GraphAD: Interaction Scene Graph for End-to-end Autonomous Driving

Yunpeng Zhang[1], Deheng Qian[1], Ding Li[1], Yifeng Pan[1], Yong Chen[2], Zhenbao Liang[2], Zhiyao Zhang[2], Shurui Zhang[2], Hongxu Li[2], Maolei Fu[2], Yun Ye[1], Zhujin Liang[1], Yi Shan[1], and Dalong Du[1⋆]

[1] PhiGent Robotics
[2] Geely Automobile Research Institute (Ningbo) Co., Ltd

Abstract. Modeling complicated interactions among the ego-vehicle, road agents, and map elements has been a crucial part for safety-critical autonomous driving. Previous works on end-to-end autonomous driving rely on the attention mechanism for handling heterogeneous interactions, which fails to capture the geometric priors and is also computationally intensive. In this paper, we propose the Interaction Scene Graph (ISG) as a unified method to model the interactions among the ego-vehicle, road agents, and map elements. With the representation of the ISG, the driving agents aggregate essential information from the most influential elements, including the road agents with potential collisions and the map elements to follow. Since a mass of unnecessary interactions are omitted, the more efficient scene-graph-based framework is able to focus on indispensable connections and leads to better performance. We evaluate the proposed method for end-to-end autonomous driving on the nuScenes dataset. Compared with strong baselines, our method significantly outperforms in the full-stack driving tasks, including perception, prediction, and planning. Code will be released at `https://github.com/zhangyp15/GraphAD`.

Keywords: End-to-end Autonomous Driving · Graph Neural Network

1 Introduction

The conventional Autonomous Driving (AD) system is manually divided into multiple sequential modules, including perception [13, 23, 27], prediction [8, 14], planning [10], and control [41]. However, the manual division prevents the system from being optimized jointly and globally, resulting in sub-optimal performance. To address this issue, end-to-end autonomous driving algorithms [12,16,48] optimize different modules altogether, making the whole system differentiable. With the potential of reducing accumulated errors and achieving higher performance, end-to-end algorithms are drawing increasing attention [4].

In end-to-end driving algorithms, both the prediction [5] and the planning [10, 46] modules share the same task of predicting future trajectories of agents (*i.e.*,

⋆ Corresponding author.

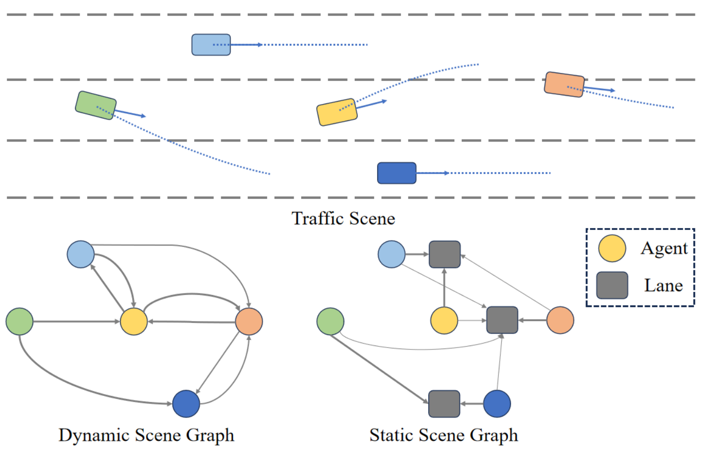

Fig. 1: The Interaction Scene Graph is composed of the Dynamic Scene Graph (DSG) and the Static Scene Graph (SSG). In DSG, the traffic agents, represented by the round nodes, pay attention to the surrounding agents by the directed connections. In SSG, the traffic agents reason about their trajectories based on the connected lanes which are represented by the rectangular nodes.

road agents in the prediction task and the ego vehicle in the planning task). The future trajectories are affected by the interactions among the agents and surrounding environments. Hence, modeling the interactions plays a central role in conventional end-to-end algorithms, which is commonly concreted by the attention mechanism [12,16]. However, the attention mechanism, mainly based on correlations of implicit features, lacks the prior knowledge of geometry about which driving elements are more important. As a result, the attention-based interactions inevitably waste their modeling capacities on the unimportant driving elements, while performing worse when being impaired by nuisance elements.

In this paper, we propose the Interaction Scene **Graph** for end-to-end **A**utonomous **D**riving (**GraphAD**) to enhance the interactions among driving elements. GraphAD encodes strong prior knowledge of the interactions into a graph model, the Interaction Scene Graph (ISG). The ISG is a directed graph model whose nodes represent key driving elements in the environment, including traffic agents and lanes. The driving elements are carefully selected for information aggregation, such that only important driving elements are represented. The directed edges in the graph represent the interactions between the nodes. Each node is linked to only a small number of other nodes, making the edges sparse. As a result, the ISG is a concise and efficient representation of the interactions.

Specifically, the ISG consists of two complementary parts, including the Dynamic Scene Graph (DSG) and the Static Scene Graphs (SSG), as shown in Fig. 1. The DSG focuses on the interactions among agents. Each node of DSG corresponds to an agent. Each edge has a weight, measuring the attention one agent pays to the other. The weights are employed to predict the future trajectories of the agents. Note that the interactions among agents depend on the future trajectories, *i.e.*, an agent would pay more attention to one another if their trajectories would collide in the future. Hence, the weights in DSG and the predicted future trajectories are interdependent. The predicted future trajectories can in turn refine the weights in the DSG. So we optimize the DSG and the predicted future trajectories iteratively. The SSG depicts the interaction between the agent and the surrounding map elements. Each agent is represented as a node in the SSG. In the meanwhile, the surrounding lanes in the map are also represented as nodes. The directed edges come from the agent and go to the lanes, modeling the attention the agents pay to the lanes. We apply graph neural networks [18, 36] on the DSG and SSG. The extracted features are utilized to predict the future trajectories of the ego and all other agents. By such means, we are able to adopt a unified method to accomplish both the prediction and the planning tasks.

We evaluate our method on the nuScenes dataset [1]. Extensive ablation studies are conducted to demonstrate the effectiveness of our design choices. We summarize our main contributions as follows:

- To our knowledge, GraphAD is the first end-to-end autonomous algorithm which employs a graph model to describe the complex interactions in traffic scenes. The graph model allows us to introduce strong prior knowledge of the traffic scene into the algorithm effectively and efficiently.
- We elaborately devise the ISG which concisely presents the heterogeneous interactions among ego vehicle, traffic agents, and map elements. In particular, the DSG is able to iteratively refine the prediction of future trajectories and describe subtle interactive games among agents.
- When compared with strong baselines [11,12,16], our method achieves state-of-the-art performance on multiple tasks.

2 Related Work

2.1 End-to-end Autonomous Driving.

Instead of adopting a modular paradigm in the traditional AD framework, end-to-end methods, which aim to output future actions based on sensor inputs, have attracted considerable attention. When formulated in an end-to-end manner, the whole framework can be optimized towards the ultimate planning task with high computational efficiency [4,12]. Some pioneering approaches attempt to directly predict the planned trajectory while lacking explicit supervision of intermediate perception and prediction tasks [15, 32, 40, 41, 48]. Considering the transparency and interpretability for safety, recent works [2,3,11,12,16,45] introduce requisite preceding tasks in the end-to-end framework, thus unifying perception, prediction, and planning into a holistic model. For instance, UniAD [12], which regards task-specific queries as a powerful tool for message passing throughout the AD pipeline, has achieved remarkable performance in both multi-object tracking, online mapping, motion forecasting, occupancy prediction, and planning. FusionAD [45] extends the capacity of UniAD [12] with multi-modal input. In the meantime, some researchers focus on the impact of different privileged inputs. VAD [16] contends that end-to-end AD can be performed in a fully vectorized manner with high efficiency, while OccNet [35] attempts to perform the planning task based on the predicted occupancy.

Despite previous methods that have gained impressive performance, the interactions between traffic agents and the surrounding environment are not fully explored. In this work, we propose the Interaction Scene Graph to explicitly model the heterogeneous interactions between the dynamic and static driving elements.

2.2 Graph Neural Networks.

Thanks to the success of Graph Neural Networks in graph data, GNNs [6,18,36] have been widely adopted in various fields, such as object detection [34,37,42], skeleton-based action recognition [20, 43], person re-identification [44]. Also, GNN-related advances attract researchers in the autonomous driving community, several studies propose to leverage the ability of GNNs for scene perception and motion prediction. GNN3DMOT [38] and PTP [39] attempt to model inherit interactions among detected targets for 3D multi-object tracking. For online mapping, LaneGCN [24] constructs a lane graph from an HD map, and TopoNet [21] introduces relation modeling among lane and traffic elements with a learned scene knowledge graph. In multi-agent motion forecasting, both moving agents and map elements are designed as nodes in graph construction, and the introduction of the relationship among them would benefit trajectory prediction [30,33]. HDGT [14] devises a heterogeneous graph and explicitly models all semantics and relations in the scene. Different from prior works, GraphAD is the first to capture the interactions among dynamic agents and static map elements in the end-to-end AD framework. Also, GraphAD proposes to consider the potential movements of dynamic agents in graph construction by introducing the trajectory proposals.

3 Method

The overall framework of GraphAD is presented in Fig. 2. First, with multi-view video sequences, camera parameters, and ego-poses as input, the image features are extracted by the image encoder and then lifted to the Bird-Eye-View (BEV) features. The multi-frame BEV features are further aggregated to form the spatiotemporal scene representation. Second, GraphAD employs two transformer decoders, *i.e.* the TrackFormer and the MapFormer, to extract the structured representations for the dynamic and static driving elements. Third, the Interaction Scene Graph is explicitly constructed to model the interactions among the ego vehicle, dynamic elements, and static elements, by considering the potential movements. Finally, the graph-aggregated ego query feature, combined with ego status features and high-level driving command, is processed by the planning head to predict the ego-vehicle trajectory. We elaborate on the designs of these steps in the following sections.

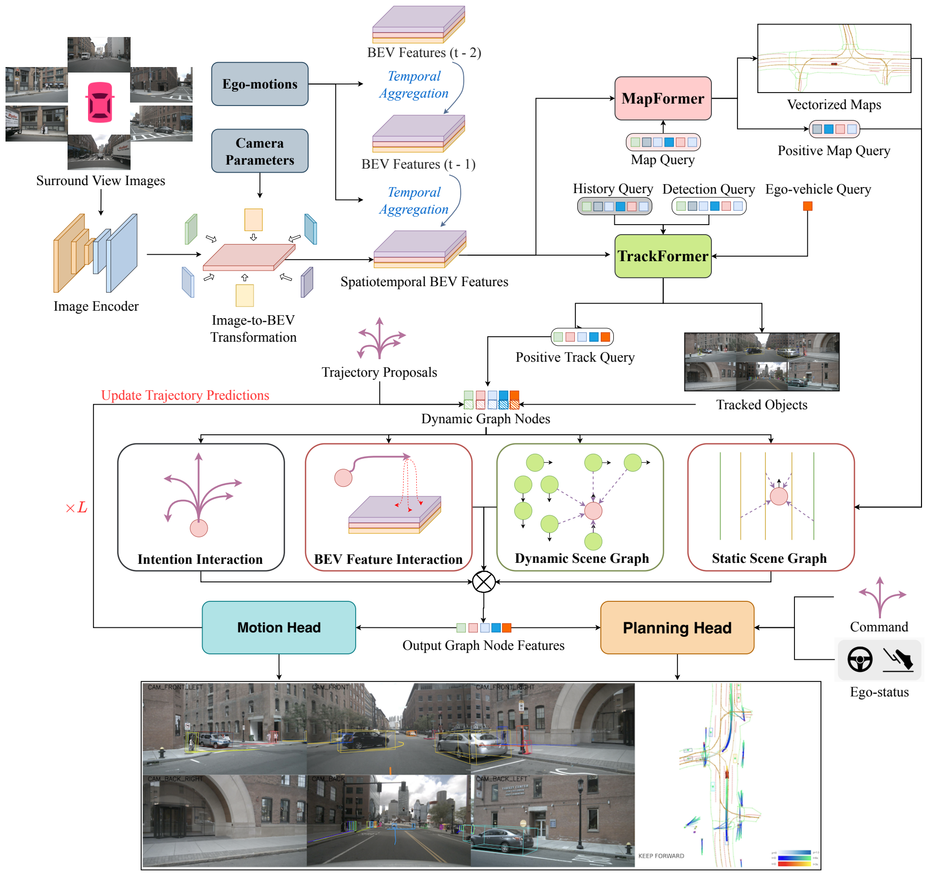

Fig. 2: GraphAD features the graph-based interactions between the structured instances in the driving environment, including the dynamic traffic agents and the static map elements. GraphAD first constructs the spatiotemporal scene feature on the Bird-Eye-View as the unified representation for downstream tasks. Then, GraphAD extracts the structured instances by the TrackFormer and the MapFormer. Taking these instances as graph nodes, GraphAD proposes the Interaction Scene Graph to iteratively refine the features of dynamic nodes, by considering the inter-agent and agent-map interactions. Finally, the processed node features are utilized for motion prediction and end-to-end planning.

3.1 Spatiotemporal Scene Representation

Image Encoder. The image encoder includes a backbone network for multi-scale feature extraction and a neck for fusing these features. Formally, with the multi-view images $I \in \mathbb{R}^{N\times 3\times H_I\times W_I}$ as input, the image encoder creates the extracted visual features $\mathbf{F}_{2d} \in \mathbb{R}^{N\times C_I\times H_I'\times W_I'}$, where N is the number of camera views, C_I is the channel number, (H_I, W_I) and (H_I', W_I') are the input and downsampled image sizes. The output visual feature can contain the fundamental semantics and geometry of the surrounding environment.

Image-to-BEV Transformation. To build a unified scene representation for temporal aggregation and multi-task inference, we lift the multi-view image features to BEV representations with the Lift-Splat-Shoot paradigm [13, 22, 31]. Specifically, the image features $\mathbf{F}_{2d}^t$ at time t are processed to create the context features $\mathbf{F}_{con}^t \in \mathbb{R}^{N\times C\times H'_I\times W'_I}$ and categorical depth distributions $\mathbf{D}^t \in \mathbb{R}^{N\times D\times H'_I\times W'_I}$, where C is the channel number and D is the number of depth bins. The outer product $\mathbf{F}_{con}^t \otimes \mathbf{D}^t$ is then computed as lifted feature point cloud $\mathbf{P}^t \in \mathbb{R}^{NDH'_IW'_I\times C}$. Finally, the voxel-pooling is employed to process the feature points and generate the BEV feature $\mathbf{F}_{BEV}^t \in \mathbb{R}^{C\times H\times W}$ at time t.

Temporal Feature Aggregation. The multi-frame BEV features $\{\mathbf{F}_{BEV}^t\}_{t=t_{cur}-T+1}^{t_{cur}}$, where t_{cur} is the current time and $T \in \mathbb{N}_+$ is the number of frames, are first warped into the ego-centric coordinate system at the current time so that the ego-motion misalignment is removed. Afterward, the aligned multi-frame BEV features are concatenated along the channel dimension and further processed by a convolutional BEV encoder. The output spatiotemporal BEV feature $\mathbf{F}_{BEV} \in \mathbb{R}^{C_o\times H\times W}$ will serve as the unified spatiotemporal scene representation for downstream tasks.

3.2 Structured Element Learning

Based on the spatiotemporal scene features, the extraction of structured elements, including traffic agents and map elements, is important for safety-critical planning in autonomous driving. Therefore, GraphAD utilizes the TrackFormer and the MapFormer to predict these driving-related instances.

TrackFormer. With the spatiotemporal BEV representation, the TrackFormer aims to perform end-to-end 3D object detection and tracking. Following the design of [12], we employ two groups of object queries and the transformer decoder to solve the problem. Specifically, one group of track queries, which corresponds to previously detected objects, is still required to predict the updated 3D bounding boxes of the same object identities. The other group of detection queries is responsible for the objects which are visible for the first time. For each timestamp, the positive queries, including the tracked and newborn, will serve as the track queries for the next timestamp. The transformer decoder layer includes the self-attention between all object queries and the deformable attention for attending to the spatiotemporal BEV features.

MapFormer. To better capture the geometric constraints of map elements, we follow recent practices [16, 26] to learn vectorized representations of local maps. Specifically, the MapFormer utilizes the instance-level and point-level queries to form the hierarchical map queries, which are processed by a similar transformer decoder as in TrackFormer. Finally, the output map queries are projected to the class scores and a series of BEV coordinates of potential map elements. To fully capture the map information, four kinds of elements are modeled, including the lane centerline, lane divider, road boundary, and pedestrian crossing.

3.3 Interaction Scene Graph

With the extracted driving instances in structured formats, including traffic agents and map elements, the key challenge lies in how the network can perceive heterogeneous interactions. These interactions, including the driving game between dynamic agents, or the simple centerline-following heuristics, are important for forecasting the future of the surrounding environment and making driving decisions. To this end, we construct the Interaction Scene Graph to capture these heterogeneous interactions. As an iterative process, the Interaction Scene Graph functions in three steps. First, all dynamic and static elements are formulated as graph node representations, including explicit geometry and implicit features. Second, the Interaction Scene Graph is constructed based on strong geometric priors. Third, the graph node features are updated based on the established graph edges, which are further processed to update the geometry. The detailed formulation is elaborated in the following paragraphs.

Graph Node Representation. The Interaction Scene Graph is constructed on the structured nodes of traffic agents and map elements. Each graph node is designed to include both the explicit geometry and the implicit features. Note that the ego-vehicle is treated as one of the traffic agents to participate in graph-based interactions.

Specifically, the graph nodes of traffic agents, *i.e.* dynamic graph nodes, are organized as one set $P^d = \{p_1^d, \dots, p_{N_d}^d\}$, where N_d is the number of dynamic graph nodes. Also, $p_i^d = (\mathbf{x}_i^d, \mathbf{f}_i^d)$ represents the node representation with its trajectory proposal $\mathbf{x}_i^d \in \mathbb{R}^{M_d \times 2}$ as BEV coordinates and its node feature $\mathbf{f}_i^d \in \mathbb{R}^{C_g}$ with C_g channels, where M_d is the time horizon of trajectory prediction. The trajectory proposals are the trajectory predictions from the previous layer. For the first layer, the clustering results from k-means are utilized instead. The implicit node features are computed as the combination of previous node features, queries from the TrackFormer, embeddings of trajectory proposals, and learnable intention embeddings, following [12]. For the unified formulation, we treat different modalities of the same agent as different dynamic graph nodes.

Similarly, the graph nodes of map elements, *i.e.* static graph nodes, are organized as the other set $P^s = \{p_1^s, \dots, p_{N_s}^s\}$, where N_s is the number of static graph nodes and $p_i^s = (\mathbf{x}_i^s, \mathbf{f}_i^s)$ represents one map element by a series of BEV coordinates $\mathbf{x}_i^s \in \mathbb{R}^{M_s \times 2}$ with M_s points and its node feature $\mathbf{f}_i^s \in \mathbb{R}^{C_g}$ with C_g channels. The structured predictions from the MapFormer, including the BEV coordinates and the output query features, are directly utilized as the static graph nodes. Since the map elements in the driving scenes usually serve as constant environment constraints, their node features are not updated in the iterative layers.

Graph Connection Construction. To capture the heterogeneous interactions between all graph nodes, the Interaction Scene Graph consists of the Dynamic Scene Graph (DSG) and the Static Scene Graph (SSG). The Dynamic Scene Graph is formulated as $G^d = (P^d, E^d)$ by using the traffic agents as dynamic graph nodes, which intends to model the driving game between these agents. The Static Scene Graph is formulated as $G^s = (P^d, P^s, E^s)$ by incorporating both the dynamic and static graph nodes, which focuses on providing the appropriate map information for the dynamic agents. For both DSG and SSG, we follow the same high-level philosophy for computing the edge connections. Specifically, we compute the pair-wise distances between graph nodes and connect each node to its K nearest neighbors. Despite the straightforward formulation, the design choices of pair-wise distance functions are still underexplored.

Existing graph-based methods [14, 37] usually exploit the pair-wise distance in feature or coordinate spaces. However, the heterogeneous and evolutionary interactions in the constructed scene graph, with dynamic agents and map elements, cannot be well processed by existing approaches. To this end, we propose to utilize the geometric distances based on trajectory proposals to measure the correlations between graph nodes. On the Dynamic Scene Graph, the distance $\mathcal{H}^d(p_i^d, p_j^d)$ between two dynamic graph nodes is computed as the minimal distance between their trajectory proposals at each time, as in Eq. (1):

$$\mathcal{H}^d(p_i^d, p_j^d) = \min_{t=1}^{M_d} \|\mathbf{x}_i^d(t) - \mathbf{x}_j^d(t)\|_2, \tag{1}$$

where $\mathbf{x}_i^d(t)$ refers to the predicted future position at time t. On the Static Scene Graph, the distance $\mathcal{H}^s(p_i^d, p_j^s)$ between a dynamic node and a static node is computed as the minimal distance between the dynamic trajectory proposal and the static map coordinates, as in Eq. (2):

$$\mathcal{H}^s(p_i^d, p_j^s) = \min_{t=1}^{M_d} \left(\min_{k=1}^{M_s} \|\mathbf{x}_i^d(t) - \mathbf{x}_j^s(k)\|_2 \right), \tag{2}$$

where $\mathbf{x}_i^d(t)$ refers to the predicted future position at time t and $\mathbf{x}_j^s(k)$ refers to the k-th coordinate point of the predicted map element. When the pair-wise distances are computed, the nearest K graph nodes with minimal distances are selected as the graph neighbors.

Graph Feature Aggregation. Since the interaction connections have been established, the final part is to refine the node feature by aggregating the information from its connected neighbors. A simple yet effective approach is proposed for the feature aggregation in the Interaction Scene Graph. Specifically, the feature of each neighbor node is concatenated with the target node and then processed by a Multi-Layer Perceptron (MLP). Finally, the permutation-invariant max-pooling is employed to aggregate the processed neighbor features into the target node. Also, the Dynamic Scene Graph and Static Scene Graph share the same approach for graph feature aggregation. At the end of each iteration layer, the updated features of dynamic agents are utilized to predict their multi-modal trajectories, including the probability score and the trajectory points for each modality. The predicted trajectory points are further used to update the geometric node features into the next iteration layer.

3.4 Planning Head

Planning Head Structure. The input information for the planning head includes the high-level driving command, the ego-status features, and the processed ego-query from the Interaction Scene Graph. The three groups of features are concatenated and processed by a simple MLP for the final planning predictions.

Ego-status Features. The ego-status information, which mainly includes the velocity, acceleration, and angular velocity, is important for the open-loop planning performance. Therefore, we use a small Multi-Layer Perceptron (MLP) to encode the ego-status information, along with the history trajectories of the ego-vehicle, into the ego-status features.

Occupancy-based Post-optimization. To further avoid the collision with other road agents and ensure the driving safety, we follow the implementation of UniAD [12] to train an occupancy head, whose predictions can be utilized to post-optimize the predicted planning trajectories.

3.5 Training

Loss Functions. The loss functions include the depth estimation loss $\mathcal{L}_{depth}$, the TrackFormer loss $\mathcal{L}_{track}$, the MapFormer loss $\mathcal{L}_{map}$, the motion trajectory loss $\mathcal{L}_{motion}$, the occupancy loss $\mathcal{L}_{occ}$, and the planning loss $\mathcal{L}_{plan}$. GraphAD is end-to-end trained with the summation of multi-task losses:

$$\mathcal{L} = \mathcal{L}_{depth} + \mathcal{L}_{track} + \mathcal{L}_{map} + \mathcal{L}_{motion} + \mathcal{L}_{occ} + \mathcal{L}_{plan}. \tag{3}$$

Specifically, we use binary cross-entropy for $\mathcal{L}_{depth}$ and follow existing methods [12,16] for training other tasks.

Multi-stage Training. With only the image backbone network initialized from ImageNet [19]-pretrained weights, GraphAD is trained in three stages. First, GraphAD is trained to jointly predict the 3D object detection and vectorized map elements. Second, we freeze the image backbone and train GraphAD for tracking, vectorized map, and graph-based motion prediction. Finally, we further add the tasks of occupancy prediction and planning for end-to-end training.

4 Experiments

Our experiments are conducted on the challenging nuScenes dataset [1], where 1000 complex driving scenes are included, and each scene roughly lasts for 20 seconds. In data collection, six cameras with various views are utilized for capturing the driving scene, thus covering 360° FOV horizontally. For annotations, over 1.4M 3D bounding boxes of 23 categories are provided in total, and the key-frames are annotated at 2 Hz.

4.1 Implementation Details

For benchmark results, GraphAD adopts the input size of 640 and ResNet101-DCN [7] as the image backbone. The image neck generates feature maps with 512 channels and 16× downsampling. For image-to-BEV transformation, GraphAD uses the method in BEVDepth [22] to generate the BEV features with 80 channels. Four frames of BEV features are fused to create the spatiotemporal scene representation $\mathbf{F}_{BEV} \in \mathbb{R}^{256\times200\times200}$. The TrackFormer strictly follows the settings of UniAD [12], while the MapFormer uses 100 map queries and a six-layer transformer decoder. The Interaction Scene Graph stacks three iterative layers for motion prediction with six modalities. The number of neighbours is set to 24 for the Dynamic Scene Graph and 8 for the Static Scene Graph. For ego-status features in the planning head, we follow the preprocessing of CAN-bus information from VAD [16]. For ablation studies, we adopt the input size of 256×704 and ResNet50 as the image backbone.

4.2 Metrics

We follow the same evaluation protocol of previous state-of-the-art method UniAD [12]. Specifically in tracking task, AMOTA and AMOTP are introduced to evaluate the perception performance. For motion prediction task, we employ widely-used metrics to evaluate the capability of our model, including End-to-end Prediction Accuracy (EPA), Average Displacement Error (ADE), Final Displacement Error (FDE), and Miss Rate (MR). In the evaluation of planning, Displacement Error (DE, L2 distance) and Collision Rate (CR) are commonly used to evaluate the planning performance, where the collision rate is considered as the main metric. Specifically, we follow UniAD to calculate DE and CR values at each planning step.

Table 1: Benchmark results for open-loop planning performance. † denotes LiDAR-based methods. $*$ represents the reproduced results with official checkpoints. GraphAD achieves the state-of-the-art planning performance.

Method	L2 (m) ↓				Collision (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.
NMP† [46]	-	-	2.31	-	-	-	1.92	-
SA-NMP† [46]	-	-	2.05	-	-	-	1.59	-
FF† [10]	0.55	1.20	2.54	1.43	0.06	0.17	1.07	0.43
EO† [17]	0.67	1.36	2.78	1.60	0.04	0.09	0.88	0.33
ST-P3 [11]	1.33	2.11	2.90	2.11	0.23	0.62	1.27	0.71
UniAD [12]	0.48	0.96	1.65	1.03	0.05	0.17	0.71	0.31
VAD* [16]	0.54	1.15	1.98	1.22	0.00	0.33	1.07	0.47
GPT-Driver [28]	0.27	0.74	1.52	0.84	0.07	0.15	1.10	0.44
Agent-Driver [29]	**0.22**	0.65	1.34	0.74	**0.02**	0.13	0.48	0.21
GraphAD	0.32	**0.61**	**1.10**	**0.68**	0.03	**0.07**	**0.25**	**0.12**

4.3 Benchmark Results

Planning Results. As shown in Tab. 1, GraphAD achieves the state-of-the-art performance for open-loop planning on the nuScenes validation set. When compared to the second best method, Agent-Driver [29], GraphAD achieves a 42.9% reduction of collision rate, which demonstrates the effectiveness of the proposed Interaction Scene Graph for aggregating information from related traffic agents and map elements.

Prediction Results. The benchmark results for motion prediction on the nuScenes validation set are summarized in Tab. 2. GraphAD achieves the best performance with 0.68 minADE and 0.514 EPA, significantly outperforming the previous best method UniAD [12]. The improved performance on motion prediction validates the enhanced capacity of Interaction Scene Graph in modeling the map guidance and intention interaction from other driving instances.

Table 2: Benchmark results for motion-forecasting.

Method	minADE(m)↓	minFDE(m)↓	MR↓	EPA↑
Constant Pos.	5.80	10.27	0.347	-
Constant Vel.	2.13	4.01	0.318	-
PnPNet [25]	1.15	1.95	0.226	0.222
ViP3D [5]	2.05	2.84	0.246	0.226
UniAD [12]	0.71	1.02	**0.151**	0.456
GraphAD	**0.68**	**0.98**	0.161	**0.514**

Perception Results. In Tab. 3, GraphAD achieves significant improvements over the existing state-of-the-art methods, including UniAD and MUTR3D. Benefits from the reliable perception results, the downstream tasks would have more potential to obtain accurate motion forecasting and planning results.

Table 3: Benchmark results for multi-object tracking.

Method	AMOTA↑	AMOTP↓	Recall↑	IDS↓
ViP3D [5]	0.217	1.625	0.363	-
QD3DT [9]	0.242	1.518	0.399	-
MUTR3D [47]	0.294	1.498	0.427	3822
UniAD [12]	0.359	1.320	0.467	906
GraphAD	**0.397**	**1.267**	**0.486**	**497**

4.4 Ablation Studies

To demonstrate the effectiveness of the proposed Interaction Scene Graph, we conduct extensive ablation studies on the nuScenes validation set.

Table 4: The ablation studies for the Interaction Scene Graph.

DSG	SSG	minADE(m)↓	minFDE(m)↓	MR↓
✓		0.683	1.014	0.165
	✓	0.684	1.018	0.167
✓	✓	**0.665**	**0.989**	**0.160**
Attention	Attention	0.678	1.000	**0.160**

Effectiveness of Interaction Scene Graph. In Tab. 4, we ablate the influence of Dynamic Scene Graph (DSG) and Static Scene Graph (SSG) on the motion prediction of traffic agents. We can observe that both types of scene graphs make significant contribution to the performance boost. Since the DSG can model the driving game between dynamic agents and the SSG is able to provide explicit map constraints, both types of graph-based interactions can provide valuable and complementary information for the trajectory prediction. For comprehensive evaluation, we also implement an attention-based variant, where the inter-agent and agent-map interactions are entirely realized by the vanilla attention mechanism. However, we find the attention-based variant, without explicit geometric prior, fails to extract valid information and generates inferior performance.

Table 5: The ablation studies for the choices of node similarity functions.

Similarity	minADE(m)↓	minFDE(m)↓	MR↓
Feature Distance	0.673	0.993	**0.160**
Current Distance	0.677	0.999	**0.160**
Trajectory Distance	**0.665**	**0.989**	**0.160**

Design choices of graph node distance. In Tab. 5, we analyze the influence of different methods for computing the distance between graph nodes. "Feature Distance" and "Current Distance" denote distances in the feature space and distances between current locations respectively, while "Trajectory Distance" is the distance between potential trajectories. Since the distance function directly determines which neighbour nodes will participate in the feature aggregation, its design choice is of vital importance. From the experimental results, we can find that the proposed trajectory distance significantly outperforms the current distance because it explicitly considers the potential interactions in the future, which is crucial for accurate trajectory estimation. On the other hand, the geometric distance on trajectories also outperforms the feature distance. It is possibly because the graph nodes, including both traffic agents and map elements, with different sources and modalities have heterogeneous features.

Table 6: The ablation studies for the graph feature aggregation methods.

Method	minADE(m)↓	minFDE(m)↓	MR↓
Attention	0.682	1.017	0.170
MLP + Avg-pooling	0.680	1.014	0.164
MLP + Max-pooling	**0.665**	**0.989**	**0.160**

Design choices of methods for graph feature aggregation. In Tab. 6, we compare different methods for aggregating the neighbour node features to update the vertex. As observed in the table, MLP-based aggregation methods performs better than attention-based methods. Furthermore, the max-pooling operation outperforms the avg-pooling method, reaching 0.665m minADE, 0.989m minFDE and 0.160 MR. Thus, we choose MLP with max-pooling as default setting.

Design choices of planning head. In Tab. 7, we explore the effects of different components for the planning task, where "Graph" refers to the proposed In-

Table 7: The ablation studies for designs in the planning head.

Graph	Ego-states	Post-optim.	Planning	
			L2 (m) ↓	Col. (%) ↓
			1.39	1.13
✓			1.35	1.07
	✓		0.65	0.63
✓	✓		**0.64**	0.47
✓	✓	✓	0.73	**0.15**
	✓	✓	0.74	0.22

teraction Scene Graph, "Ego-states" means the utilization of ego-vehicle status, and "Post-optim." represents the optimization strategy with the predicted occupancy. The following effects can be observed: (1) The incorporation of ego-state features can bring a significant improvement on the planning performance, since the information, like velocity and acceleration, makes it much easier to recover the ego-trajectory. (2) Whether or not the ego-state features are utilized, the proposed method of Interaction Scene Graph consistently improves the planning performance. (3) The post-optimization with the predicted occupancy plays an important role in ensuring the driving safety, by avoiding the potential collisions with explicit adjustments. With all above components, GraphAD, with smaller input sizes and image backbone, achieves a remarkable collision rate of 0.15%.

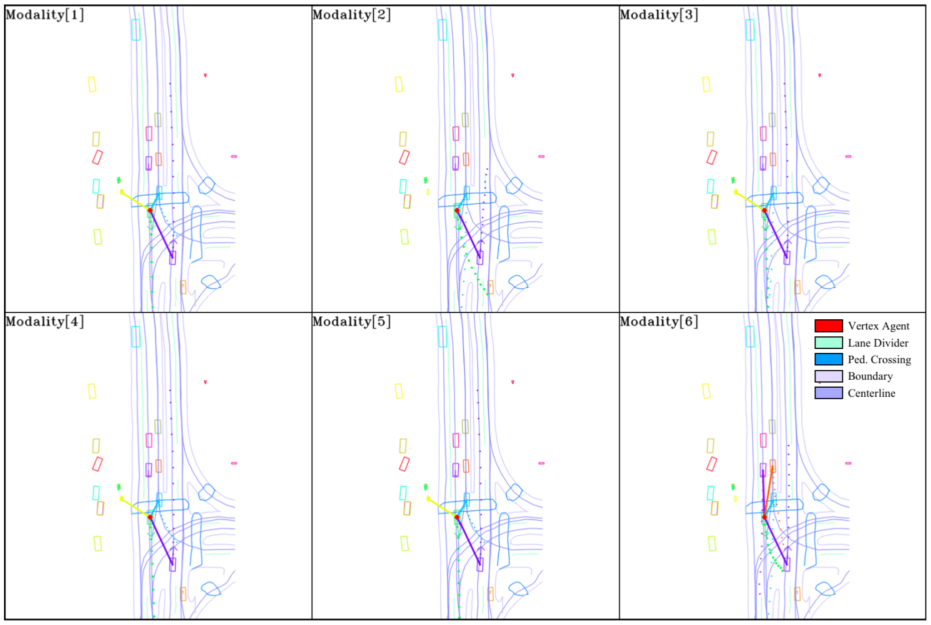

Fig. 3: The qualitative visualization of the Dynamic Scene Graph. The agent of interest, marked by the red dot, has 6 different modalities of future trajectories. With each motion intention, this agent interacts with the most influential traffic agents, which are denoted by the connections. Faraway connections are omitted for clarity.

4.5 Qualitative Results

To qualitatively evaluate our method for better understanding, we visualize both the intermediate interactions and the final results of GraphAD. As shown in Fig. 3, the agent of interest has 6 predicted future trajectories for different potential intentions. The dynamic scene graph for each trajectory automatically links the agent to other traffic agents nearby. With these explicit geometry priors, the agent can focus on the interactions with the important agents. From the cases in Fig. 4, GraphAD enables the ego vehicle to maneuver safely in complex situations like road junction and opposite meeting. These planning abilities results from the accurate motion prediction and necessary inter-agent interactions, based on the proposed graph designs.

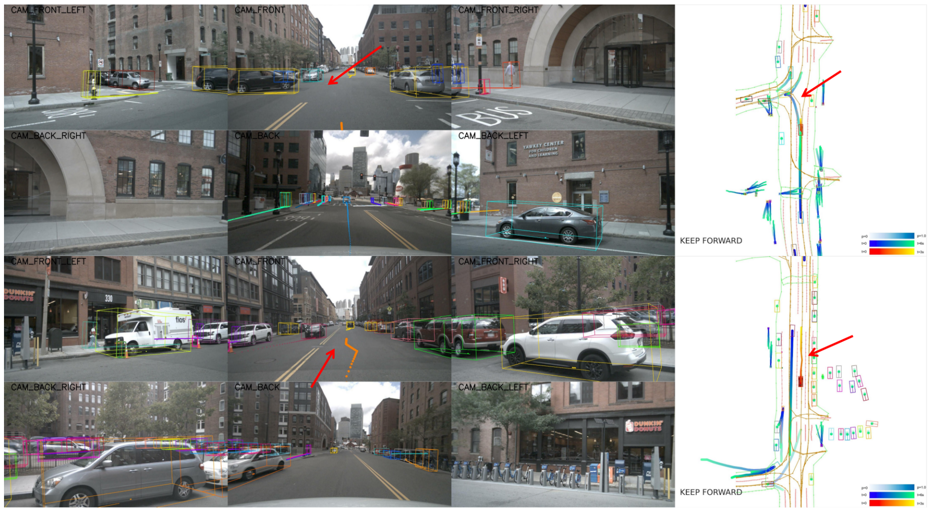

Fig. 4: The qualitative visualization of the planning trajectories. The images from six cameras are shown on the left. The predicted trajectories of traffic agents and the planning result of the ego vehicle are shown on the right. The color intensities of these trajectories vary according to the probability p and the time t. The red arrows highlight the environments which most likely influence the ego vehicle planning.

5 Conclusion

In this paper, we propose a new end-to-end autonomous driving algorithm, GraphAD, which employs an elaborately designed graph to describe heterogeneous interactions in complex traffic scenes. The graph explicitly encodes key driving elements and their relations, allowing us to introduce strong prior knowledge into the algorithm. As a consequence, GraphAD achieves state-of-the-art performance in both the prediction and the planning tasks. The way using graphs to encode more complex interactions among diverse traffic instances, such as traffic lights and routing decisions, needs further exploration.

References

1. 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: CVPR (2020)
2. Casas, S., Sadat, A., Urtasun, R.: Mp3: A unified model to map, perceive, predict and plan. In: CVPR (2021)
3. Chen, D., Krahenbuhl, P.: Learning from all vehicles. In: CVPR (2022)
4. Chen, L., Wu, P., Chitta, K., Jaeger, B., Geiger, A., Li, H.: End-to-end autonomous driving: Challenges and frontiers. arXiv preprint arXiv:2306.16927 (2023)
5. 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: CVPR (2023)
6. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NeurIPS (2017)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
8. Hu, A., Murez, Z., Mohan, N., Dudas, S., Hawke, J., Badrinarayanan, V., Cipolla, R., Kendall, A.: Fiery: future instance prediction in bird's-eye view from surround monocular cameras. In: ICCV (2021)
9. Hu, H.N., Yang, Y.H., Fischer, T., Darrell, T., Yu, F., Sun, M.: Monocular quasi-dense 3d object tracking. TPAMI (2022)
10. Hu, P., Huang, A., Dolan, J., Held, D., Ramanan, D.: Safe local motion planning with self-supervised freespace forecasting. In: CVPR (2021)
11. 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: ECCV (2022)
12. 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: CVPR (2023)
13. Huang, J., Huang, G., Zhu, Z., Du, D.: Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. arXiv preprint arXiv:2112.11790 (2021)
14. Jia, X., Wu, P., Chen, L., Liu, Y., Li, H., Yan, J.: Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding. TPAMI (2023)
15. 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: CVPR (2023)
16. 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. In: ICCV (2023)
17. Khurana, T., Hu, P., Dave, A., Ziglar, J., Held, D., Ramanan, D.: Differentiable raycasting for self-supervised occupancy forecasting. In: ECCV (2022)
18. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1097–1105 (2012)
20. Li, D., Tang, Y., Zhang, Z., Zhang, W.: Cross-stream contrastive learning for self-supervised skeleton-based action recognition. arXiv preprint arXiv:2305.02324 (2023)
21. Li, T., Chen, L., Geng, X., Wang, H., Li, Y., Liu, Z., Jiang, S., Wang, Y., Xu, H., Xu, C., et al.: Topology reasoning for driving scenes. arXiv preprint arXiv:2304.05277 (2023)

22. 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. arXiv preprint arXiv:2206.10092 (2022)
23. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Yu, Q., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. arXiv preprint arXiv:2203.17270 (2022)
24. Liang, M., Yang, B., Hu, R., Chen, Y., Liao, R., Feng, S., Urtasun, R.: Learning lane graph representations for motion forecasting. In: ECCV (2020)
25. Liang, M., Yang, B., Zeng, W., Chen, Y., Hu, R., Casas, S., Urtasun, R.: Pnpnet: End-to-end perception and prediction with tracking in the loop. In: CVPR (2020)
26. 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)
27. Liu, Y., Wang, T., Zhang, X., Sun, J.: Petr: Position embedding transformation for multi-view 3d object detection. arXiv preprint arXiv:2203.05625 (2022)
28. Mao, J., Qian, Y., Zhao, H., Wang, Y.: Gpt-driver: Learning to drive with gpt. arXiv preprint arXiv:2310.01415 (2023)
29. Mao, J., Ye, J., Qian, Y., Pavone, M., Wang, Y.: A language agent for autonomous driving. arXiv preprint arXiv:2311.10813 (2023)
30. Mo, X., Huang, Z., Xing, Y., Lv, C.: Multi-agent trajectory prediction with heterogeneous edge-enhanced graph attention network. IEEE Transactions on Intelligent Transportation Systems (2022)
31. Philion, J., Fidler, S.: Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In: ECCV (2020)
32. Prakash, A., Chitta, K., Geiger, A.: Multi-modal fusion transformer for end-to-end autonomous driving. In: CVPR (2021)
33. Salzmann, T., Ivanovic, B., Chakravarty, P., Pavone, M.: Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In: ECCV (2020)
34. Shi, W., Rajkumar, R.: Point-gnn: Graph neural network for 3d object detection in a point cloud. In: CVPR (2020)
35. Tong, W., Sima, C., Wang, T., Chen, L., Wu, S., Deng, H., Gu, Y., Lu, L., Luo, P., Lin, D., et al.: Scene as occupancy. In: ICCV (2023)
36. Velivckovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
37. Wang, Y., Solomon, J.M.: Object dgcnn: 3d object detection using dynamic graphs. In: NeurIPS (2021)
38. Weng, X., Wang, Y., Man, Y., Kitani, K.M.: Gnn3dmot: Graph neural network for 3d multi-object tracking with 2d-3d multi-feature learning. In: CVPR (2020)
39. Weng, X., Yuan, Y., Kitani, K.: Ptp: Parallelized tracking and prediction with graph neural networks and diversity sampling. IEEE Robotics and Automation Letters (2021)
40. Wu, P., Chen, L., Li, H., Jia, X., Yan, J., Qiao, Y.: Policy pre-training for end-to-end autonomous driving via self-supervised geometric modeling. arXiv preprint arXiv:2301.01006 (2023)
41. 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. In: NeurIPS (2022)
42. Xu, H., Jiang, C., Liang, X., Li, Z.: Spatial-aware graph relation network for large-scale object detection. In: CVPR (2019)

43. Yan, S., Xiong, Y., Lin, D.: Spatial temporal graph convolutional networks for skeleton-based action recognition. In: AAAI (2018)
44. Yang, J., Zheng, W.S., Yang, Q., Chen, Y.C., Tian, Q.: Spatial-temporal graph convolutional network for video-based person re-identification. In: CVPR (2020)
45. 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)
46. Zeng, W., Luo, W., Suo, S., Sadat, A., Yang, B., Casas, S., Urtasun, R.: End-to-end interpretable neural motion planner. In: CVPR (2019)
47. Zhang, T., Chen, X., Wang, Y., Wang, Y., Zhao, H.: MUTR3D: A Multi-camera Tracking Framework via 3D-to-2D Queries. In: CVPRW (2022)
48. Zhang, Z., Liniger, A., Dai, D., Yu, F., Van Gool, L.: End-to-end urban driving by imitating a reinforcement learning coach. In: ICCV (2021)