforementioned issues, we design our new model based on two principles:

- **Fully utilize the capacity of the encoder**. Instead of leaving all future-related tasks to the decoder, we should reuse the features from the encoder conditioned on the predicted decision.
- **Extend the capacity of the decoder with dense supervision**. Instead of simply adding depth/width of MLP which would cause severe overfit, we should enlarge the encoder with prior structure and corresponding supervision so that it could capture the inherent driving logical reasoning.

To instantiate these two principles, we propose a cascaded decoder paradigm to predict the future action of the ego vehicle in a coarse-to-fine fashion as shown in Fig. 1. Concretely, (i) We first adopt an MLP similar to classical approaches to generate the coarse future trajectory and action. (ii) We then retrieve features around the predicted future location from the encoder and further feed them into several convolutional layers to obtain goal-related scene features (we denote the module as *Look Module* and the feature as *Look Feature*). This follows the intuition that human drivers would check their intended target to ensure safety and legitimacy. (iii) Inspired by the fact that human drivers would anticipate other agents' future motion to avoid possible collisions, we design a *Prediction Module*, which takes the coarse action and features of the current scene as input and generates future scene representation features (denoted as *Prediction Feature*). Considering the difficulty of obtaining supervision of the future scene representation conditioned on the predicted action during open-loop imitation learning, we adopt the teach-forcing technique [2]: during training, we additionally feed samples with ground-truth action/trajectory into Prediction Module and supervise the corresponding *Prediction Feature* with ground-truth future scene. As for the target of the supervision, we choose features from Roach [77], an RL-based teacher network with privileged input, which contains decision-related information. (iv) Based on the *Look Feature* and *Prediction Feature*, we predict the offset between the coarse prediction and ground-truth for refinement. The aforementioned process could be stacked cascadedly, which enlarges the capacity of the decoder with spatial-temporal prior knowledge about the conditioned future.

We conducted experiments on two competitive closed-loop autonomous driving benchmarks with CARLA [21] and achieved state-of-the-art performance. We also conducted extensive ablation studies to demonstrate the effectiveness of the components of the proposed method.

In summary, our work has three-fold contributions:

1. We propose a scalable decoder paradigm for end-to-end autonomous driving, which, to the best of our knowledge, is the first to emphasize the importance of enlarging the capacity of the decoder in this field.
2. We devise a decoder module to look back to the safety critical areas and anticipate the future scene conditioned on the predicted action/trajectory, which injects spatial-temporal prior knowledge and dense supervisions into the training process.
3. We demonstrate state-of-the-art performance on two competitive benchmarks and conduct extensive ablation studies to verify the effectiveness of the proposed module.

We believe that the decoder (decision part) is equally important as the encoder (perception part) in end-to-end autonomous driving. We hope our exploration could inspire further efforts in this line of study for the community.

2. Related Work

2.1. End-to-end Autonomous Driving

Unlike traditional modular autonomous driving frameworks, end-to-end methods, which predict actions based on sensor observations, have shown great potential. CIL [15] proposes a simple structure to directly map front-view image features to control signals based on navigation commands. Based on that, CILRS [16] adds a speed prediction branch to alleviate the inertia problem. LBC [10] and

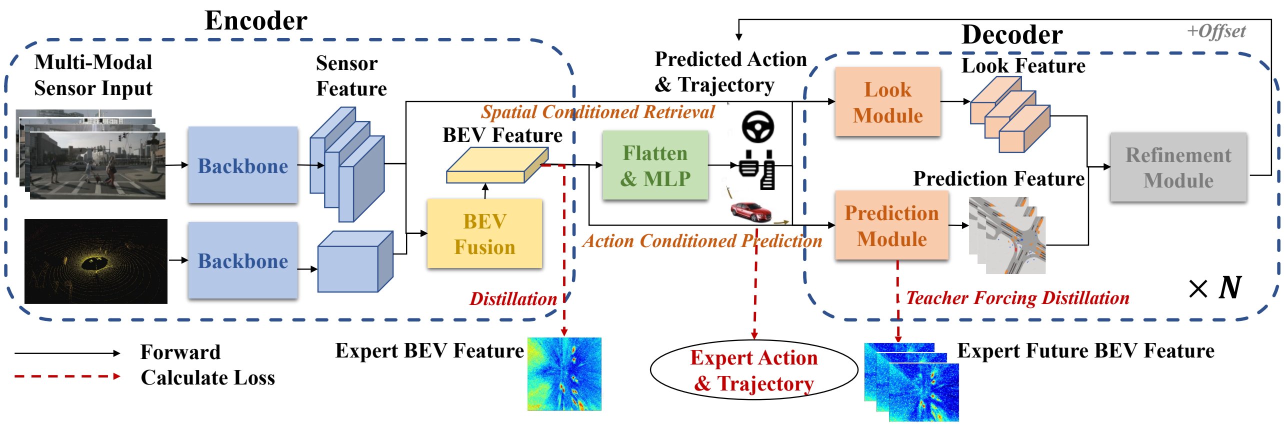

Figure 2. **Overall architecture of ThinkTwice.** (a) The encoder first processes raw data from different sensors with corresponding backbones and then fuses them into a bird's-eye-view (BEV) representation. (b) With BEV features as input, the decoder first generates the coarse trajectory/action of the ego agent. Then, the *Look Module* retrieves sensor features around the predicted location - *Look Feature* and the *Prediction Module* anticipates the future BEV features conditioned on the predicted action - *Prediction Feature*. (c) Based on these two features, the refinement module predicts the offset between the coarse output and the ground-truth (GT).

Roach [77] use learning-based privilege experts to better teach student models. Control and trajectory planning are combined in [68] in a multi-task learning approach. Reinforcement learning methods [7, 64] use useful representations from pre-training tasks to accelerate the training process. NEAT [12] decodes way-points and semantics in bird's-eye-view (BEV). Effective sensor fusion approaches to extract useful features from sensors for planning are explored in [9, 49, 57, 76]. Model-based imitation learning is studied in [27] to explicitly model the environment.

Such methods have gained impressive performance in closed-loop evaluation. Nevertheless, most of them focus on the encoder part only, usually adopting a simple MLP or GRU-based decoder for final planning. In our work, we explore increasing the capacity of the decoder and fully exploiting the capacity of the encoder simultaneously.

2.2. BEV Representation for Autonomous Driving

Learning BEV representation for perception and planning tasks in autonomous driving is a heated topic [36] in both industry and academia. The BEV representation inherently preserves the spatial relationships on the ground plane, making it preferable for joint perception-planning and sensor fusion.

Perception tasks in BEV including detection [31, 37, 39, 41, 51, 54], segmentation [28, 39, 47, 69, 78], and lane detection [4, 11, 23] have been rapidly pushing the frontier of 3D vision for autonomous driving. For planning, BEV representation has also shown great potential, where the model could reason about the important geometry relationships. ChauffeurNet [1] renders the privileged environment and routed information to BEV as input for planning. LBC [10], LAV [9] and Roach [77] train a strong expert based on privileged BEV representation as input. The cost-maps for planning can also be constructed or learned based on BEV representations [6, 17, 28, 30, 55, 73, 74]. NEAT [12] updates an attention map iteratively to aggregate features and decode them to semantic categories and way-points in BEV. However, NEAT does not explicitly convert image features in perspective view to BEV, and it only uses a simple MLP to directly decode results from the aggregated feature. In our work, after explicitly projecting and aligning the image feature with LiDAR feature in BEV, we retrieve BEV features in critical regions and anticipate the future scene to iteratively refine the planning outputs.

2.3. Coarse-to-fine Strategy

The coarse-to-fine strategy has been widely studied and used in the field of computer vision. Typical two-stage methods for 2D detection [24, 25, 52] and 3D detection [18, 38, 58, 60, 71, 72] usually first propose coarse region proposals and then extract features based on the proposals to generate the refined final predictions. The coarse-to-fine approach also gains great success in tasks like optical flow estimation [22, 32, 61, 63], salient object detection [19,40,50,65], and trajectory prediction [33–35,44,59].

As for planning in autonomous driving, LAV [9] also iteratively refines the predicted way-points. However, their refinement is only based on the original feature with a simple RNN. Our method uses the coarse prediction to retrieve features in critical regions and anticipate the future to better refine the coarse prediction.

3. Approach

The proposed ThinkTwice is an end-to-end autonomous driving framework consisting of an encoder to transform the raw sensor data into a representation vector, and a decoder to generate future trajectories or actions of the ego agent based on the representation vector. The overall architecture is shown in Fig. 2.

3.1. BEV Encoder

In this work, we consider two commonly used sensors in autonomous driving: cameras and LiDAR. To fuse their information, we first transform the raw sensor data into bird's-eye-view (BEV) features respectively, and then directly concatenate BEV features since they have already been aligned in space.

For **camera** inputs - RGB images from multiple views, we first use an image backbone (such as ResNet [26]) on each image to obtain its compact feature map. To transform 2D images into BEV space, LSS [47] is adopted: We first predict the discrete depth distribution of each pixel and scatter each pixel into discrete points along the camera ray, where the feature at each point is the product of its predicted depth and corresponding pixel feature. For each grid in BEV, we aggregate features from those points within the grid by Frustum Pooling[1]. In this way, we could aggregate images from an arbitrary number of cameras into one $C \times B_H \times B_W$ feature map, where C is the hidden dimension, B_H and B_W are the height and width of the BEV grid. Further, to introduce temporal cues, we aggregate the previous BEV of historical images by transforming it to current egocentric coordinate system according to the relative ego movement. Previous and current feature maps are thus spatially aligned and we could simply concatenate them to obtain the final BEV feature. Additionally, we found that (i) ground-truth supervision for the depth prediction module is important, which aligns with the finding in the object detection field [37]. (ii) When scattering the image features, it is beneficial to add a semantic segmentation module and scatter the predicted semantic scores as well. We conjecture that it increases the generalization ability of the end-to-end model by filtering out the unrelated texture information.[2]

For **LiDAR** inputs - point clouds, we employ the popular SECOND [70] which applies sparse 3D-convolution on the voxelized point clouds [79]. Its final output is also a BEV feature map with a size of $C \times B_H \times B_W$. To utilize temporal information, similar to existing works in object detection field [29, 62, 72], we concatenate aligned point clouds from multiple-frames with an additional channel to indicate the time-step.

When **fusing** the two BEV feature maps, we simply concatenate them into one and process it by a series of 2D convolutional layers. Since actions are the only direct supervision in end-to-end autonomous driving which are too sparse for the high-dimensional multi-sensor input, we provide extra feature-level supervision for the BEV feature map. Specifically, we use middle BEV feature maps from

[1] Please refer to [47] for details.

[2] We give ablations to support these claims in the experiment section.

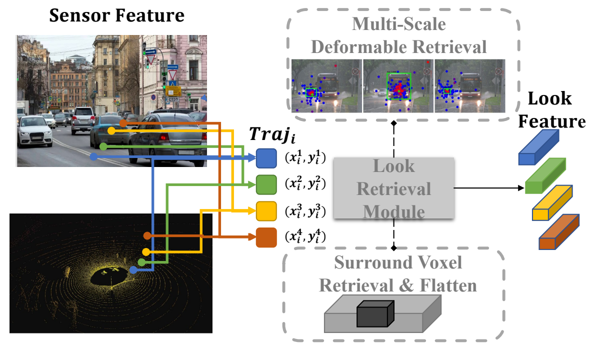

Figure 3. **Overall architecture of Look Module.**

Roach [77] as the target, an RL-based teacher network with privileged input, which takes rasterized BEV surrounding environment as privileged input and achieves decent performance with several convolutional layers. Note that any learnable expert model with a BEV feature representation could be adopted here such as [10, 53] and we adopt Roach here due to its robustness from RL training. By letting the middle BEV feature maps of the student network (*i.e.*, the encoder of ThinkTwice) be similar to the teacher network's, each BEV grid obtains dense supervision regarding decision-related information. In the experiment section, we empirically show that this supervision is necessary and is better than the commonly used BEV segmentation supervision signals in previous SOTA works [9, 27, 49].

3.2. Decoder

3.2.1 Coarse Prediction Module

Recall that from the encoder part, we have obtained a BEV feature map $\boldsymbol{H}_{\text{BEV}}$ with the shape $C \times B_H \times B_W$. We use 2D convolutional layers to downsample it and then flatten the small feature map into a 1D vector $\boldsymbol{H}_{\text{env}}$, which contains information about surrounding environments from cameras and LiDAR. For routing information including the target point, high-level command (go straight, turn left, turn left, *etc.*), and current speed, we use an MLP to encode them into a compact vector $\boldsymbol{H}_{\text{mst}}$, similar to [9, 68, 77]. With $\boldsymbol{H}_{\text{env}}$ and $\boldsymbol{H}_{\text{mst}}$ as input, we use another MLP to predict the ego vehicle's future action $\mathbf{Ctrl}_0$ and trajectories $\mathbf{Traj}_0$, where 0 denotes it is the initialization of the prediction. Note that this module follows the common practice in existing works: *flatten + MLP*, which ignores the spatial-temporal association between the prediction and current observation. In the following part, we propose *Look Module* and *Prediction Module* to extend the capacity of the decoder with the aforementioned prior knowledge and dense supervision.

3.2.2 Look Module

The intuition behind *Look Module* is that human drivers would check their target location to make sure there are no collisions with other agents and no violations of traffic rules before they actually go there, which is demonstrated to be effective in the trajectory prediction field [45]. Thus, with the predicted trajectory $\mathbf{Traj}_i$ from the last layer with the shape $T \times 2$ where T is the prediction horizon and 2 represents (x, y), we retrieve sensor features based on the coordinate of $\mathbf{Traj}_i$. The overall architecture is shown in Fig. 3.

For cameras, we project the coordinate back into the image plane with the cameras' extrinsics and intrinsics. Considering information on one pixel is limited and there might be errors during the projection, we adopt multi-scale deformable attention [80] to aggregate information:

$$\boldsymbol{H}_{i+1}^{\text{img-look}} = \text{DeformAttn}(\boldsymbol{H}_{\text{Img}}; \mathbf{Traj}_i; \boldsymbol{H}_{\text{env}}, \boldsymbol{H}_{\text{mst}}), \quad (1)$$

where $\boldsymbol{H}_{\text{Img}}$ is the multi-scale image feature maps; $\mathbf{Traj}_i$ serves as the reference point of the deformable attention; $\boldsymbol{H}_{\text{env}}$ and $\boldsymbol{H}_{\text{mst}}$ serve as the query of the attention.

For LiDAR, since it is already in the form of voxel features and the coordinate could be directly used, we simply retrieve the surrounding voxels of each coordinate in $\mathbf{Traj}_i$ and flatten them followed by an MLP to obtain $\boldsymbol{H}_{i+1}^{\text{lidar-look}}$. Finally, we concatenate $\boldsymbol{H}_{i+1}^{\text{img-look}}$ and $\boldsymbol{H}_{i+1}^{\text{lidar-look}}$, and use an MLP to get the look feature $\boldsymbol{H}_{i+1}^{\text{look}}$. H_{env} is also updated by $\boldsymbol{H}_{i+1}^{\text{look}}$ with another MLP.

By far, we reuse the representation power of the encoder and inject sample specific spatial prior, *i.e.*, intended location, into the features, which makes the model easier to be optimized and could lead to better generalization ability [5, 75].

3.2.3 Prediction Module

The intuition behind the *Prediction Module* is that human drivers would anticipate how surrounding agents would react to their action and check whether there would be collisions before they actually execute any action, *i.e.*, action-conditioned prediction [43]. Thus, to model the intended action-conditioned future of the scene, we use a spatial-GRU, which simply replaces linear layers in GRU [14] with 2D convolutional layers. It takes the current BEV feature as its initial states and at each time-step takes the predicted coarse action $\mathbf{Ctrl}_i^t$ and trajectories $\mathbf{Traj}_i^t$ from the last layer as input. We denote its output as $\boldsymbol{H}_{\text{predict}}^{i+1}$ with the shape $T \times B_H \times B_W \times C$ where T is the prediction horizon, B_H and B_W are the height and width of BEV grid, and C is the hidden dimension.

To provide supervision for $\boldsymbol{H}_{\text{predict}}^{i+1}$, we need to know the actual future scene when the ego vehicle takes the action $\mathbf{Ctrl}_i$, which is difficult during the open-loop imitation learning process. To deal with this issue, inspired by the teacher forcing [2] technique in NLP field, during training, we feed a set of extra inputs to the spatial GRU: the current BEV feature with the ground-truth action and trajectory $\mathbf{Ctrl}^{gt}$ and $\mathbf{Traj}^{gt}$. Denoting its corresponding output

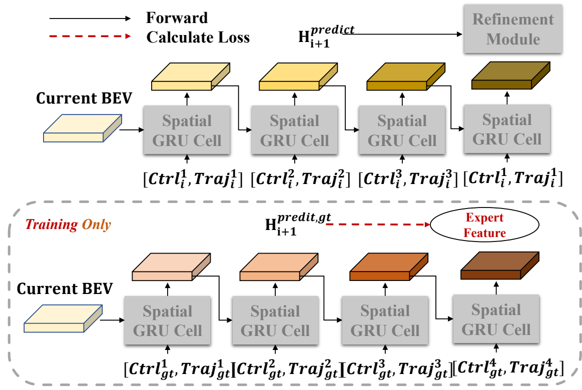

Figure 4. **Overall architecture of Prediction Module.**

as $\boldsymbol{H}_{i+1}^{\text{predict,gt}}$, we can supervise this hidden feature with the collected future scene. Here, we choose the Roach BEV feature as the target. Its overall structure is given in Fig. 4.

3.2.4 Refinement Module

With the *Look Feature* $\boldsymbol{H}_{i+1}^{\text{look}}$ and *Prediction Feature* $\boldsymbol{H}_{i+1}^{\text{predict}}$, the refinement module utilizes them to adjust the predicted action $\mathbf{Ctrl}_i$ and trajectories $\mathbf{Traj}_i$ from the last layer by:

$$\begin{aligned} \mathcal{O}_{i+1}^{\text{ctrl}}, \mathcal{O}_{i+1}^{\text{traj}} = \mathbf{MLP}([&\boldsymbol{H}_{i+1}^{\text{look}}; \boldsymbol{H}_{i+1}^{\text{predict}}; \mathbf{Ctrl}_i; \\ &\mathbf{Traj}_i; \boldsymbol{H}_{\text{env}}; \boldsymbol{H}_{\text{mst}}]), \end{aligned} \quad (2)$$

where $\mathcal{O}_{i+1}^{\text{ctrl}}$ and $\mathcal{O}_{i+1}^{\text{traj}}$ are the predicted offsets of action and trajectory between coarse prediction and ground-truth respectively. They are supervised as follows:

$$\begin{aligned} \mathcal{O}_{i+1}^{\text{ctrl}} &= \mathcal{L}(\mathbf{Ctrl}_i, \mathbf{Ctrl}^{gt}), \\ \mathcal{O}_{i+1}^{\text{traj}} &= \mathcal{L}(\mathbf{Traj}_i, \mathbf{Traj}^{gt}), \end{aligned} \quad (3)$$

where $\mathcal{L}$ represents the loss function which is the commonly used Smooth L1 loss. Finally, we update the predicted action and trajectory by:

$$\begin{aligned} \mathbf{Ctrl}_{i+1} &= \mathbf{Ctrl}_i + \mathcal{O}_{i+1}^{\text{ctrl}}, \\ \mathbf{Traj}_{i+1} &= \mathbf{Traj}_i + \mathcal{O}_{i+1}^{\text{Traj}}. \end{aligned} \quad (4)$$

By then, we finish one round of refinement. Similar to DETR-based methods [5, 75] in the general vision domain, the proposed decoder module could be stacked in cascade and we observe notable performance gain with multi-layers.

3.3. Supervision Signals

In end-to-end autonomous driving, since the direct supervision - the action (2 float number) or trajectories (T*2 float number) is rather sparse compared to the multi-modal multi-view inputs, to avoid overfitting and increase the generalization ability, it is a common practice in SOTA

Method	Encoder	Decoder	Modality	Extra Supervision	DS↑	RC↑	IS↑
CILRS [16]	ResNet + Flatten	MLP	C1	None	7.8±0.3	10.3±0.0	0.75±0.05
LBC [10]	ResNet + Flatten	MLP	C3	Expert	12.3±2.0	31.9±2.2	0.66±0.02
Transfuser [13]	Fusion via Transformer	GRU	C3L1	Dep+Seg+Map+Box	31.0±3.6	47.5±5.3	**0.77**±0.04
Roach [77]	ResNet + Flatten	MLP	C1	Expert	41.6±1.8	**96.4**±2.1	0.43±0.03
LAV [9]	PointPaiting	Multi-layer GRUs	C4L1	Expert+Seg+Map+Box	46.5±2.3	69.8±2.3	0.73±0.02
TCP [68]	ResNet + Flatten	GRU	C1	Expert	57.2±1.5	80.4±1.5	0.73±0.02
ThinkTwice	Geometric Fusion in BEV	Look-Predict-Refine	C4L1	Expert+Dep+Seg+Map	**65.0**±1.7	95.5±2.0	0.69±0.05
MILE*† [27]	ResNet + Flatten	GRU	C1	Map+Box	61.1±3.2	**97.4**±0.8	0.63±0.03
Interfuser* [57]	Fusion via Transformer	Transformer + GRU	C3L1	Map+Box	68.3±1.9	95.0±2.9	-
ThinkTwice*	Geometric Fusion in BEV	Look-Predict-Refine	C4L1	Expert+Dep+Seg+Map	**70.9**±3.4	95.5±2.6	**0.75**±0.05

Table 1. **Performance on Town05 Long benchmark.** ↑ means the higher the better. * denotes using extra data. † denotes no scenarios are involved in the evaluation, which is a much easier benchmark. For *Modality*, C denotes the camera sensor and L denotes the LiDAR sensor. *Extra Supervision* refers to labels required to train their student model besides actions and states of the ego vehicle. *Expert* denotes the distillation from privileged agents' outputs or features. *Seg* and *Depth* denotes the depth and semantic segmentation labels of the 2D images. *Box* denotes the bounding box of surrounding agents.

works [9,27,49,57,68] to apply multiple auxiliary supervisions. In ThinkTwice, we apply dense supervision on both the encoder and decoder and we summarize as follows:

Image Depth & Segmentation: Compared to LiDARs, images contain more semantics yet with lots of abundance [42]. and it is more difficult to be applied in autonomous driving due to the discrepancy between 2D images and the 3D world. To address this, we apply depth supervision for more accurate BEV projection, and project semantic feature maps to BEV along with pixel feature maps so that the influence of texture could be mitigated.

Feature Distillation: Existing works have shown that with privileged input (ground-truth agent location, lanes, traffic light states, *etc.*), a learning-based teacher network could achieve decent performance [53, 77]. Compared to BEV segmentation - an auxiliary task, distilling the teacher network's middle feature maps provides more direct supervision and is more related to the major task - driving [68, 77]. Here, we apply distillations on both the current BEV and the predicted future BEV (with teacher forcing).

Auxiliary Tasks: We use $\boldsymbol{H}_{\text{env}}$ to predict the ego vehicle's current speed which is helpful to mitigate the inertia problem [68, 77]. We use $\boldsymbol{H}_{\text{env}}$ and $\boldsymbol{H}_{\text{mst}}$ to estimate the value function of current states supervised by Roach - an RL teacher network. Similarly, for future scene representation generated by teacher forcing, we predict their speed and value as well.

Direct Supervision: Since we need coordinates to retrieve sensor features and actions to predict the conditioned future, our model outputs the ego vehicle's future action and trajectory together at each layer in a coarse-to-fine fashion.

To this end, the encoder receives dense supervision from depth, segmentation, and feature distillation, while the decoder receives dense supervision from multi-layer direct supervision and future feature map supervision in the prediction module.

4. Experiments

4.1. Benchmark

We use CARLA [21] as the simulator to conduct closed-loop autonomous driving evaluations. We conduct experiments on two widely used benchmarks, **Town05 Long** and **Longest6**. Each benchmark contains several routes and each route is defined by a sequence of sparse navigation points together with high-level commands (straight, turn left/right, lane changing, and lane following). The closed-loop driving task requires the autonomous agent to drive toward the destination point. It is designed to simulate realistic traffic situations and includes different challenging scenarios such as obstacle avoidance, crossing an unsignalized intersection, and sudden control loss.

4.2. Data Collection

ThinkTwice is an imitation learning framework that requires an expert to collect driving logs: a sequence of vehicle states and sensor data. Here, we use 4 cameras (front, left, right, back), one LiDAR, IMU, GPS, and speedometer. We adopt Roach [77], an RL-based network, as our expert similar to [27,68] due to the strong supervision it could provide. We collect data in 2 Hz on town01, town03, town04, and town06. We collect 189K frames in total which is similar to [9, 13, 68] to conduct most experiments and ablation studies. To match the size of the dataset and number of seen towns with concurrent works [57] (3M, 8 towns, 2Hz) and [27] (2.9M, 4 towns, 25Hz), we additionally collect a dataset of 2M [3] with all 8 towns where we would only use to compare with them and we denote this setting with *.

4.3. Metrics

We use the official metrics of CARLA Leaderboard: **Route Completion (RC)** is the percentage of the route completed by the autonomous agent. **Infraction Score (IS)**

[3] We stop collecting when our hard-disk is full.

Method	DS↑	RC↑	IS↑
WOR [8]	23.6	52.3	0.59
LAV [9]	34.2	73.5	0.53
Transfuser [13, 49]	56.7	**92.3**	0.62
ThinkTwice	61.3	73.0	0.81
ThinkTwice*	**66.7**	77.2	**0.84**

Table 2. **Performance on Longest6 benchmark.**

measures the number of infractions made along the route, with pedestrians, vehicles, road layouts, red lights, and *etc.*[4] **Driving Score (DS)** is the *main metric* which is the product of Route Completion and Infraction Score.

4.4. Comparsion with SOTA

We compare our work with state-of-the-art works in two competitive benchmarks of closed-loop evaluation. The results are shown in Tab. 1 and Tab. 2. We could observe that our model outperforms previous SOTA by a large margin on both benchmarks. Specifically, in Town5Long as shown in Tab. 1, ThinkTwice achieves the best DS under both protocols while Roach and MILE could run for a long time (highest RC) but have much more collision or violation of traffic rules. On other hand, Transfuser runs most safely (highest IS) but is too cautious to complete the route. As for the Longest6 benchmark as shown in Tab. 2, which is proposed by Transfuser, they could obtain a very high route completion score but our method achieves the best DS and IS which suggests a much safer driving process.

4.5. Component Analysis

In this section, we provide an empirical analysis of design choices among ThinkTwice. We use **Town05-Long** benchmark with 3 repeats to reduce the variance and we use 189K data of 4 towns, which means our model has never seen town05 during training.

Encoder Design: In Tab. 3, we give the performance of different encoder design choices in ThinkTwice. From **Model1**, we can find that simply adopting geometric fusion without any relevant supervision leads to poor performance, which aligns with the conclusion in [13]. **Model2** with depth and semantic segmentation tasks has slightly better results, which may come from the regularization effects of the two auxiliary tasks on the image features. In **Model3**, the explicit usage of depth and segmentation prediction during the projection process from image features to BEV features boosts the performance significantly, which demonstrates the importance of supervised geometric projection. In **Model4**, we further use two frames as inputs instead of one. Contrary to intuition, the improvement is very marginal considering the motion clue is introduced. It might be related to the inertia/copycat problem [13, 66] where the model learns to cheat by simply copying the movement between the previous and current frame. It could lead to degenerated performance during closed-loop evaluation. However, since the *Prediction Module* in the decoder requires the motion clues of surrounding agents, we keep the input as 2 frames in all the following experiments. In **Model5**, we replace the Expert Feature Distillation to the BEV feature with the BEV segmentation task as in [9, 13, 27]. We can observe a performance drop which might be due to the fact that the BEV segmentation task could only serve as an implicit regularization while the expert feature contains decision-related information.

[4]Please refer to https://leaderboard.carla.org/get_started_v1/ for details.

ID	Method	DS↑	RC↑	IS↑
1	Baseline	43.0±3.9	74.2±3.2	0.56±0.05
2	+Segmentation & Depth Task	45.0±2.2	73.2±4.0	0.60±0.04
3	+Segmentation & Depth Projection	57.8±1.5	78.8±3.3	**0.74**±0.03
4	+Two Frames	**59.3**±1.4	**80.2**±3.6	**0.74**±0.04
5	Expert Feature -> BEV Segmentation	50.5±2.3	73.4±3.2	0.70±0.03

Table 3. **Ablation for Encoder Design.** Baseline means original LSS [47] for camera inputs and SECOND [70] for LiDAR input with a TCP [68] head. Model2 adds the depth and segmentation auxiliary for images. Model3 additionally uses depth and segmentation during the projection to BEV. Model4 uses 2 frames as input. Model5 changes the BEV supervision from BEV features of the expert to BEV segmentation task.

ID	Method	DS↑	RC↑	IS↑
4	TCP-Head	59.3±1.4	80.2±3.6	**0.74**±0.04
5	+1 Decoder	61.6±1.4	90.8±2.2	0.68±0.04
6	+5 Decoders	**65.0**±1.7	95.5±2.0	0.69±0.05
7	w/o Look	62.4±1.8	93.5±2.1	0.67±0.08
8	w/o Predict	61.7±1.9	96.2±3.0	0.63±0.03
9	w/o TF	62.0±2.2	**97.1**±2.9	0.62±0.04

Table 4. **Ablation for Decoder Design.** The baseline is Model4 in the encoder. Model5 and Model6 add 1 and 5 proposed decoder modules respectively. Model7 removes the Look Module from Model6 while Model8 removes the Prediction Module from Model6. Model9 removes the teacher forcing technique.

Due to Model4's superior performance, we adopt it as the encoder of ThinkTwice.

Decoder Design: Based on **Model4** in Tab. 3, we conduct component analysis for the decoder and the results are in Tab. 4. In **Model5**, we could observe performance improvement with one additional decoder layer. Specifically, RC improves a lot which indicates less stuck while IS drops a little bit which is natural since it has a larger possibility to have any events during the much longer driving process. In **Model6**, with 5 stacked decoder layers, the results are significantly boosted, which demonstrates the effectiveness of the proposed decoder paradigm and its strong scalability. Until now, we obtain ThinkTwice's final model - Model6 and we further conduct ablation studies with it. In **Model7**, the removal of Look Modules causes an explicit performance drop. In **Model8**, without the Prediction Module, it has a much lower DS, slightly higher RC, and much lower IS, which indicates a more reckless agent who tends to simply drive forward and ignore the environment. It makes sense since it lacks the ability to know the ramification of its decision with the Prediction Module. In **Model9**, we verify the effectiveness of the teacher-forcing technique. Without it, the model exhibits similar behaviors with Model8, *i.e.*, removing the prediction module. It is in line with expectations since there is no extra information injected if we use the Prediction Module without any supervision, which has no significant difference with an enlarged MLP.

In conclusion, we verify the claims and approaches proposed in Sec. 3. We found that it is essential to add prior knowledge to the encoder and decoder. We also demonstrate the effectiveness of stacking decoder layers with dense supervision, which leads to our state-of-the-art performance.

4.6. Discussion about Enlarging Model Capacity

In this section, we aim to investigate different ways of enlarging the capacity of an end-to-end autonomous driving model and compare them in a fair setting.

Enlarge Encoder: One natural idea is to enlarge the size of the encoder, which works perfectly well and demonstrates strong scalability in both natural language processing [20] and computer vision [26] fields. However, it does not apply in the end-to-end autonomous model, which has been observed in the community. In our early exploration experiments, we have conducted experiments with TCP [68], a most recent SOTA method with single image input and it uses a single ResNet-34 as the encoder. It provides a single-variable environment to observe the effects of enlarging the encoder size. The results are in Tab. 5. We could find that enlarging the original TCP from ResNet-34 to ResNet-101 causes a significant performance drop. We conjecture the reason why simply enlarging the encoder does not work is that in end-to-end autonomous driving, the encoder is only responsible for processing the multi-sensor input while the decoder is responsible for finding out decision-related information. Large encoders could lead to better scene representation feature but it does not contribute much to the decision process. Actually, it is the major motivation of ThinkTwice: enlarging the capacity of the decoder in a proper way.

Encoder	DS↑	RC↑	IS↑
ResNet-18	48.7±1.7	**81.3**±2.2	0.58±0.04
ResNet-34	**57.2**±1.5	80.4±1.5	**0.73**±0.02
ResNet-101	38.0±4.5	79.9±3.7	0.49±0.02

Table 5. **Performance of TCP [68] under different encoder sizes.**

Method	#Param	DS↑	RC↑	IS↑
Baseline	78.2M	59.3±1.4	80.2±3.6	**0.74**±0.04
Ours	120.2M	**65.0**±1.7	**95.5**±2.0	0.69±0.05
MLP/GRU	123.3M	59.4±2.4	81.2±3.0	0.72±0.04
Backbone	119.3M	58.4±3.1	80.0±3.6	0.73±0.06

Table 6. **Performance of ThinkTwice with different major parts under the same number of parameters.** *Baseline* uses the Model4 as the encoder and TCP head as the decoder. *Ours* uses 5 stacked proposed decoder layers with supervision. *MLP/GRU* uses the TCP head with 2x depth and 4x width. *Backbone* uses ResNet-152 as the backbone instead of ResNet-50.

Enlarge MLP/GRU: Besides enlarging the encoder size and the proposed stacking decoders, another choice to increase the models' capacity is to increase the width/depth of the classical MLP/GRU decoder. Here, we take Model4 as the baseline which includes the proposed encoder and a TCP head and we enlarge the model in the three aforementioned ways. The results are in Tab. 6. We can observe that simply increasing the depth/width of the encoder or the decoder would not bring performance gain. On the contrary, ThinkTwice enlarges the decoder's capacity in a coarse-to-fine fashion with dense supervision and spatial-temporal knowledge, which injects strong priors into the model and thus leads to better performance.

5. Conclusion

In this work, we present ThinkTwice, an end-to-end autonomous driving paradigm that emphasizes enlarging the capacity of the decoder. We propose a scalable decoder layer with dense supervision and spatial-temporal priors. By stacking the proposed decoder layer, we achieve state-of-the-art performance on two competitive closed-loop autonomous driving benchmarks.We hope the attempts and successful parts illustrated in this study could provide useful information for this line of study in the community.

Acknowledgements. We are grateful to Mingjie Pan for technical assistance. We also thank the reviewers for their constructive comments and suggestions. This work was in part supported by NSFC (62206172, 62222607), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102), and Shanghai Committee of Science and Technology (21DZ1100100).

References

[1] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018. 3

[2] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015. 2, 5

[3] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs A. Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. 2

[4] Yigit Baran Can, Alexander Liniger, Danda Pani Paudel, and Luc Van Gool. Structured bird's-eye-view traffic scene understanding from onboard images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15661–15670, 2021. 3

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 5

[6] Sergio Casas, Abbas Sadat, and Raquel Urtasun. Mp3: A unified model to map, perceive, predict and plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14403–14412, 2021. 3

[7] Raphael Chekroun, Marin Toromanoff, Sascha Hornauer, and Fabien Moutarde. Gri: General reinforced imitation and its application to vision-based autonomous driving. *arXiv preprint arXiv:2111.08575*, 2021. 3

[8] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. Learning to drive from a world on rails. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021. 7

[9] Dian Chen and Philipp Krähenbühl. Learning from all vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3, 4, 6, 7

[10] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR, 2020. 2, 3, 4, 6

[11] Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, et al. Persformer: 3d lane detection via perspective transformer and the openlane benchmark. In *European Conference on Computer Vision*, pages 550–567. Springer, 2022. 3

[12] Kashyap Chitta, Aditya Prakash, and Andreas Geiger. Neat: Neural attention fields for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15793–15803, 2021. 2, 3

[13] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *Pattern Analysis and Machine Intelligence (PAMI)*, 2022. 6, 7

[14] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, 2014. 5

[15] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE international conference on robotics and automation*, pages 4693–4700. IEEE, 2018. 2

[16] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338, 2019. 2, 6

[17] Alexander Cui, Sergio Casas, Abbas Sadat, Renjie Liao, and Raquel Urtasun. Lookout: Diverse multi-future prediction and planning for self-driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16107–16116, 2021. 3

[18] Jiajun Deng, Shaoshuai Shi, Peiwei Li, Wengang Zhou, Yanyong Zhang, and Houqiang Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 3

[19] Zijun Deng, Xiaowei Hu, Lei Zhu, Xuemiao Xu, Jing Qin, Guoqiang Han, and Pheng-Ann Heng. R^3Net: Recurrent residual refinement network for saliency detection. In *IJCAI*, 2018. 3

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 8

[21] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017. 2, 6

[22] Philipp Fischer, Alexey Dosovitskiy, Eddy Ilg, Philip Häusser, Caner Hazırbaş, Vladimir Golkov, Patrick Van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. *arXiv preprint arXiv:1504.06852*, 2015. 3

[23] Noa Garnett, Rafi Cohen, Tomer Pe'er, Roee Lahav, and Dan Levi. 3d-lanenet: end-to-end 3d multiple lane detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2921–2930, 2019. 3

[24] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015. 3

[25] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2014. 3

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 4, 8

[27] Anthony Hu, Gianluca Corrado, Nicolas Griffiths, Zak Murez, Corina Gurau, Hudson Yeo, Alex Kendall, Roberto

Cipolla, and Jamie Shotton. Model-based imitation learning for urban driving. *arXiv preprint arXiv:2210.07729*, 2022. 2, 3, 4, 6, 7

[28] Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *European Conference on Computer Vision*, pages 533–549. Springer, 2022. 3

[29] Yihan Hu, Zhuangzhuang Ding, Runzhou Ge, Wenxin Shao, Li Huang, Kun Li, and Qiang Liu. Afdetv2: Rethinking the necessity of the second stage for object detection from point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022. 4

[30] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 3

[31] Junjie Huang, Guan Huang, Zheng Zhu, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv preprint arXiv:2112.11790*, 2021. 3

[32] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2462–2470, 2017. 3

[33] Xiaosong Jia, Li Chen, Penghao Wu, Jia Zeng, Junchi Yan, Hongyang Li, and Yu Qiao. Towards capturing the temporal dynamics for trajectory prediction: a coarse-to-fine approach. In *6th Annual Conference on Robot Learning*, 2022. 3

[34] Xiaosong Jia, Liting Sun, Masayoshi Tomizuka, and Wei Zhan. Ide-net: Interactive driving event and pattern extraction from human data. *IEEE Robotics and Automation Letters*, 6(2):3065–3072, 2021. 3

[35] Xiaosong Jia, Liting Sun, Hang Zhao, Masayoshi Tomizuka, and Wei Zhan. Multi-agent trajectory prediction by combining egocentric and allocentric views. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1434–1443. PMLR, 08–11 Nov 2022. 3

[36] Hongyang Li, Chonghao Sima, Jifeng Dai, Wenhai Wang, Lewei Lu, Huijie Wang, Enze Xie, Zhiqi Li, Hanming Deng, Hao Tian, et al. Delving into the devils of bird's-eye-view perception: A review, evaluation and recipe. *arXiv preprint arXiv:2209.05324*, 2022. 3

[37] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *arXiv preprint arXiv:2206.10092*, 2022. 3, 4

[38] Zhichao Li, Feng Wang, and Naiyan Wang. Lidar r-cnn: An efficient and universal 3d object detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7546–7555, 2021. 3

[39] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *European Conference on Computer Vision*, 2022. 3

[40] N. Liu and J. Han. Dhsnet: Deep hierarchical saliency network for salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 3

[41] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. *arXiv preprint arXiv:2203.05625*, 2022. 3

[42] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. *arXiv preprint arXiv:2205.13542*, 2022. 6

[43] Srikanth Malla, Behzad Dariush, and Chiho Choi. Titan: Future forecast using action priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11186–11196, 2020. 5

[44] Karttikeya Mangalam, Yang An, Harshayu Girase, and Jitendra Malik. From goals, waypoints & paths to long term human trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15233–15242, 2021. 3

[45] Francesco Marchetti, Federico Becattini, Lorenzo Seidenari, and Alberto Del Bimbo. Mantra: Memory augmented networks for multiple trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7143–7152, 2020. 5

[46] Urs Muller, Jan Ben, Eric Cosatto, Beat Flepp, and Yann Cun. Off-road obstacle avoidance through end-to-end learning. *Advances in neural information processing systems*, 18, 2005. 2

[47] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *European Conference on Computer Vision*, pages 194–210. Springer, 2020. 3, 4, 7

[48] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988. 2

[49] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 2, 3, 4, 6, 7

[50] Xuebin Qin, Zichen Zhang, Chenyang Huang, Chao Gao, Masood Dehghan, and Martin Jagersand. Basnet: Boundary-aware salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 3

[51] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8555–8564, 2021. 3

[52] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015. 3

[53] Katrin Renz, Kashyap Chitta, Otniel-Bogdan Mercea, A. Sophia Koepke, Zeynep Akata, and Andreas Geiger. Plant: Explainable planning transformers via object-level representations. In *Conference on Robotic Learning*, 2022. 4, 6

[54] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*, 2018. 3

[55] Abbas Sadat, Sergio Casas, Mengye Ren, Xinyu Wu, Pranaab Dhawan, and Raquel Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *European Conference on Computer Vision*, pages 414–430. Springer, 2020. 3

[56] Axel Sauer, Nikolay Savinov, and Andreas Geiger. Conditional affordance learning for driving in urban environments. In *Conference on Robot Learning*, pages 237–252. PMLR, 2018. 2

[57] Hao Shao, Letian Wang, RuoBing Chen, Hongsheng Li, and Yu Liu. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. *arXiv preprint arXiv:2207.14024*, 2022. 2, 3, 6

[58] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020. 3

[59] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Motion transformer with global intention localization and local movement refinement. *arXiv preprint arXiv:2209.13508*, 2022. 3

[60] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 3

[61] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018. 3

[62] Pei Sun, Weiyue Wang, Yuning Chai, Gamaleldin Elsayed, Alex Bewley, Xiao Zhang, Cristian Sminchisescu, and Dragomir Anguelov. Rsn: Range sparse net for efficient, accurate lidar 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5725–5734, 2021. 4

[63] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision*, pages 402–419. Springer, 2020. 3

[64] Marin Toromanoff, Emilie Wirbel, and Fabien 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*, pages 7153–7162, 2020. 3

[65] Tiantian Wang, Ali Borji, Lihe Zhang, Pingping Zhang, and Huchuan Lu. A stagewise refinement model for detecting salient objects in images. In *The IEEE International Conference on Computer Vision*, 2017. 3

[66] Chuan Wen, Jierui Lin, Trevor Darrell, Dinesh Jayaraman, and Yang Gao. Fighting copycat agents in behavioral cloning from observation histories. *Advances in Neural Information Processing Systems*, 33:2564–2575, 2020. 7

[67] Penghao Wu, Li Chen, Hongyang Li, Xiaosong Jia, Junchi Yan, and Yu Qiao. Policy pre-training for end-to-end autonomous driving via self-supervised geometric modeling. *International Conference on Learning Representations*, 2023. 2

[68] Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. *Advances in Neural Information Processing Systems*, 2022. 2, 3, 4, 6, 7, 8

[69] Enze Xie, Zhiding Yu, Daquan Zhou, Jonah Philion, Anima Anandkumar, Sanja Fidler, Ping Luo, and Jose M Alvarez. Mˆ 2bev: Multi-camera joint 3d detection and segmentation with unified birds-eye view representation. *arXiv preprint arXiv:2204.05088*, 2022. 3

[70] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 4, 7

[71] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Std: Sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1951–1960, 2019. 3

[72] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11784–11793, 2021. 3, 4

[73] Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun. End-to-end interpretable neural motion planner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8660–8669, 2019. 3

[74] Wenyuan Zeng, Shenlong Wang, Renjie Liao, Yun Chen, Bin Yang, and Raquel Urtasun. Dsdnet: Deep structured self-driving network. In *European Conference on Computer Vision*, pages 156–172. Springer, 2020. 3

[75] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022. 5

[76] Qingwen Zhang, Mingkai Tang, Ruoyu Geng, Feiyi Chen, Ren Xin, and Lujia Wang. Mmfn: Multi-modal-fusion-net for end-to-end driving. *arXiv preprint arXiv:2207.00186*, 2022. 3

[77] Zhejun Zhang, Alexander Liniger, Dengxin Dai, Fisher Yu, and Luc 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. 2, 3, 4, 6

[78] Brady Zhou and Philipp Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13760–13769, 2022. 3

[79] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings*

of the IEEE Conference on Computer Vision and Pattern Recognition, pages 4490–4499, 2018. 4

[80] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 5