Divide and Merge: Motion and Semantic Learning in End-to-End Autonomous Driving

Anonymous Author(s)
Affiliation
Address
email

Abstract

Perceiving the environment and its changes over time corresponds to two fundamental yet heterogeneous types of information: semantics and motion. Previous end-to-end autonomous driving works represent both types of information in a single feature vector. However, including motion related tasks, such as prediction and planning, impairs detection and tracking performance, a phenomenon known as negative transfer in multi-task learning. To address this issue, we propose NeuralBayes motion decoding, a novel parallel detection, tracking, and prediction method that separates semantic and motion learning. Specifically, we employ a set of learned motion queries that operate in parallel with detection and tracking queries, sharing a unified set of recursively updated reference points. Moreover, we employ interactive semantic decoding to enhance information exchange in semantic tasks, promoting positive transfer. Experiments on the nuScenes dataset with UniAD and SparseDrive confirm the effectiveness of our divide and merge approach, resulting in performance improvements across perception, prediction, and planning. The code will be released.

1 Introduction

Modular end-to-end (E2E) autonomous driving (AD) is gaining attention for combining the strengths of traditional pipeline methods with strict E2E approaches. In this framework, perception, prediction, and planning form the core set of tasks, which ideally complement one another to enhance overall system performance. However, the modular E2E framework also presents a multi-task learning challenge. A poorly designed multi-task learning structure could not only fail to facilitate mutual learning but also adversely affect individual tasks, a phenomenon known as negative transfer [1]. The prevalent modular E2E approaches [2–5] typically employ a sequential structure (Fig. 1a). This structure aligns with how humans perform driving tasks and has demonstrated promising planning performance. However, these approaches exhibit negative transfer in object detection and tracking. In other words, the perception performance of jointly trained E2E models is typically inferior to those trained without the motion prediction and planning tasks.

We analyze the underlying causes of negative transfer by inspecting the types of learned heterogeneous information: semantic and motion. Semantic information encompasses the categories of surrounding objects, lanes, crossings, *etc*., while motion information describes the temporal changes occurring within the environment. Sequential methods [2–4, 6] execute these two processes in succession. They first conduct detection and tracking and then use the extracted object features for trajectory prediction. This sequential design forces the features to contain motion information, compromising the initially learned semantic and leading to negative transfer in perception. The SHAP values analysis [7] provides supporting evidence for our argument. Another E2E structure is depicted in Fig. 1b. It

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

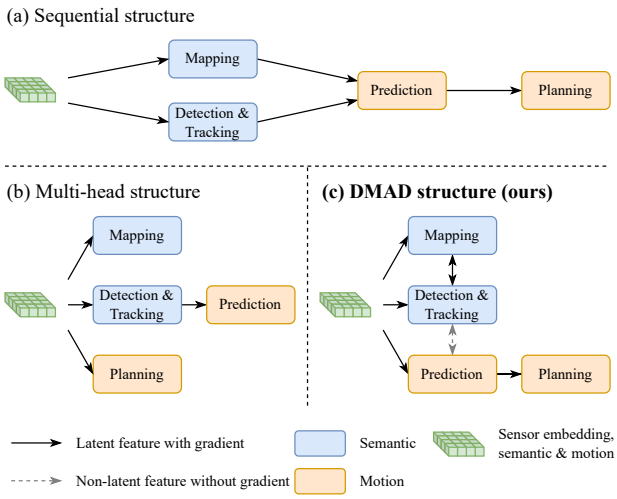


Figure 1: **Comparison of E2E structures.** In (a), semantic and motion learning occur sequentially. In (b), the multi-head structure parallelizes tasks with different heads; however, motion and semantic learning remain sequential in detection, tracking, and prediction. In (c), semantic and motion learning are performed in parallel without latent feature sharing or gradient propagation. In contrast, the exchange of information between the object and map perception modules is enhanced.

executes most tasks with different heads in parallel, as PARA-Drive [8] and NMP [9]. However, since detection and prediction remain sequential, the issue of negative transfer persists.

In this work, we propose **DMAD structure** (Fig. 1c), **D**ividing and **M**erging motion and semantic learning for E2E **A**utonomous **D**riving. DMAD addresses the issue of negative transfer by separating semantic and motion learning. Furthermore, it leverages correlations among semantic tasks by merging them.

For dividing, we propose **Neural-Bayes motion decoder**. We maintain a set of motion queries that attend to the sensor embeddings parallel to the object (detection and tracking) queries. The key difference between motion and object queries is that they are decoded into past and future trajectories rather than bounding boxes with classes. Motion and object queries share a single set of reference points, updated recursively by detection and prediction. It allows only limited information exchange between both types of queries, mediated through the reference points without gradient flow. Moreover, we calculate the object's velocity using the predicted trajectory with finite differences, thereby removing the requirement for object queries to learn the velocity directly. In this manner, the object query focuses on learning semantic and appearance features, while the motion query is dedicated to capturing motion features. The two types of heterogeneous information are learned separately along distinct paths, effectively preventing negative transfer. Notably, the DMAD structure promotes motion learning to the same level of semantic learning, treating detection, tracking, and prediction as concurrent tasks for the first time, to the best of our knowledge.

For merging, we propose **interactive semantic decoder** to enhance the exchange of semantic insights in detection and map segmentation. Object perception and map perception are inherently related tasks. Previous methods often overlook this connection, typically executing the two along parallel paths [2–4]. DualAD [6] leverages this correlation but allows only object perception to learn from the map. Our method uses layer-wise iterative self-attention [10] to enable mutual learning between object and map tasks, fostering positive transfer.

Experiments on the nuScenes [11] dataset showcase the effectiveness of DMAD structure in mitigating negative transfer. Our approach achieves significant performance gains in perception and prediction, which benefits the planning module and outperforms state-of-the-art (SOTA) E2E AD models.

Our key contributions are summarized as follows:

- We examine the similarity and heterogeneity among tasks in modular E2E AD and argue that the prevailing design—learning information for conflicting tasks within a single feature—is

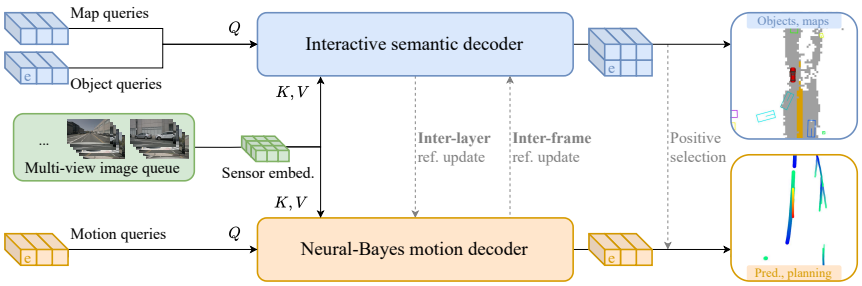


Figure 2: **An overview of DMAD.** A backbone processes multi-view images into sensor embeddings. Map and object queries are initialized, then interactively attend to the sensor embeddings for map and object perception. Motion queries, mapped one-to-one with object queries, share reference points that are iteratively updated. Finally, motion queries corresponding to detected objects are decoded into future trajectories. The ego motion query ("e") is used for planning. Gray dashed lines indicate operations without gradient flow.

the cause of negative transfer in perception. We analyze SHAP values to validate this hypothesis. Conversely, we propose that information exchange between similar tasks can facilitate positive transfer.

- We propose DMAD, a modular E2E AD paradigm that divides and merges tasks according to the information they are supposed to learn. This design eliminates negative transfer between different types of tasks while reinforcing positive transfer among similar tasks.
- We introduce two decoders: the Neural-Bayes motion decoder for concurrent trajectory prediction with object detection and tracking; the interactive semantic decoder to enhance information sharing between object and map perception. The proposed decoders improve existing SOTA methods, leading to better performance across all tasks.

2 Method

Figure 2 shows an overview of DMAD structure. Sensor embeddings are extracted from multi-view camera images and are shared across all tasks, including detection, tracking, mapping, prediction, and planning. We initialize three distinct types of queries—object, map, and motion—which attend to the sensor embeddings to extract the specific information required for each respective task. Based on the type of information learned, the decoding process is divided into two pathways. On one way, object and map decoding are jointly performed within the **Interactive semantic decoder**, where both types of queries iteratively exchange latent semantic information at each decoding layer. On the other way, motion queries extract motion information from the sensor embeddings within the **Neural-Bayes motion decoder**. Each motion query is paired with an object query, using the object's coordinates as a reference point at each decoding layer. After decoding each frame, the motion query's predicted future waypoint becomes the object query's reference point in the next frame, similar to the recursion of a Bayes filter [12]. The exchange of reference points is always without gradient. At last, the motion queries are passed on to the planning module. The system is fully E2E trainable, with motion and semantic gradients propagated in distinct paths.

2.1 Interactive semantic decoder

To leverage the semantic correlation between individual objects and map elements, we introduce the Interactive Semantic Decoder. In contrast to the unidirectional interaction in DualAD [6], our approach enables a bidirectional exchange of information.

We initialize a set of object queries $Q_{\text{obj}} \in \mathbb{R}^{N_{\text{obj}} \times d}$ and a set of map queries $Q_{\text{map}} \in \mathbb{R}^{N_{\text{map}} \times d}$. The number of queries could be different, while the dimensions d must be the same. Each decoding layer first concatenates both types of queries. Self-attention [10] is then applied, where both tasks exchange their semantic information. Subsequently, the two types of queries are divided, each performing self-attention and cross-attention on the sensor embeddings, respectively, as shown in Fig. 3.

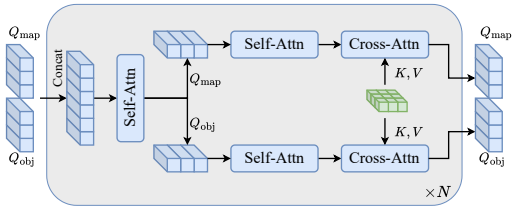


Figure 3: **Interactive semantic decoding.** Object and map queries are concatenated and interact through a self-attention module before being separated to independently attend to the sensor embeddings. This process is repeated across N stacked layers.

After interactive semantic decoding, each object query is classified into a category c and regressed into a vector $[\Delta x, \Delta y, \Delta z, w, h, l, \theta]^{\mathsf{T}}$. The object query is associated with a reference point $[x_{\text{ref}}, y_{\text{ref}}, z_{\text{ref}}]^{\mathsf{T}}$. Rather than directly learning the absolute coordinates of the object, it learns the offsets relative to its corresponding reference points. Thus, the bounding boxes can be represented as $[x_{\text{ref}} + \Delta x, y_{\text{ref}} + \Delta y, z_{\text{ref}} + \Delta z, w, h, l, \theta]^{\mathsf{T}}$. Notably, velocities are not regressed, as they pertain to motion information. We design the object queries to focus solely on semantic information, *i.e.*, the object's category, center point, size, and orientation.

2.2 Neural-Bayes motion decoder

We introduce a novel motion decoder operating in parallel with the semantic decoder, aimed at fully decoupling motion and semantic learning to reduce the negative transfer in semantic tasks. Given the correlation between motion and semantics, we design a recursive process to facilitate the exchange of human-readable information between the two decoders as illustrated in Fig. 4, which comprises the processes of prediction, measurement, and updating, similar to the Bayes filter [12]. Appendix C provides a brief introduction to the Bayes filter. We proceed with the elaboration of the proposed motion decoder.

Initialization. We initialize a set of motion queries $Q_{\text{mt}} \in \mathbb{R}^{N_{\text{mt}} \times d}$ in the same way we initialize object queries. The motion queries correspond one-to-one with the object queries, *i.e.*, $N_{\text{mt}} = N_{\text{obj}}$. However, since they do not directly interact in the latent space, their dimensionalities d can differ. Each motion query represents the motion state of an object, although the model does not initially know whether the object exists. Additionally, motion queries and object queries share a common set of reference points.

Measurement. The detection, already introduced in Sec. 2.1, is treated as the measurement in Bayes filter. After each semantic decoding layer, the object queries are regressed, yielding the coordinate vectors $\mathbf{ref} = [x, y, z]^{\mathsf{T}}$ of the tentative object, which then serves as reference points for the next layer:

$$\mathbf{ref}^{l+1} = f_{\text{reg}}(f^{l}_{\text{Semantic-Dec}}(Q^{l}_{\text{obj}}, Z, \mathbf{ref}^{l})), \tag{1}$$

where the superscript denotes the layer and Z is the sensor embeddings.

Updating. With the reference points $\mathbf{ref}^{l}$ from the semantic decoding (the inter-layer reference points update in Fig. 2), the motion queries also attend to the sensor embeddings via cross-attention:

$$Q^{l+1}_{\text{mt}} = f^{l}_{\text{Motion-Dec}}(Q^{l}_{\text{mt}}, Z, \mathbf{ref}^{l}), \tag{2}$$

where the motion queries are updated conditioned on the measured reference points.

Prediction. We employ MLPs to extract trajectories from the motion queries. We note that motion extraction occurs in two stages: first through the unimodal trajectory construction, followed by the multimodal prediction.

The first stage computes the unimodal velocity and future reference points, guiding the motion query to learn aggregated motion states from the past and predict the near future. It produces a single

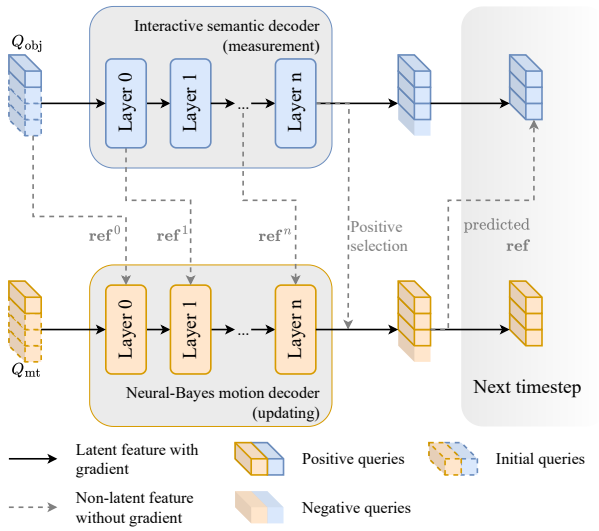


Figure 4: **Neural-Bayes motion decoding.** After each decoding layer, the semantic decoder updates the reference points, which are then shared with the motion decoder. At the end of each frame, positive object query indices are used to select corresponding motion queries and are together propagated to the subsequent frame, with the motion query predictions serving as reference points for the next frame. This process is similar to the measurement, updating, and prediction steps in a Bayes filter. Map queries, ego queries and sensor embeddings are omitted for simplicity.

trajectory that spans from the past timestep t_{past} to the future timestep $t_{\text{fut-1}}$. The velocity is calculated using the finite difference method on waypoints around the current timestep. We use the first future waypoint as the initial reference point for the object query in the next frame, *i.e.*, inter-frame reference points update in Fig. 2, for object tracking.

The second stage performs multimodal intention modeling and generates multiple future trajectories within the future $t_{\text{fut-2}}$ timesteps, along with their corresponding confidence scores.

Tracking. Multi-object tracking is performed using the query propagation mechanism [13, 14]. Each object query is associated with an unique instance ID. A positive query propagates across consecutive frames, ensuring that corresponding detections are assigned the same ID. During training, object queries associated with ground truth are referred to as positive queries; during inference, positivity is determined by whether the confidence score exceeds a specified threshold. The propagation of motion queries follows that of object queries, as they are related. This mechanism enables continuous measuring, updating, and predicting, similar to the Bayes filter.

3 Experiments

We conduct experiments on the nuScenes [11] dataset to validate the effectiveness of our method. We present results in three parts. The first part focuses on perception (detection, tracking, and mapping). In the second part, we evaluate motion prediction and planning. Lastly, we provide an extensive ablation study and SHAP values [7] visualization.

3.1 Training configuration

We reproduce UniAD [2] and SparseDrive [5] as baselines. Both utilize the query propagation mechanism; however, UniAD extracts dense bird's-eye view (BEV) features from image inputs, while SparseDrive employs sparse scene representations. Beside the aforementioned tasks, UniAD additionally performs occupancy prediction. We also retain the occupancy module in comparisons with UniAD for task consistency. As occupancy prediction serves merely as another representation of upstream tasks, we describe it in Appendix D. We adhere as closely as possible to default

Table 1: **Perception results.** The performance changes in stage 2 are expressed as percentages, with red indicating a decline and blue representing improvement.

(a) Object detection results.

Method	NDS↑	mAP↑	mAVE↓
VAD [3]	0.460	0.330	0.405
GenAD [4]	0.280	0.213	0.669
PARA-Drive [8]	0.480	0.370	-
UniAD - stage 1	0.497	0.382	0.411
UniAD - stage 2	0.491 (-1.2%)	0.377 (-1.3%)	0.412 (+0.2%)
DMAD - stage 1	0.504	0.395	0.406
DMAD - stage 2	0.506 (+0.4%)	0.396 (+0.3%)	0.395 (-2.7%)
SparseDrive - stage 1	0.531	0.419	0.257
SparseDrive - stage 2	0.523 (-1.5%)	0.417 (-0.5%)	0.269 (+4.7%)
SparseDMAD - stage 1	**0.536**	0.424	0.260
SparseDMAD - stage 2	0.534 (-0.4%)	**0.427** (+0.7%)	**0.253** (-2.7%)

(b) Multi-object tracking results.

Method	AMOTA↑	AMOTP↓	IDS↓
ViP3D [16]	0.217	1.63	-
MUTR3D [17]	0.294	1.50	3822
PARA-Drive [8]	0.350	-	-
UniAD - stage 1	0.374	1.31	816
UniAD - stage 2	0.354 (-5.3%)	1.34 (+2.3%)	1381 (+69%)
DMAD - stage 1	0.394	1.32	781
DMAD - stage 2	0.393 (-0.3%)	1.30 (-1.5%)	767 (-1.8%)
SparseDrive - stage 1	0.395	1.25	602
SparseDrive - stage 2	0.376 (-4.8%)	1.26 (+0.8%)	**559** (-7.1%)
SparseDMAD - stage 1	**0.396**	**1.23**	608
SparseDMAD - stage 2	0.395 (-0.3%)	**1.23** (0%)	571 (-6.1%)

configurations of the baseline; however, to ensure a rigorous comparisons, some adjustments are made. Following paragraphs outline the adjustments and the rationale behind them.

Two-stage training. We follow the two-stage training scheme of our baseline. In the first stage, we train object detection, tracking, and mapping. In the second stage, we train all modules together. Notably, because our tracking relies on reference points provided by unimodal prediction, we incorporate unimodal prediction training in the first stage. Multimodal prediction is trained only in the second stage, which is consistent with the baseline.

Queue length. Since AD is a time-dependent task, the model typically processes a sequence of consecutive frames as a training sample. The number of input frames, *i.e.*, the queue length q, defines the temporal horizon the model can capture, impacting the performance of related tasks. UniAD employs different queue lengths across its two training stages: 5 in the first stage and 3 in the second. The reduced queue length in the second stage degrades perception performance due to reduced temporal aggregation, shown in Appendix E. This degrading hinders the identification of negative transfer effects caused by the sequential structure. To mitigate this interference, we standardize the queue length to 3 across both training stages in comparisons with UniAD. Unless otherwise specified, the performance of UniAD in all result tables is reproduced with a queue length of 3 using the official codebase [15]. SparseDrive does not have this issue, and we use the default setting of 4.

Ego query represents the features directly used for motion planning, which is intended to capture the motion information of the ego vehicle. SparseDrive generates the ego query from the front camera image and the estimated previous ego status, which blends semantics and motion, thus contradicting our dividing design. To align with our proposal, we eliminate the use of the front image for the ego query when applying DMAD to SparseDrive. For UniAD, we retain the planning module unchanged, as it initializes the ego query randomly.

3.2 Perception

Metrics. For object detection and tracking, we use the metrics defined in the nuScenes benchmark. The primary metrics for detection are nuScenes Detection Score (NDS) and mean average precision (mAP). For multiple object tracking, we report the average multi-object tracking accuracy (AMOTA) and the average multi-object tracking precision (AMOTP). For map segmentation, we use the intersection over union (IoU) metric of drivable areas, lanes, and dividers. Vectorized mapping adopts mAP of lane divider, pedestrian crossing and road boundary.

Object detection. Table 1a presents the detection performance across two training stages. In the first stage, thanks to the interactive semantic decoding, our approach slightly outperforms the baseline. After the second stage of training, baseline's performance shows a decline. In contrast, our method preserves the perceptual performance of the first stage, benefiting from separated motion learning that mitigates negative transfer. Our method finally surpasses UniAD and SparseDrive by 3.1% and 2.1% in NDS, respectively.

Table 2: **Map perception results.**

(a) Map segmentation results.

Method	Lanes↑	Drivable↑	Dividers↑
BEVFormer [18]	0.239	**0.775**	-
PARA-Drive [8]	**0.330**	0.710	-
UniAD - stage 1	0.293	0.650	0.248
UniAD - stage 2	0.312 (+6.5%)	0.678 (+4.3%)	0.267 (+7.7%)
DMAD - stage 1	0.292	0.655	0.242
DMAD - stage 2	0.321 (+9.9%)	0.691 (+5.5%)	**0.271** (+12%)

(b) Vectorized mapping results.

Method	AP_{ped}↑	$AP_{divider}$↑	$AP_{boundary}$↑	mAP↑
MapTR [19]	**0.562**	0.598	0.601	**0.587**
VAD [3]	0.406	0.515	0.506	0.476
SparseDrive - stage 1	0.533	0.579	0.575	0.562
SparseDrive - stage 2	0.494 (-7.3%)	0.569 (-1.7%)	0.583 (+1.4%)	0.549 (-2.3%)
SparseDMAD - stage 1	0.553	0.599	**0.606**	0.586
SparseDMAD - stage 2	0.554 (+0.2%)	**0.601** (+0.3%)	**0.606** (0%)	**0.587** (+0.2%)

Table 3: **Trajectory prediction results.** C and P stand for cars and pedestrians respectively.

Method	EPA↑		minADE↓	
	C	P	C	P
ViP3D [16]	0.226	-	2.05	-
GenAD [4]	**0.588**	0.352	0.84	0.84
UniAD	0.495	0.361	0.69	0.79
DMAD	0.535	**0.416**	0.72	0.77
SparseDrive	0.487	0.406	**0.63**	0.73
SparseDMAD	0.500	0.410	**0.63**	**0.71**

Multi-object tracking. Due to using a single feature vector to represent semantics and motion, UniAD and SparseDrive exhibit negative transfer of 5.3% and 4.8% in AMOTA, as shown in Tab. 1b. Our dividing design enables object queries to learn about appearance more effectively. At the same time, unimodal predictions offer enhanced tracking reference points. Consequently, our method achieves a gain of 11.0% and 5.1% in AMOTA, respectively.

Map perception. UniAD does not encounter negative transfer in map segmentation. Leveraging the advantages of interactive semantic decoding, our method marginally surpasses UniAD. Our method mitigates the negative transfer in vectorized online mapping, significantly surpassing SparseDrive by 7.0% in mAP, (see Tab. 2).

3.3 Prediction and planning

Metrics. For motion prediction, we utilize E2E perception accuracy (EPA) proposed in ViP3D [16] as the main metric. We also report the minimum average displacement error (minADE). However, since minADE is a true positive metric, it does not fully capture the predictive capabilities of the E2E system, whereas EPA accounts for the number of false positives. For open-loop planning, we use L_2 distances and collision rates. Moreover, we evaluate driving safety in a closed-loop environment using NeuroNCAP [20]. This framework reconstructs scenes from the nuScenes dataset and inserts safety-critical objects. The resulting scores are derived from collision rates and impact speeds.

Trajectory prediction. We report car and pedestrian prediction metrics in Tab. 3. Our method surpasses both baselines in EPA, especially achieving improvements of 0.040 for cars and 0.055 for pedestrians over UniAD. However, our method does not improve the minADE of cars. One possible reason is that once detection performance exceeds a certain threshold, further detection improvements often come from reducing false negatives of challenging objects that are either distant or occluded. These hard-to-detect objects typically have limited historical motion data and larger coordinate errors, making them more difficult to predict. A similar issue is observed in UniAD [2]: in the supplementary materials, UniAD-Large substantially surpasses UniAD-Base in EPA (thanks to better detection and tracking performance), yet it falls short of UniAD-Base in minADE.

Planning. For open-loop evaluation, we adopt the evaluation method of VAD [3], which accommodates the widest range of models to our knowledge. We report our results in Tab. 4. Notably, jointly optimizing L_2 distances and collision rates proves challenging. While PARA-Drive achieves the lowest L_2 distances, it also exhibits the highest collision rates. In the closed-loop evaluation, our structure benefits both baselines in all three cases with stationary, frontal, and side critical objects.

Table 4: **Open-loop planning.** Ego-MLP and AD-MLP are faded since both learn only the ego motion. *Results from the checkpoint in the official repository [15], trained with a queue length of 5 in stage 1. †Ego-MLP employs a different strategy in the evaluation of collision rates, therefore the results are not comparable. We reproduce SparseDrive using the official code, but the results differ from its paper because some errors have been fixed after publication.

Method	Perception tasks	Ego states in planner	L_2 distances (m) ↓				Collision rates (%) ↓			
			1s	2s	3s	Avg.	1s	2s	3s	Avg.
Ego-MLP [21]	✗	✓	0.17	0.34	0.60	0.370	0†	0.27†	0.85†	0.373†
AD-MLP [22]	✗	✓	0.14	0.10	0.41	0.217	0.10	0.10	0.17	0.123
VAD [3]	✓	✗	0.41	0.70	1.05	0.720	0.07	0.17	0.41	0.217
DualVAD [6]	✓	✗	0.30	0.53	0.82	0.550	0.11	0.19	0.36	0.220
GenAD [4]	✓	✗	0.28	0.49	0.78	0.517	0.08	0.14	0.34	0.187
UniAD* [2]	✓	✗	0.42	0.63	0.91	0.656	0.07	0.10	0.22	0.130
PARA-Drive [8]	✓	✗	**0.25**	**0.46**	**0.74**	**0.483**	0.14	0.23	0.39	0.253
UniAD	✓	✗	0.48	0.76	1.12	0.784	0.07	0.11	0.27	0.150
DMAD	✓	✗	0.38	0.60	0.89	0.625	0.07	0.12	**0.19**	0.127
SparseDrive	✓	✗	0.32	0.61	1.00	0.643	0.01	**0.06**	0.22	0.097
SparseDMAD	✓	✗	0.30	0.61	1.01	0.643	**0**	0.07	0.21	**0.093**

Table 5: **Closed-loop planning.** We use the official implementation of NeuroNCAP, but our results differ from those in the original paper because the codebase has been updated since its publication.

Method	NeuroNCAP Scores ↑				Collision rates (%) ↓			
	Stat.	Frontal	Side	Avg.	Stat.	Frontal	Side	Avg.
UniAD	3.50	1.17	1.67	2.11	32.4	77.6	71.2	60.4
DMAD	4.40	1.47	2.07	2.65	**14.8**	74.0	61.6	50.1
SparseDrive	4.42	2.96	2.30	3.23	22.4	62.8	60.4	48.5
SparseDMAD	**4.57**	**3.14**	**2.42**	**3.37**	18.4	**60.0**	**59.1**	**45.8**

We validate that the improvements in perception can be propagated to planning, achieving SOTA collision rates and NeuroNCAP Scores.

3.4 Ablation study

We ablate our proposed decoders, as shown in Tab. 6, decomposing the motion decoder into three components: motion query, inter-layer, and inter-frame reference point updating.

Model profile. In methods with multi-view camera images as inputs, the primary computational cost is concentrated in the image backbone [18]. In contrast, our approach focuses on the decoding component, resulting in minimal impact on model size and inference speed. Compared to UniAD [2], our decoders add 13.1M parameters and increase inference latency by 0.02 seconds on an NVIDIA RTX 6000 Ada.

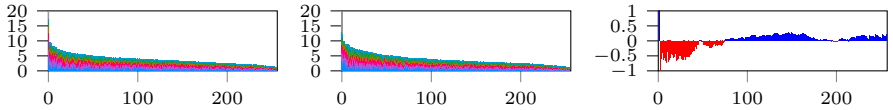
Effect of dividing and merging. Experiments ID 1, 2, 3, 7 demonstrate the effectiveness of both proposed decoders. The standalone application of the interactive semantic decoder (ID 2) significantly enhances the performance of object detection, tracking, and map segmentation. The standalone application of the Neural-Bayes motion decoder (ID 3) markedly improves prediction and planning. Notably, ID 3 also significantly enhances detection and tracking, attributed to freeing object queries from learning velocities and the higher-quality reference points provided by the unimodal prediction. Experiments ID 4, 5, 6, 7 show the importance of inter-layer and inter-frame updating in the Neural-Bayes motion decoder.

3.5 Visualizations

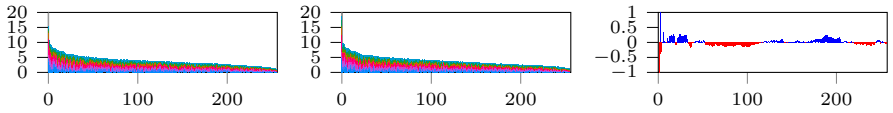
We use SHAP values [7]—which quantify the contribution of each feature to the change in a model's output—to inspect the negative transfer in detection and tracking. We visualize the SHAP values of the object query with respect to the object classification output. Changes in SHAP values across

Table 6: **Ablation of DMAD.**

Method ID	Interactive semantic dec.	Motion queries	Inter-layer ref. update	Inter-frame ref. update	#Params (M)	Inference time (s)	NDS↑	AMOTA↑	Lanes↑	EPA↑	Avg. L_2↓	Avg. Col.↓
1 (UniAD)	✗	✗	✗	✗	127.3	0.47	0.491	0.354	0.312	0.495	0.784	0.150
2	✓	✗	✗	✗	128.0	0.48	0.503	0.382	0.320	0.524	0.683	0.150
3	✗	✓	✓	✓	139.3	0.49	0.502	0.387	0.313	**0.535**	0.661	0.143
4	✓	✓	✗	✗	140.4	0.49	0.481	0.339	0.322	0.485	0.655	0.163
5	✓	✓	✓	✗	140.4	0.49	0.489	0.352	**0.323**	0.498	0.648	0.160
6	✓	✓	✗	✓	140.4	0.49	0.495	0.364	0.319	0.512	0.631	0.137
7 (DMAD)	✓	✓	✓	✓	140.4	0.49	**0.506**	**0.393**	0.321	**0.535**	**0.625**	**0.127**



(a) SHAP values of UniAD



(b) SHAP values of DMAD

Figure 5: **SHAP values of stage 1 (left), stage 2 (middle), and the difference (right).** Each bar represents the SHAP values of a single feature with respect to different classes. The object query consists of 256 features, forming 256 bars in each chart. The difference is computed as stage 1 minus stage 2, aggregating all classes, where red indicates a negative value and blue signifies a positive value.

the two training stages reveal the negative transfer in UniAD and highlight the effectiveness of our method.

Figure 5a compares the SHAP values between stage 1 and stage 2 of UniAD, sorted in descending order. The left half of the difference bar chart predominantly shows negative values, whereas the right half shows positive values. This indicates that SHAP values in stage 1 are more uniformly distributed, while those in stage 2 are more concentrated. Compared with a flat distribution, this concentration indicates that fewer features are contributing to the classification task, reducing detection and tracking performance. This observation aligns with our argument that during the second stage, object queries are expected to learn motion information, which does not benefit the perception task. Specifically, while the velocity learned in stage 1 is sufficient for tracking (predicting the next timestep), it is inadequate for the long-term prediction over 12 timesteps (6 seconds). Therefore, the object query is forced to learn more motion states that offer limited utility for identifying objects, interfering with the space for semantic information. In contrast, the SHAP values in DMAD maintain a similar distribution across both stages, as shown in Fig. 5b.

Beyond SHAP values, we provide qualitative comparisons between DMAD and UniAD in Appendix G.

4 Conclusion

In this work, we show that by decoupling semantic and motion learning, we eliminate the negative transfer that E2E training typically imposes on object and map perception. Besides, we leverage the correlation between semantic tasks to promote positive transfer during E2E training. We validate that our improvements in perception and prediction directly enhance planning performance, achieving SOTA collision rates. However, our approach cannot be applied to E2E methods that are without query propagation mechanism, *e.g.*, VAD [3]. Addressing this limitation can be our future work.

References

[1] Crawshaw, M. Multi-Task Learning with Deep Neural Networks: A Survey. *arXiv preprint arXiv:2009.09796*, 2020.

[2] Hu, Y., J. Yang, L. Chen, et al. Planning-oriented Autonomous Driving. In *CVPR*, pages 17853–17862. 2023.

[3] Jiang, B., S. Chen, Q. Xu, et al. VAD: Vectorized Scene Representation for Efficient Autonomous Driving. In *ICCV*, pages 8340–8350. 2023.

[4] Zheng, W., R. Song, X. Guo, et al. GenAD: Generative End-to-End Autonomous Driving. In *ECCV*, pages 87–104. 2025.

[5] Sun, W., X. Lin, Y. Shi, et al. SparseDrive: End-to-End Autonomous Driving via Sparse Scene Representation. *arXiv preprint arXiv:2405.19620*, 2024.

[6] Doll, S., N. Hanselmann, L. Schneider, et al. DualAD: Disentangling the Dynamic and Static World for End-to-End Driving. In *CVPR*, pages 14728–14737. 2024.

[7] Lundberg, S. M., S.-I. Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett, eds., *NeurIPS*, pages 4765–4774. 2017.

[8] Weng, X., B. Ivanovic, Y. Wang, et al. PARA-Drive: Parallelized Architecture for Real-time Autonomous Driving. In *CVPR*, pages 15449–15458. 2024.

[9] Zeng, W., W. Luo, S. Suo, et al. End-to-end Interpretable Neural Motion Planner. In *CVPR*, pages 8660–8669. 2019.

[10] Vaswani, A. Attention Is All You Need. In *NeurIPS*. 2017.

[11] Caesar, H., V. Bankiti, A. H. Lang, et al. nuScenes: A multimodal dataset for autonomous driving. In *CVPR*, pages 11621–11631. 2020.

[12] Thrun, S., W. Burgard, D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[13] Zeng, F., B. Dong, Y. Zhang, et al. MOTR: End-to-End Multiple-Object Tracking with Transformer. In *ECCV*, pages 659–675. 2022.

[14] Lin, X., Z. Pei, T. Lin, et al. Sparse4D v3: Advancing End-to-End 3D Detection and Tracking. *arXiv preprint arXiv:2311.11722*, 2023.

[15] UniAD-contributors. Planning-oriented Autonomous Driving. `https://github.com/OpenDriveLab/UniAD`, 2023.

[16] Gu, J., C. Hu, T. Zhang, et al. ViP3D: End-to-end Visual Trajectory Prediction via 3D Agent Queries. In *CVPR*, pages 5496–5506. 2023.

[17] Zhang, T., X. Chen, Y. Wang, et al. MUTR3D: A Multi-camera Tracking Framework via 3D-to-2D Queries. In *CVPR*, pages 4537–4546. 2022.

[18] Li, Z., W. Wang, H. Li, et al. BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers. In *ECCV*, pages 1–18. 2022.

[19] Liao, B., S. Chen, X. Wang, et al. MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction. In *ICLR*. 2023.

[20] Ljungbergh, W., A. Tonderski, J. Johnander, et al. Neuroncap: Photorealistic closed-loop safety testing for autonomous driving. In *ECCV*, pages 161–177. 2024.

[21] Zhai, J.-T., Z. Feng, J. Du, et al. Rethinking the Open-Loop Evaluation of End-to-End Autonomous Driving in nuScenes. *arXiv preprint arXiv:2305.10430*, 2023.

[22] Li, Z., Z. Yu, S. Lan, et al. Is Ego Status All You Need for Open-Loop End-to-End Autonomous Driving? In *CVPR*, pages 14864–14873. 2024.

[23] Carion, N., F. Massa, G. Synnaeve, et al. End-to-End Object Detection with Transformers. In *ECCV*, pages 213–229. 2020.

[24] Wang, Y., V. C. Guizilini, T. Zhang, et al. DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries. In *CoRL*, pages 180–191. 2022.

[25] Lin, X., T. Lin, Z. Pei, et al. Sparse4D: Multi-view 3D Object Detection with Sparse Spatial-Temporal Fusion. *arXiv preprint arXiv:2211.10581*, 2022.

[26] Liu, Y., T. Wang, X. Zhang, et al. PETR: Position Embedding Transformation for Multi-view 3D Object Detection. In *ECCV*, pages 531–548. 2022.

[27] Liu, Y., J. Yan, F. Jia, et al. PETRv2: A Unified Framework for 3D Perception from Multi-Camera Images. In *ICCV*, pages 3262–3272. 2023.

[28] Wang, S., Y. Liu, T. Wang, et al. Exploring Object-Centric Temporal Modeling for Efficient Multi-View 3D Object Detection. In *ICCV*, pages 3621–3631. 2023.

[29] Philion, J., S. Fidler. Lift, Splat, Shoot: Encoding Images from Arbitrary Camera Rigs by Implicitly Unprojecting to 3D. In *ECCV*, pages 194–210. 2020.

[30] Yang, C., Y. Chen, H. Tian, et al. BEVFormer v2: Adapting Modern Image Backbones to Bird's-Eye-View Recognition via Perspective Supervision. In *CVPR*, pages 17830–17839. 2023.

[31] Pan, C., B. Yaman, S. Velipasalar, et al. CLIP-BEVFormer: Enhancing Multi-View Image-Based BEV Detector with Ground Truth Flow. In *CVPR*, pages 15216–15225. 2024.

[32] Liao, B., S. Chen, Y. Zhang, et al. MapTRv2: An End-to-End Framework for Online Vectorized HD Map Construction. *IJCV*, pages 1–23, 2024.

[33] Meinhardt, T., A. Kirillov, L. Leal-Taixe, et al. TrackFormer: Multi-Object Tracking with Transformers. In *CVPR*, pages 8844–8854. 2022.

[34] Chen, J., Y. Wu, J. Tan, et al. MapTracker: Tracking with Strided Memory Fusion for Consistent Vector HD Mapping. In *ECCV*, pages 90–107. 2025.

[35] Chai, Y., B. Sapp, M. Bansal, et al. MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction. In *CoRL*. 2019.

[36] Cui, H., V. Radosavljevic, F.-C. Chou, et al. Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks. In *ICRA*, pages 2090–2096. 2019.

[37] Bansal, M., A. Krizhevsky, A. Ogale. ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst. In *Robotics: Science and Systems*. 2019.

[38] Gao, J., C. Sun, H. Zhao, et al. VectorNet: Encoding HD Maps and Agent Dynamics From Vectorized Representation. In *CVPR*, pages 11525–11533. 2020.

[39] Zhou, Z., L. Ye, J. Wang, et al. HiVT: Hierarchical Vector Transformer for Multi-Agent Motion Prediction. In *CVPR*, pages 8823–8833. 2022.

[40] Ngiam, J., B. Caine, V. Vasudevan, et al. Scene Transformer: A unified architecture for predicting future trajectories of multiple agents. In *ICLR*. 2022.

[41] Wagner, R., O. S. Tas, M. Klemp, et al. RedMotion: Motion Prediction via Redundancy Reduction. *Transactions on Machine Learning Research*, 2024.

[42] Shi, S., L. Jiang, D. Dai, et al. Motion Transformer with Global Intention Localization and Local Movement Refinement. *NeurIPS*, 35:6531–6543, 2022.

[43] Gu, J., C. Sun, H. Zhao. DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets. In *ICCV*, pages 15303–15312. 2021.

[44] Zhang, L., P. Li, S. Liu, et al. SIMPL: A Simple and Efficient Multi-agent Motion Prediction Baseline for Autonomous Driving. *IEEE Robotics and Automation Letters*, 2024.

[45] Bojarski, M. End to End Learning for Self-Driving Cars. *arXiv preprint arXiv:1604.07316*, 2016.

[46] Prakash, A., K. Chitta, A. Geiger. Multi-Modal Fusion Transformer for End-to-End Autonomous Driving. In *CVPR*, pages 7077–7087. 2021.

[47] Chen, D., P. Krähenbühl. Learning from All Vehicles. In *CVPR*, pages 17222–17231. 2022.

[48] Luo, W., B. Yang, R. Urtasun. Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net. In *CVPR*, pages 3569–3577. 2018.

[49] Casas, S., W. Luo, R. Urtasun. IntentNet: Learning to Predict Intention from Raw Sensor Data. In *CoRL*, pages 947–956. 2018.

[50] Djuric, N., H. Cui, Z. Su, et al. MultiXNet: Multiclass Multistage Multimodal Motion Prediction. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 435–442. 2021.

[51] Fadadu, S., S. Pandey, D. Hegde, et al. Multi-View Fusion of Sensor Data for Improved Perception and Prediction in Autonomous Driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2349–2357. 2022.

[52] Liang, M., B. Yang, W. Zeng, et al. PnPNet: End-to-End Perception and Prediction with Tracking in the Loop. In *CVPR*, pages 11553–11562. 2020.

[53] Weng, X., Y. Yuan, K. Kitani. PTP: Parallelized Tracking and Prediction with Graph Neural Networks and Diversity Sampling. *IEEE Robotics and Automation Letters*, 6(3):4640–4647, 2021.

[54] Chitta, K., A. Prakash, A. Geiger. NEAT: Neural Attention Fields for End-to-End Autonomous Driving. In *ICCV*, pages 15793–15803. 2021.

[55] Casas, S., A. Sadat, R. Urtasun. MP3: A Unified Model to Map, Perceive, Predict and Plan. In *CVPR*, pages 14403–14412. 2021.

[56] Hu, S., L. Chen, P. Wu, et al. ST-P3: End-to-end Vision-based Autonomous Driving via Spatial-Temporal Feature Learning. In *ECCV*, pages 533–549. 2022.

A Extended related work

Semantic learning. Semantic learning includes object detection and map segmentation. Multi-view cameras have become popular due to their cost-effectiveness and strong capability in capturing semantic information. Current SOTA object detection and mapping approaches are built on the DETR [23] architecture, utilizing a set of queries to extract semantic information from environment features through cross-attention [10] mechanisms. Sparse methods [24, 25] learn semantic information by projecting queries onto the corresponding image features, focusing on the relevant regions. The PETR series [26–28] embed 3D positional encoding directly into 2D image features, eliminating the need for query projection. Another line of work aggregates all image features into a BEV feature [29, 18, 30, 31, 19, 32]. Propagating the object queries over time enables multi-object tracking [13, 33]. This same technique is also used in map perception [34]. Although tracking is also a motion-related task, we classify it as a semantic task, as query-based trackers learn only velocities as the motion information, which we elaborate in Appendix B.

Motion learning. By motion, we refer to trajectory prediction and planning. Trajectory prediction studies typically use the ground truth of objects' historical trajectories along with high-definition maps as inputs. Early approaches [35–37] rasterize maps and trajectories into a BEV image, using CNNs to extract scene features. Vectorized methods [38, 39] represent elements using polygons and polylines, using GNNs or Transformers to encode the scene [40–44].

For planning, imitation learning is a straightforward approach to E2E planning, where a neural network is trained to plan future trajectories or control signals directly from sensor data, minimizing the distance between the planned path and the expert driving policy [45–47]. Many approaches incorporate semantic tasks as auxiliary components to support E2E planning, using the nuScenes [11] dataset and open-loop evaluation. These methods go beyond pure motion learning and are presented in the next paragraph. AD-MLP [21] and Ego-MLP [22] utilize only the ego vehicle's past motion states and surpass methods that rely on sensor inputs in open-loop evaluation. It aligns with our argument that semantics and motion are heterogeneous: AD-MLP and Ego-MLP can concentrate on learning from expert motion data without interference by irrelevant semantic information, thereby achieving superior open-loop planning performance.

Joint semantic and motion learning. E2E perception and prediction approaches learn semantics and motion jointly. The pioneering work FaF [48] uses a prediction head, in addition to the detection head, to decode the object features into future trajectories. Some works [49–51] enhance it with intention-based prediction and refinement. PnPNet [52] and PTP [53] involve tracking, *i.e.*, jointly optimizing detection, association, and prediction tasks. While PTP performs tracking and prediction in parallel, it cannot predict newly emerging objects due to the lack of concurrent detection—a limitation our method successfully overcomes. ViP3D [16] first extends the query-based detection and tracking framework [13] to prediction. Each query represents an object and propagates across frames. In each frame, queries are decoded into bounding boxes and trajectories using high-definition maps as additional context.

To include planning, NMP [9] extends IntentNet [49] with a sampling-based planning module, where prediction is leveraged to minimize collisions during the planning process. Other works [54–56] incorporate map perception as an auxiliary task. With the growing popularity of query-based object detectors [23, 18] and trackers [13, 33], recent modular E2E AD approaches represent objects as queries, similar to ViP3D [16]. UniAD [2] and its variants [6, 8] retain the query propagation mechanism for tracking, aiming to explicitly model objects' historical motion. In contrast, VAD [3] and GenAD [4] do not perform tracking, predicting trajectories based on the temporal information embedded within the BEV feature. The main issue with these methods is that they attempt to use a single feature (query) to represent an object's appearance and motion. Compared to pure semantic learning, motion occupies a portion of the feature channels but fails to contribute to perception, resulting a negative transfer in the perception module. Our work effectively addresses this issue.

B Tracking as a semantic task

We justify the similarity of detection and tracking on nuScenes [11] by analyzing the information learned by the object query. E2E detection and tracking models decode each query into category,

Table 7: **Effect of queue length on UniAD.**

Queue length stage 1	Queue length stage 2	NDS↑	mAP↑	AMOTA↑	AMOTP↓	IDS↓	Lanes↑	Drivable↑	EPA↑	minADE↓	Avg. L_2↓	Avg. Col.↓
3	3	0.491	0.377	0.354	1.34	1381	0.312	0.678	**0.495**	0.692	0.784	0.150
5	3	0.499	0.381	0.362	1.34	956	0.313	**0.692**	0.492	**0.655**	0.656	0.130
5	5	**0.501**	**0.384**	**0.370**	**1.32**	**885**	**0.314**	0.690	**0.495**	0.714	**0.615**	**0.123**

location, size, orientation, and velocity. The category is clearly a semantic attribute, while location, size, and orientation serve as spatial complements to the category, all being time-invariant. In contrast, velocity is derived from time, making it a motion attribute. However, measuring velocities is not a common practice in detection, but required by the nuScenes benchmark. Therefore, detection models trained on nuScenes are able to perform tracking without any additional learning effort assuming constant velocity motion [17, 2, 14, 16]. Given that current modular E2E models are all trained on nuScenes, we regard the tracking in these methods closely resembles detection, where learning semantics is dominating.

C Bayes filter

Bayes filter [12] estimates an unknown distribution based on the process model and noisy measurements as follows:

$$p(\mathbf{x}_t \mid \mathbf{z}_{1:t}) = p(\mathbf{z}_t \mid \mathbf{x}_t)\, p(\mathbf{x}_t \mid \mathbf{z}_{1:t-1}), \tag{3}$$

where $\mathbf{x}$ denotes the state, $\mathbf{z}$ represents the measurement, and the subscript indicates timesteps. The task is to estimate the state $\mathbf{x}_t$ at timestep t given all the measurements $\mathbf{z}_{1:t}$ in the past from timestep 1 to t, which is the product of the likelihood $p(\mathbf{z}_t \mid \mathbf{x}_t)$ and the prediction $p(\mathbf{x}_t \mid \mathbf{z}_{1:t-1})$.

Some special cases of Bayes filter, *e.g.*, Kalman filter, are widely used in traditional object tracking. The tracking process can be carried out in three steps: first, predicting the current position based on the object's historical states $\mathbf{x}_{1:t-1}$; second, identifying the detection most likely to match the prediction as the measurement; finally, updating the current state $\mathbf{x}_t$ according to the latest measurement $\mathbf{z}_{t-1}$. This process is recursively executed over successive timesteps. We find semantics and motion are similar to the measurement and state in Bayes filter, respectively. Therefore, we introduce the architecture of Bayes filter to transformer decoders, resulting in Neural-Bayes motion decoder.

D Occupancy prediction

We retain the occupancy prediction module from UniAD to ensure task consistency, where the BEV feature serves as the query and learns from motion prediction features (output queries) through cross-attention. Consequently, we regard occupancy prediction in UniAD as a secondary task to perception and motion prediction, as it merely offers an alternative representation of upstream tasks.

DMAD achieves similar performance (IoU_{near}: 62.7%, IoU_{far}: 39.8%) to UniAD (IoU_{near}: 62.9%, IoU_{far}: 39.6%). The advances of DMAD in upstream tasks do not generalize to occupancy prediction. The reason could be that, by dividing semantics and motion, output features of the prediction module lack spatial information desired by occupancy prediction, such as size, whereas output features of UniAD's prediction module preserve the spatial information.

E Queue length

We adopt a different queue length configuration from that of the original UniAD. As mentioned in Sec. 3.1, the rationale behind our decision is that reducing the queue length in stage 2 affects the performance, hindering the observation of negative transfer. Table 7 shows an ablation study of queue length on UniAD, presenting the performance drops by reduced queue length. As the training time scales almost linearly to the queue length, we opt for a queue length of 3 to reduce training time of each iteration.

Table 8: **Effect of unimodal prediction horizon on DMAD.**

Unimodal pred. horizon	NDS↑	mAP↑	AMOTA↑	AMOTP↓	IDS↓	Lanes↑	Drivable↑	EPA↑	minADE↓	Avg. L_2↓	Avg. Col.↓
2s	**0.516**	**0.404**	**0.400**	**1.30**	**695**	0.321	0.691	0.534	0.735	0.679	0.220
4s	0.506	0.396	0.393	**1.30**	767	0.321	0.691	**0.535**	**0.723**	**0.625**	0.127
6s	0.504	0.396	0.384	**1.30**	751	**0.322**	**0.700**	0.525	0.743	0.629	**0.117**

F Effect of unimodal prediction horizon

We conduct experiments on the number of future steps in unimodal prediction, as shown in Tab. 8. We observe that the unimodal prediction horizon influences the proportion of motion information within the BEV feature, thereby impacting the performance of both semantic and motion tasks. A long prediction horizon degrades the performance of semantic tasks, as the BEV feature is forced to prioritize motion learning in order to predict distant future outcomes. Experiments show that a prediction horizon of 6 seconds minimizes the collision rates, but performs worst in tracking. Although this phenomenon can also be referred to as negative transfer, our approach is unable to address this specific type, as the BEV feature is shared across all tasks and is expected to encapsulate both types of information. To balance motion and semantic information within the BEV feature, we set the prediction horizon to 4 seconds.

G Visualizations of reducing collisions rates

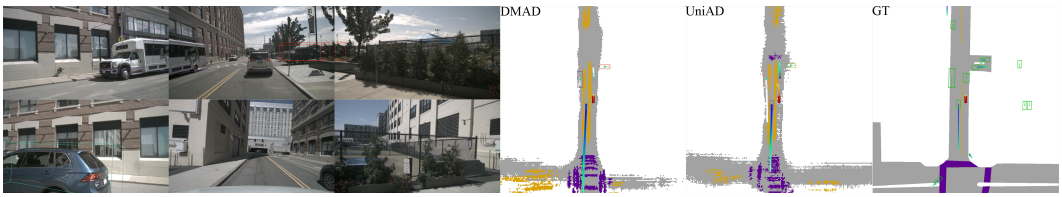
We provide qualitative comparisons between DMAD and UniAD in Fig. 6, showcasing how the improved perception and prediction reduces collision rates.

H Compute resources

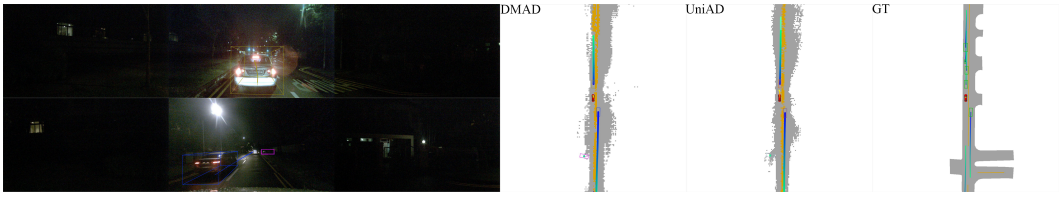
We use four NVIDIA A100 for our experiments. DMAD - stage 1 requires 44G GPU memory and 36 hours of training. DMAD - stage 2 requires 24G GPU memory and 84 hours of training. SparseDMAD - stage 1 requires 20G GPU memory and 32 hours of training. SparseDMAD - stage 2 requires 24G GPU memory and 6 hours of training. The full research work required more compute than the experiments reported in the paper.

I Potential societal impacts

This work on end-to-end autonomous driving aims to enhance traffic safety and efficiency by reducing human error. However, we also recognize significant potential risks, including reliability in edge cases, ethical dilemmas, cybersecurity threats, and socioeconomic impacts like job displacement.

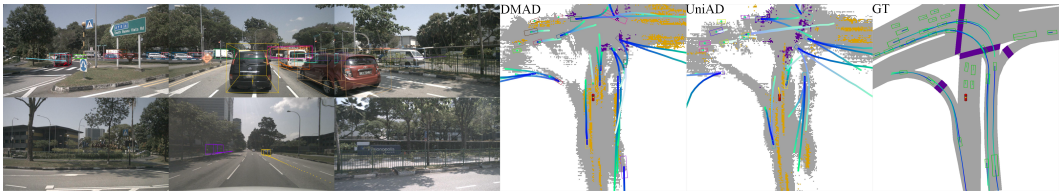


(a) The collision of UniAD is because of an inaccurate prediction of the lead vehicle.



(b) Both models make inaccurate predictions of the lead vehicle during the night. However, UniAD collides with the lead vehicle due to its aggressive driving policy.

(c) An inaccurate detection (the detected position is too close to the ego-vehicle) causes yielding, and then colliding with another vehicle.



(d) UniAD fails to detect the lead vehicle and collides with it.

Figure 6: **Qualitative comparison between DMAD and UniAD.** Each subfigure demonstrates a sample where UniAD encounters collision while DMAD does not.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims and contributions are explicitly stated in the abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Sec. 4.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: This work does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The DMAD structure is described in Sec. 2.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We will release the code along with the release of the paper.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: Both UniAD and SparseDrive are open-sourced, therefore we describe only differences of our experimental setup compared with the original setups in Section 3.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Error bars are not reported because it would be too computationally expensive.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix H.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Appendix I.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The model that we will be releasing is limited to tasks in nuScenes dataset, and therefore does not have a high risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The assets are properly cited, and the licenses are properly respected.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This paper does not include new assets. The code that we will release will include documentation.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.