PPAD: Iterative Interactions of Prediction and Planning for End-to-end Autonomous Driving

Zhili Chen[1†], Maosheng Ye[1], Shuangjie Xu[1], Tongyi Cao[2], and Qifeng Chen[1✉]

[1]HKUST [2]DeepRoute.AI

{zchenei, myeag, shuangjie.xu}@connect.ust.hk, tongyicao@deeproute.ai, cqf@cse.ust.hk

Abstract. We present a new interaction mechanism of prediction and planning for end-to-end autonomous driving, called PPAD (Iterative Interaction of **P**rediction and **P**lanning **A**utonomous **D**riving), which considers the timestep-wise interaction to better integrate prediction and planning. An ego vehicle performs motion planning at each timestep based on the trajectory prediction of surrounding agents (e.g., vehicles and pedestrians) and its local road conditions. Unlike existing end-to-end autonomous driving frameworks, PPAD models the interactions among ego, agents, and the dynamic environment in an autoregressive manner by interleaving the **Prediction** and **Planning** processes at every timestep, instead of a single sequential process of prediction followed by planning. Specifically, we design ego-to-agent, ego-to-map, and ego-to-BEV interaction mechanisms with hierarchical dynamic key objects attention to better model the interactions. The experiments on the nuScenes benchmark show that our approach outperforms state-of-the-art methods. Project page at https://github.com/zlichen/PPAD.

Keywords: End-to-end Autonomous Driving

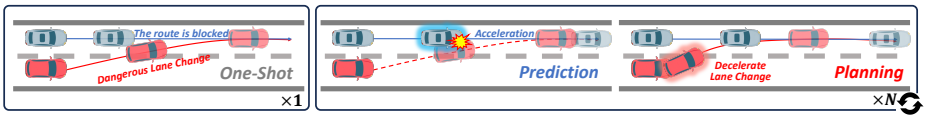
1 Introduction

The blossom of deep learning techniques has empowered autonomous driving, where many exciting milestones in autonomous driving have burst into our eyes owing to the convenient and interpretable discrete module designs. Recently, the planning-oriented [19] philosophy resonated with the community for pursuing a more effective end-to-end driving system, which is the focus of this work.

Traditional methods in an autonomous driving system often break down the system into modular components, including localization, perception, tracking, prediction, planning, and control for interpretability and visibility. However, there are several drawbacks: 1) the accumulation of errors between modules becomes more significant as the system complexity increases. 2) the performance of

[†]Work done during an internship at DeepRoute.AI.

[✉]Corresponding author.



(a) Traditional one-shot approach. **(b) Our proposed PPAD framework.**

Fig. 1: A high-level illustration of our proposed PPAD framework. The agent (in blue) intends to drive straight, while the ego (in red) plans to change lanes. Fig. 1(a) presents the typical one-shot method that might result in invalid motion plans and lead to an accident because of a lack of in-depth interactions. Fig. 1(b) demonstrates the gaming process between the ego and the agent under the PPAD architecture. During the prediction process, the agent executes an assertive plan by accelerating to stop the ego from blocking its route. The planning process of the ego plans trajectory based on the previous prediction process of the agent. The ego decelerates to avoid a potential accident and then changes lanes to achieve its driving goal.

the downstream task is highly related to the upper stream module, which makes it very difficult to construct a unified data-driven infrastructure.

Recently, end-to-end autonomous driving has gained popularity due to its simplicity. Two main lines are proposed based on the learning architecture. The first kind of method [10] takes the raw sensor data as input and directly outputs the planning trajectories or control command without any view transformation as intermediate representations for scene understanding. The other kinds of approaches [19, 23] are built upon BEV representation and fully utilize the queries to generate the intermediate outputs as guidance for producing the planning results. One of the most significant advantages lies in the interpretability. In this work, we follow the design of the second kind of work.

VAD [23] and UniAD [19] are typical one-shot motion planning methods, which only consider a single-step interaction between agents, ego-agent and surrounding environment (e.g., map elements). ThinkTwice [22] makes it a two-stage framework to enhance the gaming or interaction procedure. QCNet [56] and GameFormer [21] also recurrently model the trajectory prediction task. Given motion planning is a computational problem that finds a sequence of valid trajectories, often based on surrounding agents' forecasting, environmental understanding, and historical and future contexts. It can also be viewed as a game in which agents continuously plan their next move according to other agents' intentions and the encountering environment, further achieving their ultimate goals through incremental actions. To model these dynamic interactions of prediction and planning in end-to-end autonomous driving, it is crucial to consider the possible variance of predicted trajectories through multi-step modeling for planning feasible trajectories.

Inspired by VAD [23], we aim to introduce the step-by-step Prediction-Planning into a learning-based framework. Intuitively, the prediction and planning modules can be modeled as a motion forecasting task, which predicts future waypoints by the given historical information. The results of prediction and planning modules at each time step are highly dependent on each other. Therefore,

we need to consider the agent-agent and agent-environment interactions iteratively and bidirectionally to maximize the expectation of agents' prediction under the given observation of the other agents. We propose our **PPAD** to plan the ego agent's future trajectories step-by-step to model the timestep-wise bidirectional interaction or gaming in a vectorized learning framework as shown in Fig. 1. PPAD consists of the prediction and planning process. For each motion forecasting step, 1) **Prediction** process generates current step motion states by cross-attention and self-attention among agents and environment based on previous motion states to model the fine-grained bidirectional interactions. We take ego-agent-environment-BEV interaction into account to propagate features among all the traffic participants. 2) **Planning** process predicts the current step motion trajectories based on the expectation process. Our contributions are summarized as follows:

- We propose PPAD that optimizes ego-agent-environment interactions in an iterative prediction-planning manner. Iterative optimization could model the interactions and gaming better and more naturally in a planning task. The prediction process deals with more fine-grained and complex future uncertainties for multi-agent context learning, while the planning process plans a one-step future trajectory for the ego vehicle.
- We model fine-grained interactions among the ego vehicle, agents, environment, and BEV features map, step-by-step with hierarchical dynamic key objects attention emphasizing on the spatial locality.
- The experiments conducted on the nuScenes [3] and Argoverse [6,44] datasets have demonstrated the effectiveness of our approach over state-of-the-art approaches.

2 Related Work

2.1 Multi-stage Autonomous Driving

Most autonomous driving systems are built upon the multi-stage design philosophy, which commonly consists of localization, perception, and planning. The perception module has been well studied recently due to the emergence of deep learning. Camera-based [20, 28, 29], Lidar-based [26, 48–50, 55] or fused-based [36, 46] approaches are proposed to fully exploit the potential of raw sensor data in order to produce accurate 3D objects prediction, semantic segmentation or tracking velocity. Prediction takes the outputs of the perception module to generate the future waypoints for the ego and the agents. Current approaches [1, 5, 11–13, 16, 30, 32, 40, 42, 48, 52, 56, 57] explore different representations to encode surrounding environment (map information) and agent interactions to predict final trajectories by regression or postprocessing sampling strategies. Some other works [4, 37] propose a joint perception and prediction framework, which aggregates historical information to generate tracklets with future trajectories. This unified learning framework could help address the non-differential process and alleviate the unstable perception problem, compared

with previous works. Based on the perception and prediction results, planning module [1,7,12,39] plan its future behavior by cost-map optimization or learning-based approaches.

2.2 End-To-End Autonomous Driving

Recently, more and more works have focused on end-to-end autonomous driving due to its merits in reducing internal accumulative errors and direct yet simple learning objectives. Typical methods [2, 45] take the raw visual inputs to regress the final control command or trajectory points without any view transformations. To embrace reinforcement learning, a series of following works [8, 45, 54] utilize policy-based or valued-based approaches to improve driving behavior. With the popularity of BEV representation, more advanced architectures [9, 22] are introduced to attach more interpretability and help deal with the complex interactions in the driving scenarios. Another merit of BEV representation lies in its simplicity in fusing multi-modality sensors. Moreover, the modularized approaches [14, 18, 19, 23] decouple the end-to-end learning-based methods into several submodules or subtasks, while in a multi-task learning manner. The unified design could propagate and share learning context between modules through queries or feature maps.

2.3 BEV Representation

BEV representation has gained significant prominence in the field of autonomous driving systems due to its inherent distortion-free characteristics and its simplicity in facilitating multi-sensor fusion. There are two main lines for BEV representation, including bottom-up and top-down ways. LSS [38] stands out as a bottom-up pioneering work that explores depth distribution for the 3D space frustum sampling to form BEV representation. Works [20, 28] optimize the pipeline through better depth estimation or lightweight sampling design. BEVFormer [29] and its following works [20, 33–35, 43, 47] adopt the top-down architecture, which uses a deformable transformer for the view transformations without depth supervision.

3 Method

3.1 Framework Overview

We present the overall framework, PPAD, in Fig. 2, which comprises the principal modules of the Perception Transformer and our proposed Iterative Prediction-Planning Module. The Perception Transformer encodes the scene contexts into the BEV features map and further decodes as vectorized agents and map representations. The Iterative Prediction-Planning module consists of *Prediction* and *Planning* process in general. It dissects the dynamic interactions between the ego vehicle and the agents along the temporal dimension. Eventually, it predicts the motions of the agents and plans the future trajectory for the ego vehicle.

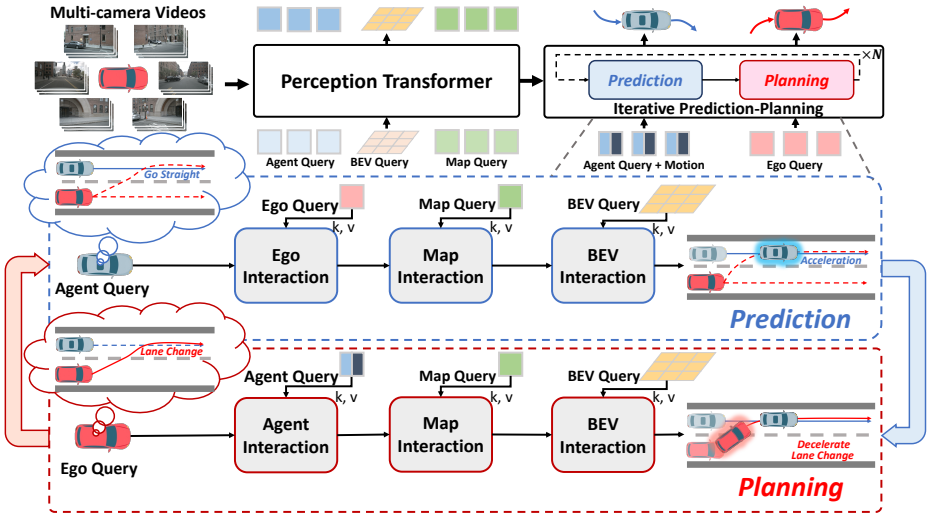


Fig. 2: Overall architecture of our proposed self-driving framework, PPAD. It consists of the Perception Transformer and the Iterative Prediction-Planning Module. The Perception Transformer encodes scene contexts into agent queries, map queries, and BEV queries. Then, the Prediction-Planning Module interleaves the processes of the agent motion prediction and the ego planning for N times. Throughout the iterative Prediction and Planning processes, in-depth interactions are conducted among the ego, agents, map elements, and BEV features. In the Prediction process, the agent initially intends to go straight and is unaware of the potential motion of the ego. After interacting with the ego, map elements, and BEV features, the agent plans to be assertive and proceeds to accelerate. In the following Planning process, the ego knows the agent will accelerate through interacting with the updated agent query. It eventually plans to decelerate first and then conduct the lane change for safety reasons.

Image Features Module uses a shared image backbone network (e.g., ResNet [15]) to extract image features for separate camera views.

BEV Features Module transform the semantic features from the multi-view cameras into a united bird's-eye-view. Specifically, we inherit the encoder from BEVFormer [29, 47] to construct the BEV features. The grid-shape learnable BEV queries $\mathbf{B} \in \mathbb{R}^{H \times W \times C}$ are randomly initialized and learned to interact with the multi-view image features through deformable attention [58] to conduct spatial modeling. Temporal modeling is conducted in a recurrent manner, which applies the deformable attention between the current frame's BEV queries and the one from the previous time step.

Vectorized Features Module Inspired by the VAD [23] paradigm, we also encode the scene contexts into vectorized representations through a detection decoder head [29, 58] and a map element decoding head [31], resulting in N_A of learned agent queries $\mathbf{A} \in \mathbb{R}^{N_A \times C}$ and N_M of learned map queries $\mathbf{M} \in \mathbb{R}^{N_M \times C}$. Separate MLP-based decoders will be attached to produce side output, which takes the learned queries as inputs and predicts with the agent attributes (lo-

cations, dimensions, classes, etc.) or map attributes (classes and map vectors described by points). Additionally, the agent queries will be combined with the learnable motion embeddings for modeling the diverse motions of the agents. The agents with motions are represented as $\mathbf{A} \in \mathbb{R}^{N_A \times N_A^{mot} \times C}$. Similarly, the ego vehicle is modeled with three modes, representing the high-level driving commands of *going straight, turn left, and turn right*, in the form of $\mathbf{E} \in \mathbb{R}^{N_E \times N_E^{mot} \times C}$.

Iterative Prediction-Planning Module predicts the future trajectories for the ego vehicle and the agents in an interleaved fashion. Different from the traditional practice that predicts all the trajectories in one go, our PPAD framework articulates each step of motion planning by iterating the agent motion prediction and the ego planning processes. Thanks to the PPAD framework, we can conduct in-depth design to enforce key objects interactions (in Sec. 3.3) on scene contexts in a coarse-to-fine manner. We further improve the driving performance for the ego vehicle by taking the noisy trajectory as each step prediction and training the PPAD framework to reconstruct its original position at the following time step (in Sec. 3.4).

3.2 Iterative Interactions of Prediction and Planning

In the real world, the driving traffic changes constantly. Drivers plan and execute their decisions by ceaselessly reasoning the relationships among traffic participants in the scene. The planning task requires the self-driving system to have a good understanding of the scene and be capable of resolving the spatial-temporal causal factors. Therefore, we innovate the PPAD to dissect the planning task into multi-steps of the agent prediction and the ego planning processes and eventually promote consensuses among the ego's and the agents' future trajectories. The PPAD framework embodies the traffic interactions as gaming along space-time, producing a more accurate planning trajectory for the ego vehicle.

Specifically, the ego and agents inherit the same philosophy of alternatively optimizing their motion behavior based on each other's motion forecasting at each future time step. In the following section, we will demonstrate the agent prediction process and elaborate on the details of the ego planning process.

Prediction Process As illustrated in Fig 2, the agent predicts its subsequent step motion during the Prediction process, conditioned on the output of the ego vehicle's outcome from the previous Planning process. Specifically, the initial state of the agent query comprises its driving intention. It will then interact with the ego query updated from the previous planning process, which indicates the latest driving plan of the ego vehicle. After that, it will interact with map elements to choose the driving paths. At last, it gathers detailed geometric information by interacting with the BEV features and comes up with its precise next-step movement.

Planning Process We consider the period of history with T_{obs} steps and the future with T_{fut} steps. The future trajectory of ego is denoted as $\{p_E^t\}^{t \in T_{fut}}$.

For each agent $a \in \mathbf{A}$, the trajectory is represented as $\{p_a^t\}^{t \in T_{fut}}$. The positions of the detected map elements are denoted as $\{p_m\}_{m \in \mathbf{M}}$. We define the operator of $\mathcal{M}(p_E^t, p_\mathbf{A}/p_\mathbf{M}, s)$ to mask out the agents or map elements that are beyond a distance of s towards the ego located at p_E^t to conduct key objects attention, and we will discuss the algorithm details in Sec. 3.3.

Agent Interaction The resulting agents' motions from the prediction process are located at $\{p_a^{t+1}\}_{a \in \mathbf{A}}$, comprising the motion states up to the time step of $t+1$. Moving the ego from p_E^t to the future one-step of p_E^{t+1}, the ego vehicle should consider the agents' traffic globally and locally. From a more global perspective, the agents at a larger range provide more extensive information on traffic flow, which is essential in long-term trajectory planning. Regarding the spatial locality, the nearby agents are recognized as the key agents, which are supposed to be vitally related to the ego's driving decision.

Therefore, we propose to conduct a hierarchical interaction with the agents through the attention mechanism to learn coarse-to-fine traffic context features for the ego. Centered at the ego's space-time position p_E^t, we initially define a distance set of $\mathbf{S}$ of $\{+\infty\ m, 15\ m, 7.5\ m\}$ which covers the coarse-to-fine perception ranges. We formed the multi-scale agent sets by applying $\mathcal{M}(p_E^t, p_\mathbf{A}, s)$ with different ranges. Then, the ego query interacts with the agents hierarchically through multi-head cross-attention, `MHCA`. We take the sum of the learned hierarchical attention results as the final values:

$$\mathbf{E}^k = \sum_{s \in \mathbf{S}} \texttt{MHCA}(\mathbf{E}, \mathbf{A}^k, \mathcal{M}(p_\mathbf{E}^t, p_{\mathbf{A}^k}^{t+1}, s)), k \in [1, N_A^{mot}], \tag{1}$$

where the ego independently queries information from different modes $k \in N_A^{mot}$ of agents, and then we stack the results output from different modes. Further, we apply the set operations to condense the features:

$$\mathbf{E}' = \texttt{MAX}([\mathbf{E}^1, .., \mathbf{E}^{N_A^{mot}}]) + \texttt{MEAN}([\mathbf{E}^1, .., \mathbf{E}^{N_A^{mot}}]), \tag{2}$$

where the `MAX` and `MEAN` are applied to aggregate features along the agents' mode dimension and output with the updated ego query $\mathbf{E}'$.

Map Interaction Existing works [19, 23] tried to summarize all the required map information for planning by simply applying the global-level interaction once. They overlook the complexity of the evolving motion dynamic and overrate that the ego can plan precisely in the longer term by a single interaction with the map information.

With our proposed PPAD framework, we can enrich the ego-map interaction by considering the ego's local road conditions based on its latest position. This results in better identifying the useful map information for each step of planning. The ego query interacts with the map queries in a similar practice as the interaction with the agents. The difference is that the map instances are not movable in the future time steps. The local and global map information can be abstracted into the ego query by `MHCA`:

$$\mathbf{E}'' = \sum_{s \in \mathbf{S}} \texttt{MHCA}(\mathbf{E}', \mathbf{M}, \mathcal{M}(p_\mathbf{E}^t, p_m, s)), \tag{3}$$

where $\mathbf{E}''$ is the ego query updated with the critical map information for the next step of planning.

BEV Interaction BEV is the fundamental representation of the whole system, which is the abstraction of the multi-camera features. Beyond the vectorized representation, there is other non-structural environmental information, including roads and fences. UniAD [19] models these non-structural pieces of stuff with an occupancy grid map. There are several drawbacks in UniAD: 1) occupancy grids consume large memory considering the whole scene's range. 2) UniAD failed to build the explicit interactions with the grid map. Therefore, we propose the BEV interactions that dynamically query the surrounding environment for each possible future step. This query process could help agents understand and learn the effects of their actions. Specifically, after applying the interactions above, the ego vehicle understands the dynamic agents' traffic better and knows its fronting road conditions. Nevertheless, planning a more precise motion requires the ego to comprehend the local detail geometric information. Hence, the ego query further interacts with BEV features to extract low-level geometric information. Specifically, we achieve by the deformable attention [58]:

$$\mathbf{E}''' = \texttt{DeformAttn}(\mathbf{E}'', p^t_{\mathbf{E}}, \mathbf{B}), \tag{4}$$

where $p^t_{\mathbf{E}}$ is the location of ego at time t, and it serves as the reference point on the BEV features. The deformable attention `DeformAttn` applies sparse attention around the reference points $p^t_{\mathbf{E}}$ and learns to pick up the low-level geometric information from the BEV for planning.

Motion Planning PPAD follows the same practice as [18, 19, 23], which uses the information of the high-level driving commands: go straight, turn left, and turn right. The concatenated features of $\mathbf{h_E} = [\mathbf{E}', \mathbf{E}'', \mathbf{E}''']$ contain the information of the dynamic agent traffic, map semantics, and the precise environmental geometry. An `MLP` takes $\mathbf{h_E}$ as input and predicts the future one-step waypoint offset $w^{t+1}_{\mathbf{E}} = (x, y)$. We then update the ego state by applying another `MLP` on $\mathbf{h_E}$ for the next step of processing.

3.3 Hierarchical Feature Learning

Hierarchical structure has a better capability to capture and recognize fine-grained patterns. For the driving scenarios, the driving behavior is based on scene understanding both globally and locally. Driving tends to focus on only a few key objects, which demonstrates the spatial locality or local attention. Therefore, we design hierarchical **key objects attention** to exploit the coarse-to-fine scene contexts. Specifically, given a set of distance ranges, we first find the key objects (agents or map elements) within the given range. Consequently, we apply dynamic local attention, which only considers the interactions among agents or map elements in the local area. The pseudo code shown in Alg. 1 delineates the implementation of dynamic key objects attention.

Algorithm 1 Pseudo code of Key objects attention in a PyTorch-like style.

```
################### initialization ###################
layer = nn.MultiheadAttention(embed_dim, num_head)
#################### forward pass ####################
def forward(layer, query, key, q_pos, k_pos, max_dis):
    # q_pos: position of the query with shape [B,Lq,2]
    # k_pos: position of the key with shape [B,Lk,2]
    # layer: attention layer as initialization
    # max_dis: distance threshold
    diff = (q_pos.unsqueeze(2) - k_pos.unsqueeze(1))
    dist = (diff ** 2).sum(-1).sqrt()
    attn_mask = (dist > max_dis).repeat(num_head, 1, 1)
    return attention(query, key, attn_mask=attn_mask)
```

3.4 Noisy Trajectory as Prediction

PPAD interleaves the prediction and the planning processes to plan the ego and agents' trajectories step-by-step. Expert driving knowledge is then enforced into the model through imitation learning. Thanks to our multi-step framework and the inspiration from [27], we introduce the noisy trajectory as the prediction to the PPAD while training. Specifically, we perturb each step of the ground truth ego trajectory by adding noise. The ego is then trained to predict the original next step waypoint offset of the ego regardless of disturbance on its starting noisy positions. The system is learned to predict the accurate waypoint offset by interacting with vectorized instances and the environment even though it starts at an inaccurate position. This strategy brings improvement to the planning performance.

3.5 End-to-End Learning

Scene Context Loss Similar to VAD [23], we formulate the loss for the agents' motion and map as follows:

$$\mathcal{L}_S = \lambda_1 \mathcal{L}_{agent} + \lambda_2 \mathcal{L}_{map}, \tag{5}$$

where λ_1 and λ_2 are set as 1.0.

Constraint Loss Inspired by [23], we propose the confidence-aware collision loss $\mathcal{L}_{CA-Col}$, which considers the potential collision of all the agents' motion modalities instead of only computing the loss on the agents' most confident mode. We multiply the resulting collision loss from each mode with the predicted confidence score. For the trajectories having the potential to collide, it will penalize more when the predicted confidence score is higher. Combined with ego-boundary overstepping $\mathcal{L}_{bd}$ and ego-lane directional $\mathcal{L}_{dir}$ constraints proposed by [23], the

	Method	L2 (m) ↓				Collision (%) ↓				Latency (ms)	FPS
		1s	2s	3s	Avg.	1s	2s	3s	Avg.		
ST-P3 Metrics	ST-P3 [18]	1.33	2.11	2.90	2.11	0.23	0.62	1.27	0.71	628.3	1.6
	VAD-Tiny [23]	0.46	0.76	1.12	0.78	0.21	0.35	0.58	0.38	**59.5**	**16.8**
	VAD-Base [23]	0.41	0.70	1.05	0.72	**0.07**	0.17	0.41	0.22	224.3	4.5
	OccNet [41]	1.29	2.13	2.99	2.13	0.21	0.59	1.37	0.72	-	-
	FusionAD [51]	-	-	-	1.03	0.25	0.13	**0.25**	0.21	-	-
	Ours (Progress.)	0.31	0.56	0.87	0.58	0.08	**0.12**	0.38	**0.19**	385	2.6
	Ours	**0.25**	**0.45**	**0.73**	**0.48**	**0.07**	0.15	0.36	**0.19**	385	2.6
UniAD Metrics	NMP† [53]	-	-	2.31	-	-	-	1.92	-	-	-
	SA-NMP† [53]	-	-	2.05	-	-	-	1.59	-	-	-
	FF† [17]	0.55	1.20	2.54	1.43	0.06	0.17	1.07	0.43	-	-
	EO† [24]	0.67	1.36	2.78	1.60	0.04	**0.09**	0.88	0.33	-	-
	UniAD [19]	0.48	0.96	1.65	1.03	0.05	0.17	**0.71**	**0.31**	555.6	1.8
	VAD-Base [23]	0.50	1.02	1.69	1.07	**0.00**	0.30	0.95	0.42	224.3	4.5
	Ours (Progress.)	0.38	0.83	1.45	0.89	0.02	0.20	0.93	0.38	385	2.6
	Ours	**0.30**	**0.69**	**1.26**	**0.75**	0.03	0.22	0.73	0.33	385	2.6

Table 1: Open-loop planning results on the nuScenes dataset [3]. The results of other methods are obtained from the original paper. We faithfully re-evaluate VAD [23] based on the UniAD [19] metrics. As for our PPAD, we provide two versions of results that utilize different training strategies. Progress. means that we follow the progressive training pipeline as proposed in VAD [23], which trained all tasks except the planning task in the first 48 epochs and then finetuned with another 12 epochs for the planning task. The second row for our method trains the whole network for 60 epochs and then finetuned for another 12 epochs incorporating noisy trajectories. The latency of ST-P3 [18], VAD [23], and ours are measured on one NVIDIA Geforce RTX 3090 GPU, while UniAD is measured on one NVIDIA Tesla A100 GPU.

overall constraint loss is

$$\mathcal{L}_C = \lambda_3 \mathcal{L}_{CA-Col} + \lambda_4 \mathcal{L}_{bd} + \lambda_5 \mathcal{L}_{dir}, \tag{6}$$

where λ_3 and λ_4 are set as 1.0, and λ_5 is set as 0.5.

Planning Loss We conduct L_1 loss between each step of the ego's prediction $w_{\mathbf{E}}^t$ and the ground truth's waypoint offset $\tilde{w}_{\mathbf{E}}^t$ along the future time horizon:

$$\mathcal{L}_{Plan} = \frac{1}{T_{fut}} \sum_{t=1}^{T_{fut}} \|w_{\mathbf{E}}^t - \tilde{w}_{\mathbf{E}}^t\|_1. \tag{7}$$

The overall end-to-end trainable loss function is formed by the sum of the perception loss, constraint loss, and planning loss. The same constraint losses $\mathcal{L}_C^{noisy}$ and planning losses $\mathcal{L}_{plan}^{noisy}$ will be applied to the predictions taking the noisy trajectories as input:

$$\mathcal{L} = \mathcal{L}_S + \zeta_1(\mathcal{L}_C + \mathcal{L}_{Plan}) + \zeta_2(\mathcal{L}_C^{noisy} + \mathcal{L}_{Plan}^{noisy}), \tag{8}$$

where ζ_1 is set as 0.6 and ζ_2 is set as 0.4.

Method	L2 (m) ↓				Collision (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.
VAD [23]	1.04	1.79	2.60	1.81	0.61	1.26	2.25	1.37
Ours	**0.73**	**1.30**	**1.98**	**1.34**	**0.51**	**0.85**	**1.31**	**0.89**

Table 2: Open-loop planning results on the Argoverse dataset [3]. The results are evaluated under the ST-P3 [18] and VAD [23] metrics.

Method	Detection		Map	Motion Forecasting		
	NDS ↑	mAP↑	mAP↑	minADE (m)↓	minFDE (m) ↓	MR↓
VAD [23]	0.459	0.329	0.476	0.678	**0.882**	0.08
UniAD [19]	**0.499**	**0.382**	-	0.708	1.02	0.13
Ours	0.465	0.332	**0.519**	**0.676**	0.889	**0.07**

Table 3: Results comparison on the tasks beyond the planning task.

4 Experiments

4.1 Experimental Setup

Dataset We evaluate our method on two challenging large-scale real-world datasets, nuScenes [3] and Argoverse2 [44]. We conduct ablation studies to evaluate the effectiveness of our proposed components on the nuScenes [3] dataset. The nuScenes dataset [3] provides about 1K 20-second diverse driving scenes collected in Boston, Pittsburgh, Las Vegas, and Singapore for open-loop settings. Key samples which contain 6 camera images are annotated at 2Hz. The Argoverse2 dataset comprises 1K 15-second scenes. It captures 7 RGB images on each frame. We align the data sampling frequency to nuScenes [3] by downsampling the annotated frames by an interval of 5 frames.

Metrics We adopt the metrics of L2 Displacement Error (L2) and Collision Rate (CR) for evaluation [18]. L2 is measured between the prediction and ground-truth trajectories over the timesteps 1-s, 2-s, and 3-s, evaluating the trajectory quality. Collision Rate (CR) measures how often the collision occurs between the ego vehicle and the other agents along the planning horizon, reflecting the trajectory safety. We notice that the UniAD [19] adheres to different calculations from ST-P3 [18] and VAD [23]: the former [19] reports the evaluations at each second. In contrast, the latters [18, 23] reports the results of the cumulative average by each second. We faithfully make comparisons of our method to others with these two kinds of computations.

Implementation Details We strictly follow the standard settings as proposed by UniAD [19] and VAD [23], which did not use the information of the historical ego trajectory. Same as VAD [23], the perception range is $60m \times 30m$. PPAD also recurrently encodes 2-s historical information into BEV and predicts the 3-s trajectory in the future. It also conducts the tasks of motion prediction and map construction. Our PPAD trains with a batch size of 1 using the Adam [25]

optimizer set with an initial learning rate of 2e-4. It takes us about six days to conduct the end-to-end training on eight NVIDIA A30 24GB GPUs.

4.2 Main Results

Planning Results As shown in Tab. 1, our PPAD outperforms the current state-of-the-art performance by a large margin. Especially for the L2 distance metrics, there are about 20% of consistent improvements can be observed along the temporal horizon. Thanks to the iterative interaction of prediction and planning, PPAD can help avoid collisions, leading to better results on collision rate compared to the one-shot representative VAD [23]. At the same time, we maintain a competitive efficiency compared to UniAD [19].

We further make a fair comparison between our method and the baseline, VAD [23], on the other dataset of Argoverse 2 [6, 44]. In Tab. 2, our method can consistently outperform the baseline with a clear margin in both L2 distance and collision rate metrics.

Subtasks Results To demonstrate the overall performance of our PPAD, we also provide the evaluation results besides planning metrics on the traditional perception and motion forecasting task in Tab. 3. Our PPAD also achieves promising performance in upstreaming perception and prediction tasks, which demonstrates that the whole system is jointly optimized.

Qualitative Results We provide qualitative results shown in Fig. 3. PPAD can perceive the scene precisely and predict with reasonable and diverse motions for the surrounding agents. It also plans a smooth and accurate trajectory for the ego vehicle.

4.3 Ablation Study

The following experiments adhere to the progressive training pipeline as proposed in VAD [23].

Effectiveness of Designs We provide ablation studies to verify the effectiveness of our proposed components. As shown in Tab. 4, the proposed PPAD framework (row 2) brings a remarkable improvement compared with one-shoot methods [23] (row 1). The multi-step interactions help the ego agent better understand the intention and potential effects brought by its actions along the temporal horizon, leading to an over 10% L2 distance error reduction. We can observe a further improvement regarding the L2 distance with our proposed key objects attention (row 3). The slight degradation in collision rate might be due to the key objects attention being conducted on the ego with each mode of the agents, and the diverse modes of the agents mislead the behavior of the ego vehicle. From rows 5-7, the confidence-aware collision loss and noisy trajectory as the prediction can circumvent this phenomenon and further escalate the capability in planning accuracy and avoiding a collision. When we keep all of the remaining components while not applying key objects attention (row 4), we observed the degradations from row 7 in L2 (0.58 m vs. 0.59 m) and collision rate (0.19% vs. 0.21%), proving the effectiveness of the key objects attention.

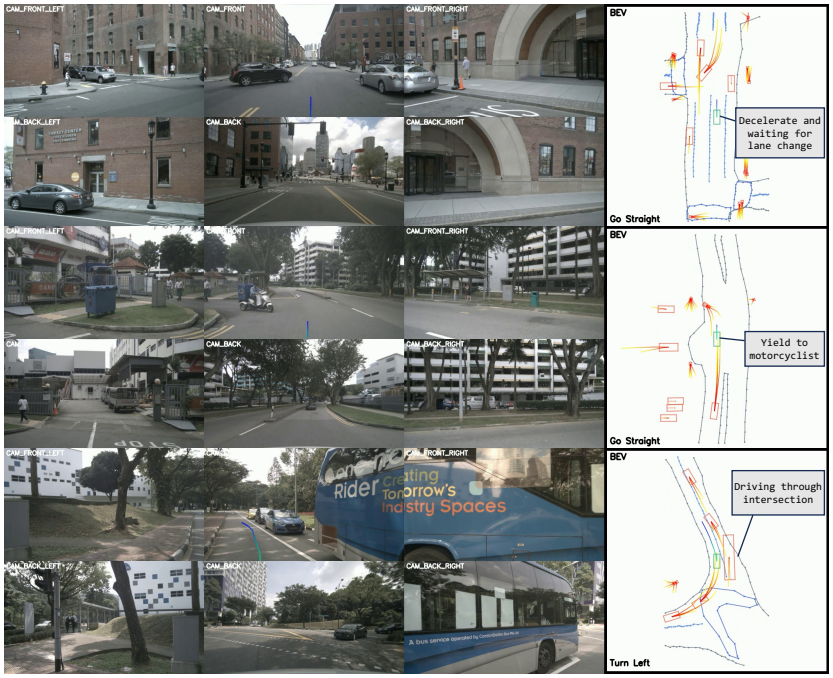


Fig. 3: Qualitative results of PPAD. The green box in the figure demonstrates the ego agent, while the red ones are agents.

	PPAD	Key Objects Attn.	$\mathcal{L}_{CA-Col}$	Noisy Traj.	L2 (m) ↓				Collision (%) ↓				Latency (ms)
					1s	2s	3s	Avg.	1s	2s	3s	Avg.	
1	-	-	-	-	0.41	0.70	1.05	0.72	**0.07**	0.17	0.41	0.22	224
2	✓	-	-	-	0.35	0.59	0.88	0.60	0.11	0.18	0.41	0.23	318
3	✓	✓	-	-	0.32	**0.55**	**0.83**	**0.57**	0.19	0.28	0.58	0.35	385
4	✓	-	✓	✓	0.33	0.57	0.88	0.59	0.10	0.14	0.38	0.21	318
5	✓	✓	✓	-	0.35	0.59	0.89	0.61	0.08	0.14	**0.32**	**0.18**	385
6	✓	✓	-	✓	0.34	0.59	0.90	0.61	0.10	0.14	0.34	0.19	385
7	✓	✓	✓	✓	**0.31**	0.56	0.87	0.58	0.08	**0.12**	0.38	0.19	385

Table 4: Component study for PPAD. Models follow the progressive training pipeline. PPAD means the auto-regressive framework with the designed stepwise interactions. Key Objects Attention represents hierarchical feature learning for the key objects. $\mathcal{L}_{CA-Col}$ represents the loss design for the confidence-aware collision loss. Noisy Traj. means that we incorporate noisy trajectories while training.

Effectiveness of Interactions Our PPAD framework enables richer interactions among scene contexts, introducing local and global understandings of the world to the model. We further conduct ablation studies to demonstrate the performance gains brought by the interactions. We conduct the ablation study on interactions under the setting (Tab. 4 in row 5) without using the noisy trajectory as the prediction for better comparison. As illustrated in Tab. 5, we can achieve

EA	Map	BEV	L2 (m) ↓				Collision (%) ↓			
			1s	2s	3s	Avg.	1s	2s	3s	Avg.
✓	-	-	0.35	0.58	0.89	0.61	0.23	0.24	0.49	0.32
✓	✓	-	0.38	0.64	0.95	0.66	0.17	0.18	0.48	0.28
✓	✓	✓	0.35	0.59	0.89	0.61	0.08	0.14	0.32	0.18

Table 5: Interaction study for PPAD. EA, Map, BEV mean the interactions of the ego with agents, the ego and agents with the map, the ego and agents with the BEV, respectively.

Interaction Iterations	L2 (m) ↓				Collision (%) ↓				Latency (ms)
	1s	2s	3s	Avg.	1s	2s	3s	Avg.	
2 (Every 1.5 sec)	0.36	0.63	0.96	0.65	0.14	0.20	0.43	0.25	306
3 (Every 1.0 sec)	0.37	0.63	0.95	0.65	0.10	0.15	0.39	0.21	326
6 (Every 0.5 sec)	**0.31**	**0.56**	**0.87**	**0.58**	**0.08**	**0.12**	**0.38**	**0.19**	385

Table 6: The ablation study of conducting different interaction iterations in the future time horizons.

the best performance in terms of L2 distance and collision by incorporating all of the interactions.

Effect of Different Iterations on Prediction and Planning Interaction Our innovative PPAD interaction mechanism not only better models motion planning as gaming among the ego and the agents but also enriches the ego's / agents' interactions with their local environments. We further conducted the ablation study on the different iterations in applying the interactions of prediction and planning, as shown in Tab. 6. Specifically, PPAD plans the trajectories with 3, 2, and 1 steps of waypoints after each of the prediction-planning processes for the interaction iterations of 2, 3, and 6 in Tab. 6. It is demonstrated that the performance reaches the best as we conduct the interactions of prediction and planning processes at every future step.

5 Conclusion

In this paper, we have presented a novel autonomous driving framework, **PPAD**. Different from the previous methods that lack in-depth modeling of interactions, we pose the planning problem as a multi-step **Prediction** and **Planning** gaming process among the ego vehicle and agents. With **PPAD** architecture, our proposed hierarchical dynamic key objects attention is incorporated to learn local and global scene contexts at each step and eventually plan with a more precise trajectory. The confidence-aware collision constraint and noisy trajectories are utilized while training to improve driving safety further. In general, our proposed novel **PPAD** achieves compelling performance upon the existing state-of-the-art methods, and we hope the **PPAD** framework can inspire the community to further exploration.

Acknowledgements

The authors are thankful for the financial support from the Hetao Shenzhen-HongKong Science and Technology Innovation Cooperation Zone (HZQB-KCZYZ-2021055), this work was also supported by Shenzhen Deeproute.ai Co., Ltd (HZQB-KCZYZ-2021055).

References

1. Bansal, M., Krizhevsky, A., Ogale, A.: Chauffeurnet. In: Robotics: Science and Systems XV (2019)
2. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., et al.: End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316 (2016)
3. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11621–11631 (2020)
4. Casas, S., Luo, W., Urtasun, R.: Intentnet: Learning to predict intention from raw sensor data. In: Conference on Robot Learning. pp. 947–956 (2018)
5. Chai, Y., Sapp, B., Bansal, M., Anguelov, D.: Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. arXiv preprint arXiv:1910.05449 (2019)
6. Chang, M.F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., et al.: Argoverse: 3d tracking and forecasting with rich maps. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8748–8757 (2019)
7. Chekroun, R., Toromanoff, M., Hornauer, S., Moutarde, F.: Gri: General reinforced imitation and its application to vision-based autonomous driving. arXiv preprint arXiv:2111.08575 (2021)
8. Chen, D., Zhou, B., Koltun, V., Krähenbühl, P.: Learning by cheating. In: Conference on Robot Learning. pp. 66–75. PMLR (2020)
9. Chitta, K., Prakash, A., Jaeger, B., Yu, Z., Renz, K., Geiger, A.: Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. IEEE Transactions on Pattern Analysis and Machine Intelligence (2022)
10. Codevilla, F., Santana, E., López, A.M., Gaidon, A.: Exploring the limitations of behavior cloning for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9329–9338 (2019)
11. Duvenaud, D.K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P.: Convolutional networks on graphs for learning molecular fingerprints. In: Advances in neural information processing systems. pp. 2224–2232 (2015)
12. Fan, H., Zhu, F., Liu, C., Zhang, L., Zhuang, L., Li, D., Zhu, W., Hu, J., Li, H., Kong, Q.: Baidu apollo em motion planner. arXiv preprint arXiv:1807.08048 (2018)
13. Gao, J., Sun, C., Zhao, H., Shen, Y., Anguelov, D., Li, C., Schmid, C.: Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11525–11533 (2020)

14. Gu, J., Hu, C., Zhang, T., Chen, X., Wang, Y., Wang, Y., Zhao, H.: Vip3d: End-to-end visual trajectory prediction via 3d agent queries. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5496–5506 (2023)
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
16. Henaff, M., Bruna, J., LeCun, Y.: Deep convolutional networks on graph-structured data. arXiv preprint arXiv:1506.05163 (2015)
17. Hu, P., Huang, A., Dolan, J., Held, D., Ramanan, D.: Safe local motion planning with self-supervised freespace forecasting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12732–12741 (2021)
18. Hu, S., Chen, L., Wu, P., Li, H., Yan, J., Tao, D.: St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In: European Conference on Computer Vision. pp. 533–549. Springer (2022)
19. Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., et al.: Planning-oriented autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17853–17862 (2023)
20. Huang, J., Huang, G., Zhu, Z., Ye, Y., Du, D.: Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv preprint arXiv:2112.11790 (2021)
21. Huang, Z., Liu, H., Lv, C.: Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. arXiv preprint arXiv:2303.05760 (2023)
22. Jia, X., Wu, P., Chen, L., Xie, J., He, C., Yan, J., Li, H.: 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. pp. 21983–21994 (2023)
23. Jiang, B., Chen, S., Xu, Q., Liao, B., Chen, J., Zhou, H., Zhang, Q., Liu, W., Huang, C., Wang, X.: Vad: Vectorized scene representation for efficient autonomous driving. arXiv preprint arXiv:2303.12077 (2023)
24. Khurana, T., Hu, P., Dave, A., Ziglar, J., Held, D., Ramanan, D.: Differentiable raycasting for self-supervised occupancy forecasting. In: European Conference on Computer Vision. pp. 353–369. Springer (2022)
25. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
26. Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O.: PointPillars: Fast Encoders for Object Detection From Point Clouds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12697–12705 (2019)
27. Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L.: Dn-detr: Accelerate detr training by introducing query denoising. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13619–13627 (2022)
28. Li, Y., Ge, Z., Yu, G., Yang, J., Wang, Z., Shi, Y., Sun, J., Li, Z.: Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 1477–1485 (2023)
29. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: European conference on computer vision. pp. 1–18. Springer (2022)

30. Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., Urtasun, R.: Learning lane graph representations for motion forecasting. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 541–556 (2020)
31. Liao, B., Chen, S., Wang, X., Cheng, T., Zhang, Q., Liu, W., Huang, C.: Maptr: Structured modeling and learning for online vectorized hd map construction. arXiv preprint arXiv:2208.14437 (2022)
32. Liu, Y., Zhang, J., Fang, L., Jiang, Q., Zhou, B.: Multimodal motion prediction with stacked transformers. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7577–7586 (2021)
33. Liu, Y., Wang, T., Zhang, X., Sun, J.: Petr: Position embedding transformation for multi-view 3d object detection. In: European Conference on Computer Vision. pp. 531–548. Springer (2022)
34. Liu, Y., Yan, J., Jia, F., Li, S., Gao, A., Wang, T., Zhang, X., Sun, J.: Petrv2: A unified framework for 3d perception from multi-camera images. arXiv preprint arXiv:2206.01256 (2022)
35. Liu, Z., Chen, S., Guo, X., Wang, X., Cheng, T., Zhu, H., Zhang, Q., Liu, W., Zhang, Y.: Vision-based uneven bev representation learning with polar rasterization and surface estimation. In: Conference on Robot Learning. pp. 437–446. PMLR (2023)
36. Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D.L., Han, S.: Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 2774–2781. IEEE (2023)
37. Luo, W., Yang, B., Urtasun, R.: Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 3569–3577 (2018)
38. Philion, J., Fidler, S.: Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16. pp. 194–210. Springer (2020)
39. Renz, K., Chitta, K., Mercea, O.B., Koepke, A., Akata, Z., Geiger, A.: Plant: Explainable planning transformers via object-level representations. arXiv preprint arXiv:2210.14222 (2022)
40. Shuman, D.I., Narang, S.K., Frossard, P., Ortega, A., Vandergheynst, P.: The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. IEEE signal processing magazine **30**(3), 83–98 (2013)
41. Sima, C., Tong, W., Wang, T., Chen, L., Wu, S., Deng, H., Gu, Y., Lu, L., Luo, P., Lin, D., et al.: Scene as occupancy. arXiv e-prints pp. arXiv–2306 (2023)
42. Song, H., Luan, D., Ding, W., Wang, M.Y., Chen, Q.: Learning to predict vehicle trajectories with model-based planning. In: Conference on Robot Learning. pp. 1035–1045. PMLR (2021)
43. Wang, S., Liu, Y., Wang, T., Li, Y., Zhang, X.: Exploring object-centric temporal modeling for efficient multi-view 3d object detection. arXiv preprint arXiv:2303.11926 (2023)
44. Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J.K., et al.: Argoverse 2: Next generation datasets for self-driving perception and forecasting. arXiv preprint arXiv:2301.00493 (2023)

45. Wu, P., Jia, X., Chen, L., Yan, J., Li, H., Qiao, Y.: Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. Advances in Neural Information Processing Systems **35**, 6119–6132 (2022)
46. Xie, Y., Xu, C., Rakotosaona, M.J., Rim, P., Tombari, F., Keutzer, K., Tomizuka, M., Zhan, W.: Sparsefusion: Fusing multi-modal sparse representations for multi-sensor 3d object detection. arXiv preprint arXiv:2304.14340 (2023)
47. Yang, C., Chen, Y., Tian, H., Tao, C., Zhu, X., Zhang, Z., Huang, G., Li, H., Qiao, Y., Lu, L., et al.: Bevformer v2: Adapting modern image backbones to bird's-eye-view recognition via perspective supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17830–17839 (2023)
48. Ye, M., Cao, T., Chen, Q.: Tpcn: Temporal point cloud networks for motion forecasting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11318–11327 (2021)
49. Ye, M., Xu, S., Cao, T.: Hvnet: Hybrid voxel network for lidar based 3d object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1631–1640 (2020)
50. Ye, M., Xu, S., Cao, T., Chen, Q.: Drinet: A dual-representation iterative learning network for point cloud segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 7447–7456 (2021)
51. Ye, T., Jing, W., Hu, C., Huang, S., Gao, L., Li, F., Wang, J., Guo, K., Xiao, W., Mao, W., et al.: Fusionad: Multi-modality fusion for prediction and planning tasks of autonomous driving. arXiv preprint arXiv:2308.01006 (2023)
52. Zeng, W., Liang, M., Liao, R., Urtasun, R.: Lanercnn: Distributed representations for graph-centric motion forecasting. arXiv preprint arXiv:2101.06653 (2021)
53. Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., Urtasun, R.: End-to-end interpretable neural motion planner. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8660–8669 (2019)
54. Zhang, Z., Liniger, A., Dai, D., Yu, F., Van Gool, L.: End-to-end urban driving by imitating a reinforcement learning coach. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 15222–15232 (2021)
55. Zhou, Y., Tuzel, O.: VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4490–4499 (2018)
56. Zhou, Z., Wang, J., Li, Y.H., Huang, Y.K.: Query-centric trajectory prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17863–17873 (2023)
57. Zhou, Z., Ye, L., Wang, J., Wu, K., Lu, K.: Hivt: Hierarchical vector transformer for multi-agent motion prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8823–8833 (2022)
58. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)