Int2Planner: An Intention-based Multi-modal Motion Planner for Integrated Prediction and Planning

Xiaolei Chen[1,2], Junchi Yan[1*], Wenlong Liao[2], Tao He[2,3*], Pai Peng[2†]

[1] School of Artificial Intelligence & Department of CSE & MoE Lab of AI, Shanghai Jiao Tong University
[2] COWAROBOT Co. Ltd.
[3] School of Electronic Engineering, University of South China
{cxlz64, yanjunchi}@sjtu.edu.cn, {volans.liao, tommie.he}@cowarobot.com, pengpai_sh@163.com

Abstract

Motion planning is a critical module in autonomous driving, with the primary challenge of uncertainty caused by interactions with other participants. As most previous methods treat prediction and planning as separate tasks, it is difficult to model these interactions. Furthermore, since the route path navigates ego vehicles to a predefined destination, it provides relatively stable intentions for ego vehicles and helps constrain uncertainty. On this basis, we construct Int2Planner, an Intention-based Integrated motion Planner achieves multi-modal planning and prediction. Instead of static intention points, Int2Planner utilizes route intention points for ego vehicles and generates corresponding planning trajectories for each intention point to facilitate multi-modal planning. The experiments on the private dataset and the public nuPlan benchmark show the effectiveness of route intention points, and Int2Planner achieves state-of-the-art performance. We also deploy it in real-world vehicles and have conducted autonomous driving for hundreds of kilometers in urban areas. It further verifies that Int2Planner can continuously interact with the traffic environment.

Code and Datasets — https://github.com/cxlz/Int2Planner

1 Introduction

In autonomous driving, motion planning is a crucial task (Hu et al. 2023a; Jia et al. 2023a) to operate ego vehicles without human intervention. Autonomous vehicles are equipped with multiple sensors that continuously observe the environment like a human driver. A key objective of this observation is to assess the movements of the surrounding agents, which is essential to create a motion plan that ensures safe, comfortable, and reliable driving. The trajectories of the surrounding agents have a significant impact on motion planning (Hagedorn et al. 2023).

However, many existing methods traditionally treat motion planning and trajectory prediction as distinct tasks. Typically, these methods first generate predictions of future trajectories for surrounding agents, which are then used as inputs for the planning task. However, they often neglect the interaction between autonomous vehicles and surrounding agents. Therefore, there is a critical need to integrate motion planning and trajectory prediction into a unified model.

In addition, the primary goal of planning is to reach the destination safely by the navigation of the route path, which determines the potential intention of autonomous vehicles. The route path can be integrated into planning models by serving as input to the model and establishing a route-oriented cost function (Sadat et al. 2020). Recent models incorporate attention mechanisms to facilitate the interaction between autonomous vehicles and route paths (Hallgarten, Stoll, and Zell 2023; Dauner et al. 2023).

Inspired by the use of target points and endpoints in prediction tasks (Zhao et al. 2021; Gilles et al. 2022; Shi et al. 2022) to reduce the uncertainty of the multi-modal future, we propose Int2Planner, a novel intention-based motion planner for integrated prediction and planning tasks. It utilizes intention points sampled from the route path to handle multi-modal planning, which is inapplicable for prediction tasks, as the route paths of surrounding vehicles are usually unknown.

Fig. 1 shows the overall framework of Int2Planner. The main contributions of this paper are listed below:

(1) We develop Int2Planner, a novel planning model, utilizing route intention points to handle the uncertainty of multi-modal planning. It combines prediction and planning in a joint model to realize the interactions between ego vehicles and surrounding agents.

(2) Instead of static intention points, we propose to sample intention points from route path to represent the potential intentions of ego vehicles. Planning trajectories are generated for each intention point to realize multi-modal planning.

(3) We are going to release a new dataset for motion planning tasks. Experiments are conducted on both the private dataset and the public nuPlan dataset. The results demonstrate that the proposed route intention points effectively improve the motion planning ability and Int2Planner achieves state-of-the-art planning performance on these datasets.

(4) We deploy Int2Planner in real-world vehicles, and the test results show that Int2Planner is capable of reacting to complex traffic scenarios and generating safe and reasonable planning trajectories.

*Corresponding authors.
†Project Lead.

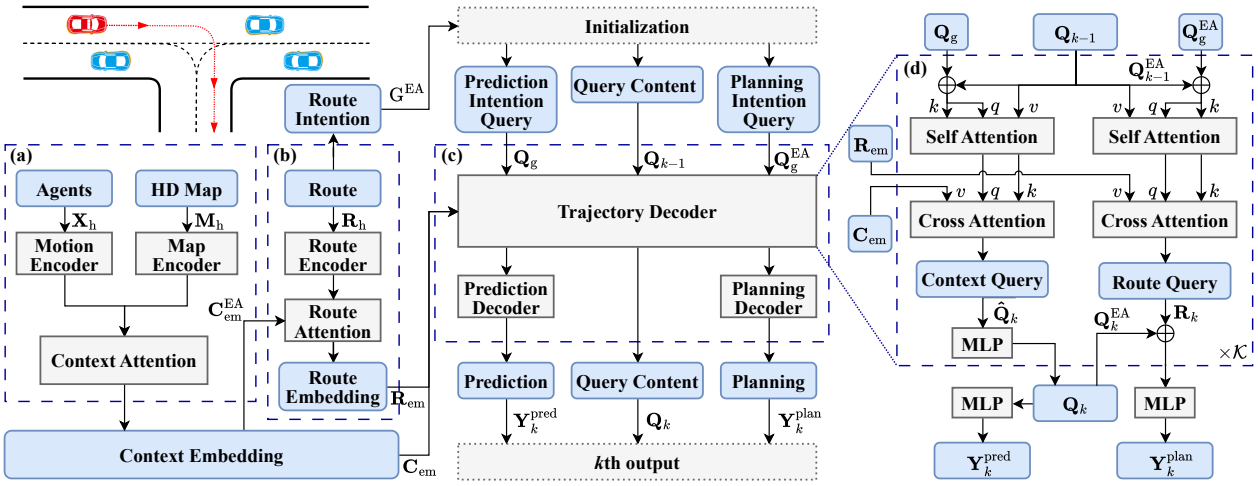


Figure 1: The overall framework of Int2Planner. (a) denotes the Context Encoder module, which encodes agent states and HD map information into context embedding, (b) denotes the Route Encoder module, which encodes route information into route embedding and (c) denotes the Trajectory Generator module, which optimizes future trajectories with K iterations, (d) denotes the detailed structure of the Trajectory Decoder layer and "$\oplus$" denotes a concatenation operation of tensors.

2 Related Work

Vehicle Trajectory Prediction Early methods (Li et al. 2017; Xie et al. 2017) focused primarily on modeling vehicle motion states, resulting in relatively accurate short-term trajectory predictions. To address this problem, probabilistic models such as Bayesian networks (Xie et al. 2018) and decision trees (Hu et al. 2017) have been used to predict the intentions of agents. Nevertheless, these models can only capture simple close-range interactions between targets. Trajectory prediction methods based on deep learning have become mainstream, including Convolutional Neural Networks (CNNs) (Ye, Cao, and Chen 2021; Gilles et al. 2022), Recurrent Neural Networks (RNNs) (Sun et al. 2022; Lin et al. 2022), Graph Neural Networks (GNNs) and Attention mechanisms (Mo et al. 2022; Varadarajan et al. 2022), enabling the consideration of more complex interactions (Li et al. 2021). However, information loss in long-time sequences is still a significant issue. Transformers can preserve long-term information from historical trajectories by incorporating position and time embedding, resulting in improvement of performance in trajectory prediction tasks (Nayakanti et al. 2023; Ngiam et al. 2022; Jia et al. 2023b; Zhou et al. 2023). However, these methods consider trajectory prediction as an independent problem without it with the motion planning task.

Integrated Motion planning Traditional motion planning depends primarily on trajectory clustering methods (Nilsson et al. 2013) and optimization-based approaches (Van Hoek, Ploeg, and Nijmeijer 2021), frequently resulting in suboptimal planning in a dynamically changing environment. Recently, motion planning methods based on machine learning have emerged to account for the evolving environment, enabling the integration of prediction and planning within a joint model. One approach of integrated planning is Robot Leader planning (Schmerling et al. 2018), where the predictions of surrounding agents are made based on the planning of the ego agent. Consequently, from the viewpoint of the ego agent, the surrounding agents conform to its planning behavior, which can lead to aggressive driving behavior. On the contrary, Human Leader planning methods (Casas, Sadat, and Urtasun 2021) aim to eliminate aggressive driving behavior, but they lack consideration for the impact of the ego agent's planning on the behavior of surrounding agents, resulting in less confident planning. To address these problems, integrated planning should consider the interaction of both ego agent and surrounding agents (Hu et al. 2023a; Ye et al. 2023; Huang, Liu, and Lv 2023). Recent work has also explored end-to-end models (Hu et al. 2023b; Jiang et al. 2023) for planning purposes to eliminate the errors introduced by the front-end process and to consider the interaction between prediction and planning.

Intention in Autonomous Driving Intention plays a pivotal role in prediction and planning tasks. Early trajectory prediction methods describe multi-modal intentions as action-based (Casas, Luo, and Urtasun 2018) or region-based (Liu et al. 2021) predictions. Recent studies (Rhinehart et al. 2019; Fang et al. 2020) introduce the concept of target points to represent vehicle intentions and generate predicted trajectories based on these target points. Some methods (Varadarajan et al. 2022; Ngiam et al. 2022) employ latent anchor features to generate target points and optimize these features through model training. Other approaches (Zhao et al. 2021) combine map information and sample target points from map lane lines. However, since the actual driving intention of a predicted vehicle cannot be known in advance, the sampled target points may not reflect the vehicle's true driving intention. DenseTNT (Gu, Sun, and Zhao 2021) addresses this issue through dense point

sampling, but the increase in target points can impact the model's inference speed. MTR (Shi et al. 2022) proposes using offline-generated clustering points to describe the formal intentions of different traffic participants. By using clustering methods, target points can be generated based on different driving behaviors, potentially improving adaptability to specific scenarios. Although target points have been extensively studied in prediction tasks, there is relatively little research in the field of motion planning. Since the intention of the ego vehicle is often determined by its route path, we suggest that resampling the route points can help eliminate irrelevant target points and thus reduce uncertainty. This approach is inapplicable for prediction tasks (Deo, Wolff, and Beijbom 2022), as the route paths of surrounding vehicles are usually unknown.

3 Methodology

As shown in Fig. 1, the overall network of Int2Planner is divided into three main components: Context Encoder, Route Encoder, and Trajectory Generator. Details of each module are introduced in this section.

Preliminaries

Commonly, a typical traffic scenario involves multiple types of vehicles, including Ego Agent (EA) and Surrounding Agents (SAs). The state histories of vehicles are adopted as the primary input features for the prediction and planning tasks. At time step t_0, the state history of a typical vehicle in the past t_h time steps is $X_i = \{x_t \mid t \in [t_0 - t_\mathrm{h}, t_0]\}$. Hence, the state histories of all the agents are denoted as:

$$\mathbf{X}_\mathrm{h} = \left\{\mathrm{X}^{\mathrm{EA}}, \mathbf{X}^{\mathrm{SA}}\right\} = \{X_i \mid i \in [0, N_a]\} \tag{1}$$

where N_a is the number of SAs, $\mathrm{X}^{\mathrm{EA}} = X_0$ denotes the state history of EA, and $\mathbf{X}^{\mathrm{SA}} = \{X_i \mid i \in [1, N_a]\}$ denotes the state histories of SAs.

In addition, map polylines from HD map are also used as input elements to provide environmental information. A typical map polyline, composed of t_m points, is $P_i = \{p_t \mid t \in [1, t_m]\}$. Hence, all the map polylines within a specific range around EA are denoted as:

$$\mathbf{M}_\mathrm{h} = \{P_i \mid i \in [1, N_m]\} \tag{2}$$

where N_m is the number of map polylines.

Furthermore, for planning tasks, route information is also a pivotal input feature for navigating EA to specific destinations. The route information used in our model is divided into two main parts. Firstly, a set of map polylines along the route to the destination is provided as the input features of the Route Encoder module. Similar to map polylines, the route polylines are denoted as:

$$\mathbf{R}_\mathrm{h} = \{P_i \mid i \in [1, N_r]\} \tag{3}$$

where N_r is the number of route polylines. Secondly, a set of intention points sampled from these map polylines along the route is adopted as the initialization for EA intention queries in the Trajectory Generator module. These intention points are sampled at an equal distance interval along the route polylines:

$$\mathrm{G}^{\mathrm{EA}} = \{g_i \mid i \in [1, N_q]\} \tag{4}$$

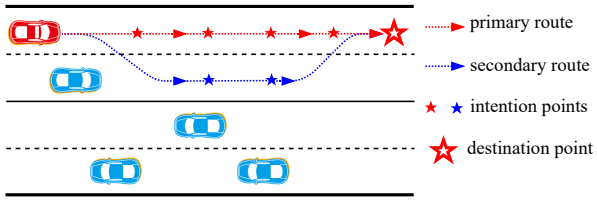


Figure 2: Explanation of route intention points sampling.

where N_q is the number of intention points g_i.

Route Intention Point. As shown in Fig. 2, EA is navigated to a unique destination, while the routes to the destination are multiple and continuously adjusted according to traffic conditions. Among these routes, one optimal route is selected as the primary route, and the others are secondary routes. Intention points of EA can be sampled from the primary route, as well as the secondary routes. These route intention points serve as short-term destinations for EA and corresponding planning trajectories are generated for each intention point. The impact of different sampling strategies is investigated in the experimental section.

For both prediction and planning tasks, the output features are represented as trajectories and each trajectory consists of t_f future points:

$$\mathbf{Y}^{\mathrm{pred}} = \left\{Y_i^{\mathrm{pred}} \mid i \in [0, N_a]\right\} \tag{5}$$

$$\mathrm{Y}^{\mathrm{plan}} = Y_0^{\mathrm{pred}} \tag{6}$$

where $Y_i^{\mathrm{pred}} = \{Y_t \mid t \in [t_0 + 1, t_0 + t_\mathrm{f}]\}$.

Context Encoder

The Context Encoder module integrates agents and HD map information into context embedding. Firstly, $\mathbf{X}_\mathrm{h}$ and $\mathbf{M}_\mathrm{h}$ are adopted as input features, and both are processed by a PointNet-like encoder, which generally consists of a Multilayer Perceptron (MLP) layer followed by a maxpooling operation, denoted as $\phi_{\mathrm{agg}}(*) = \mathbf{Maxpool}(\mathbf{MLP}(*))$. Then the aggregated agent and map embeddings are concatenated and go through a Context Attention module, which is implemented as a Self-Attention layer.

$$\mathbf{C}_{\mathrm{em}} = \phi_{\mathrm{context}}\left(\phi_{\mathrm{agg}}\left([\mathbf{X}_\mathrm{h}, \mathbf{M}_\mathrm{h}]\right)\right) \tag{7}$$

where $\phi_{\mathrm{context}}(*) = \mathbf{SelfAttn}(*)$. Through this attention module, all context elements can interact with each other.

Route Encoder

Similar to the Context Encoder module, the Route Encoder takes route polylines as input, and then aggregates route feature $\mathbf{R}_{\mathrm{agg}}$ by a PointNet-like encoder. Since only EA follows the guidance of route polylines, the context embedding of EA, denoted as $\mathrm{C}_{\mathrm{em}}^{\mathrm{EA}}$, is separated from the entire context embedding, which then goes through a Route Attention module to generate route embedding. Route embedding $\mathbf{R}_{\mathrm{em}}$ is denoted as:

$$\mathbf{R}_{\mathrm{em}} = \phi_{\mathrm{route}}\left(\left[\mathrm{C}_{\mathrm{em}}^{\mathrm{EA}}, \phi_{\mathrm{agg}}\left(\mathbf{R}_\mathrm{h}\right)\right]\right) \tag{8}$$

where $\phi_{\mathrm{route}}(*) = \mathbf{SelfAttn}(*)$ is also a Self-Attention layer. Hence, only EA interacts with route polylines.

Trajectory Generator

Trajectory Generator mainly contains a Transformer-based decoder that processes the input context embedding and route embedding. The prediction trajectory $\mathbf{Y}^{\text{pred}}$ and planning trajectory Y^{plan} are refined through an iteration process with K iterations.

To maintain the decoded features during the iteration process, the query content $\mathbf{Q}_k$ is utilized and updated every iteration. In the first iteration, the initial query content $\mathbf{Q}_1$ is initialized by a trainable embedding tensor.

Following (Shi et al. 2022), we utilize static intention points to initialize intention queries and a corresponding trajectory is generated for each intention point. Since EA is navigated with a predefined destination, instead of utilizing clustered points, the intention points of EA are sampled from route polylines $\mathbf{R}_{\text{h}}$ with an equal distance interval d_r, as illustrated in Eq. (4). The intention points of all the agents are denoted as:

$$\mathbf{G} = \left\{\mathrm{G}^{\text{EA}}, \mathbf{G}^{\text{SA}}\right\} = \{G_i \mid i \in [0, N_a]\} \tag{9}$$

where $\mathrm{G}^{\text{EA}} = G_0$ is the intention points of EA and $\mathbf{G}^{\text{SA}} = \{G_i \mid i \in [1, N_a]\}$ is the intention points of SA. The Intention Query are initialized as:

$$\mathbf{Q}_{\text{g}} = \phi_{\text{g}}\left(\mathbf{G}\right) \tag{10}$$

where ϕ_{g} is implemented as an MLP layer.

The details of the Trajectory Decoder layer is shown in the right part of Fig. 1, which basically consists of several transformer decoder layers. In the decoder part of the prediction task, query content Q_{k-1} from previous iteration firstly goes through a Self-Attention layer, with the intention query $\mathbf{Q}_{\text{g}}$ as position embedding. Secondly, the output of the Self-Attention layer is used as both the key and query of the Cross-Attention layer to extract corresponding values from context embedding $\mathbf{C}_{\text{em}}$, updating the query content to $\mathbf{Q}_k$. Finally, the updated query content is processed through an MLP layer to generate the prediction trajectory points as the outputs of kth iteration.

$$\mathbf{Q}_k = \phi_{\text{tr}}\left(\mathbf{Q}_{k-1}, \mathbf{Q}_{\text{g}}, \mathbf{C}_{\text{em}}\right) \tag{11}$$

$$\left(\mathbf{Y}_k^{\text{pred}}, \mathbf{S}_k^{\text{pred}}\right) = \phi_{\text{f}}\left(\mathbf{Q}_k\right) \tag{12}$$

where $\phi_{\text{f}}\left(*\right) = \mathbf{MLP}\left(*\right)$ is an MLP layer, $\phi_{\text{tr}}\left(*\right) = \mathbf{CrossAttn}\left(\mathbf{SelfAttn}\left(*\right)\right)$, $\mathbf{SelfAttn}$ and $\mathbf{CrossAttn}$ are Self-Attention layer and Cross-Attention layer, respectively. $\mathbf{S}_k^{\text{pred}}$ is the confidence score of multi-modal prediction trajectories.

On the other hand, the planning part utilizes a similar flow. What makes it different are three points. One is that only the query content of EA $\mathrm{Q}_{k-1}^{\text{EA}}$ and the planning intention query $\mathrm{Q}_{\text{g}}^{\text{EA}}$ are used in the Self-Attention layer. The second difference lies in the Cross-Attention layer, in which the route embedding $\mathbf{R}_{\text{em}}$ is used as the value to generate the route content $\mathbf{R}_k$. The last is that the updated query content Q_k^{EA} of EA is concatenated with route content $\mathbf{R}_k$ to generate the planning trajectory points of EA, considering both context and route features.

$$\mathbf{R}_k = \phi_{\text{tr}}\left(\mathrm{Q}_k^{\text{EA}}, \mathrm{Q}_{\text{g}}^{\text{EA}}, \mathbf{R}_{\text{em}}\right) \tag{13}$$

$$\left(\mathrm{Y}_k^{\text{plan}}, \mathrm{S}_k^{\text{plan}}\right) = \phi_{\text{f}}\left(\left[\mathrm{Q}_k^{\text{EA}}, \mathbf{R}_k\right]\right) \tag{14}$$

where $\mathrm{S}_k^{\text{plan}}$ is the confidence score of multi-modal planning trajectories.

The results of the Kth iteration are utilized as the final output of the Trajectory Generator.

Loss Function

The training process of Int2Planner is supervised by GT trajectories, with L1 loss for trajectory regression and cross-entropy loss for confidence score. Following (Shi et al. 2022), the intention point closest to the endpoint of GT trajectory is selected as the positive item, which is used to calculate the confidence score loss. The predicted trajectory corresponding to the selected intention point is used to calculate the trajectory regression loss. In addition, the final loss is the mean value of all the losses from K iterations.

4 Experiments

Experiments Setup

Datasets and Metrics We conduct experiments on a private dataset from an autonomous driving corporation. This dataset includes extensive trajectory data, localization data and route path information, making it suitable for both prediction and planning tasks. The data, collected primarily in urban environments, comes from autonomous vehicles operating under various conditions, such as daytime vs. night and sunny vs. rainy weather. These vehicles are equipped with a central roof-mounted sensor unit, which consists of one Ruby Plus 128[1] Lidar and five standard cameras for panorama vision. In addition, four fisheye cameras are installed around the vehicles for close-range vision. To generate valid ground-truth planning trajectories, all data is collected through manual driving by expert vehicle operators. Each scene contains an average of about 43 agents, detected and tracked by a state-of-the-art offline perception system.

The private dataset contains 680,964 traffic scenarios and each scenario contains 6.5 seconds of trajectory data at 10 Hz, in which 626,459 scenarios are used as train set and the rest 54,505 scenarios remain as validation set. Following the common usage, ADE and FDE are used as evaluation metrics for prediction and planning tasks. This dataset will be available at https://github.com/cxlz/Int2Planner.

In addition, experiments are also conducted on nuPlan (Caesar et al. 2021), a large-scale benchmark for planning tasks in autonomous driving. It provides a closed-loop simulator for three simulation tasks: open-loop (OL) planning, nonreactive closed-loop (NR-CL) planning and reactive closed-loop (R-CL) planning. For each task, a weighted score is calculated considering various metrics. As the online simulation engine for the nuPlan test set is closed, we use the Val14 (Dauner et al. 2023) and Test14-hard (Cheng et al. 2023) benchmarks for evaluation.

[1] https://www.robosense.ai/en/rslidar/RS-Ruby_Plus

Models	Val14				Test14-hard			
	Overall (↑)	OL (↑)	NR-CL (↑)	R-CL (↑)	Overall (↑)	OL (↑)	NR-CL (↑)	R-CL (↑)
Log-replay	*0.9133*	*1.00*	*0.94*	*0.80*	*0.8500*	*1.00*	*0.86*	*0.69*
IDM	0.6367	0.38	0.76	**0.77**	0.4600	0.20	0.56	0.62
PlanCNN	0.6967	0.64	0.73	0.72	-	-	-	-
GC-PGP	0.6433	0.82	0.57	0.54	0.5221	0.7378	0.4322	0.3963
PlanTF	**0.8360**	0.8918	**0.8483**	**0.7678**	0.7263	0.8332	**0.7286**	0.6170
Int2Planner (ours)	0.8226	**0.9097**	0.7912	0.7668	**0.7476**	**0.8673**	0.6971	**0.6784**
GameFormer Planner *	0.8216	0.8304	0.8182	0.8161	0.7035	0.7527	0.6695	0.6883
PDM-Hybrid *	**0.8967**	0.84	**0.93**	**0.92**	0.7185	0.7381	0.6595	**0.7579**
Int2Planner * (ours)	0.8385	**0.8513**	0.8372	0.8269	**0.7679**	**0.8079**	**0.7500**	0.7457

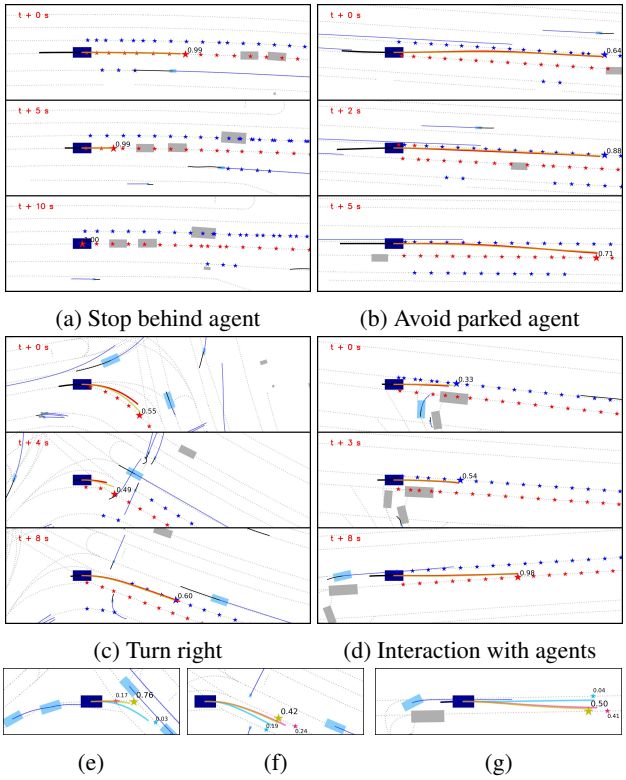
Table 1: Simulation results on nuPlan benchmark. "*" indicates the hybrid models combine learning-based and rule-based methods. "OL", "NR-CL" and "R-CL" indicate open-loop, non-reative closed-loop and reactive closed-loop simulations, respectively. "Overall" indicates the average score of three simulations.

Implementation Details We train Int2Planner on 8 NVIDIA RTX 4090 GPUs for 30 epochs with a total batch size of 96. During the training process, AdamW optimizer is used with an initial learning rate of 1×10^{-4} and a weight decay of 0.01. For the private dataset, we use $t_{\mathrm{h}} = 15$ historical points and $t_{\mathrm{f}} = 50$ future points for each agent. The feature dimension of context embedding, route embedding and the hidden dimension of attention layers are all set to 128. The distance interval d_r to sample route intention points is set to 4 meters, the number of intention points N_q is set to 64 and the number of decoder iterations K is set to 6. The range of map polylines is approximately 200 meters. For the nuPlan dataset, we adjust $t_{\mathrm{h}} = 20$ and $t_{\mathrm{f}} = 80$ to match the requirements of nuPlan benchmark, while keeping other hyper-parameters unchanged.

Main Results

Simulation Results Table 1 shows the comparison of Int2Planner with other learning-based and rule-based planners, including IDM (Treiber, Hennecke, and Helbing 2000), PlanCNN (Renz et al. 2022), GC-PGP (Hallgarten, Stoll, and Zell 2023), PlanTF (Cheng et al. 2023), PDM-Hybrid (Dauner et al. 2023) and GameFormer Planner (Huang, Liu, and Lv 2023). We train GameFormer Planner and Int2Planner on nuPlan train set and conduct simulations on Val14 and Test14-hard benchmarks. The results of other models are taken from (Dauner et al. 2023; Cheng et al. 2023).

The simulation results show that Int2Planner reaches state-of-the-art performance overall. Among purely learning-based methods, it demonstrates competitive performance with PlanTF on the Val14 benchmark. For the more complex Test14-hard benchmark, Int2Planner attains the best OL and R-CL simulation scores, as well as the highest overall score. When combined with rule-based post-processing, the performance of Int2Planner on closed-loop simulations is further enhanced. On the Test14-hard benchmark, the simulation scores of Int2Planner are either higher than or close to those of PDM-Hybrid, with an overall score of 0.7679, surpassing PDM-Hybrid by 7%.



(a) Stop behind agent (b) Avoid parked agent (c) Turn right (d) Interaction with agents (e) (f) (g)

Figure 3: Qualitative results. (a)-(d) Primary and secondary intention points are marked with red and blue "⋆". Intention points with the highest confidence are marked larger, along with their confidence scores. (e)-(g) The three most confident planning trajectories are plotted with intention points and confidence scores.

Evaluation Results Table 2 shows the evaluation results of the prediction and planning performance on the private dataset. When compared to the baseline method, Game-

Model		Plan		Prediction	
		ADE (↓)	FDE (↓)	minADE_6 (↓)	minFDE_6 (↓)
GameFormer Planner (Huang, Liu, and Lv 2023)		0.5400	1.1524	1.3957	2.0861
Int2Planner (ours, w/o Pred.)		0.4028	1.1088	-	-
Int2Planner (ours)	Cluster Intention	0.4788	1.3436	**0.5654**	**1.1592**
	Route Intention (Primary)	0.4141	1.1505	0.6067	1.3049
	Route Intention (All)	**0.3946**	**1.0877**	0.6010	1.2794

Table 2: Evaluations on the private dataset. "Cluster Intention" and "Route Intention" indicate that intention points are generated by K-means clustering method and sampled from route polylines, respectively. "Primary" indicates that only the primary route polylines are sampled and "All" indicates that the secondary route polylines are also sampled.

Former Planner, Int2Planner performs better on motion planning metrics, and also shows significant improvement in trajectory prediction performance. Both joint prediction and route intention improve the metrics, while the contributions are different. Although, the model without prediction task does not produce explicit prediction, it still takes SA features as inputs, allowing it to extract SA embeddings for EA planning.

Qualitative Results Fig. 3a-3d shows several common traffic scenarios sampled from the private validation dataset. Bounding boxes of EA, moving and static SAs are colored in deep blue, light blue and grey. Ground-truth, planning, and prediction trajectories are shown in red, yellow and blue. The planning trajectory with the highest confidence is highlighted in each figure. These scenarios show that Int2Planner interacts with SAs based on the perception and prediction results, making reasonable planning actions.

Intention Visualization As shown in Fig. 3a-3d, the specific intention point corresponding to the planning trajectory with the highest confidence is marked with a larger size and a confidence score. The selected point is almost the closest intention point to the end point of the plotted planning trajectory. This demonstrates that the proposed route intention points offer sufficient potential options and Int2Planner can select appropriate intention points for EA.

Fig. 3e-3g shows the multi-modal intentions in more detail. The end points of three most confident planning trajectories are distributed relatively sparsely and close to the intention points. It indicates that Int2Planner treats these intention points as short-term destinations, and generates corresponding planning trajectories, which are refined based on the selected intention points.

Confidence Scores Distribution In addition, the confidence score distribution for the planning task for the top 6 intention points are shown in Fig. 4 and the confidence scores tend to concentrate on the top few points.

Ablation Study

We conduct ablation studies on the designs of Int2Planner and all the simulation experiments are conducted on Test14-hard benchmark without combining rule-based post-processing.

RE	CI	RI	OL (↑)	NR-CL (↑)	R-CL (↑)
×	✓	×	0.8377	0.6775	0.6559
×	×	✓	0.8606	0.6587	0.6760
✓	✓	×	0.8453	0.6256	0.6523
✓	×	✓	**0.8673**	**0.6971**	**0.6784**

Table 3: Effects of route information on Test14-hard benchmark. "RE" indicates route embedding, "CI" indicates Cluster Intention, and "RI" indicates Route Intention.

Integrated Prediction	OL (↑)	NR-CL (↑)	R-CL (↑)
×	0.8649	0.6784	0.6736
✓	**0.8673**	**0.6971**	**0.6784**

Table 4: Effects of integrated prediction in Int2Planner.

Number of Iterations	OL (↑)	NR-CL (↑)	R-CL (↑)
K=1	0.8413	0.6154	0.6062
K=2	0.8602	0.6503	0.6416
K=3	0.8622	0.6778	0.6756
K=6	0.8673	**0.6971**	**0.6784**
K=9	**0.8736**	0.6871	0.6764

Table 5: Effects of number of decoder iterations.

Outputof kth Layer	OL (↑)	NR-CL (↑)	R-CL (↑)
k=1	0.8521	0.5651	0.6112
k=2	0.8588	0.6350	0.6504
k=3	0.8606	0.6767	0.6638
k=6	**0.8673**	**0.6971**	**0.6784**

Table 6: Effects of iteration strategy. The output of kth decoder layer is used as final output.

Effects of Route Information Table 3 shows the effects of route embedding and Route Intention Points. Route embedding indicates $\mathbf{R}_{\text{em}}$ expressed in Eq. 8. The strategies of

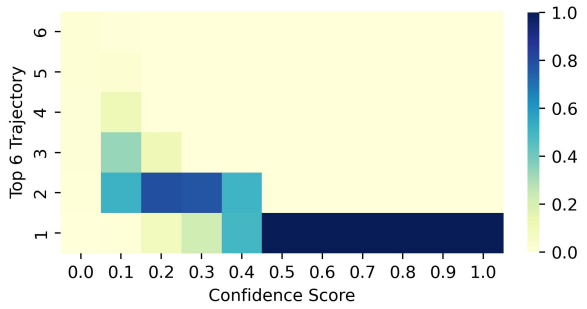


Figure 4: The distribution of planning confidence score.

generating intention points for EA are compared. Cluster intention points are generated by applying the K-means Clustering algorithm on all the endpoints of GT trajectories (Shi et al. 2022) and route intention points are sampled from the route path as shown in Fig 2. Table 3 shows that it achieves the best simulation scores, combined with route embedding and Route Intention Points.

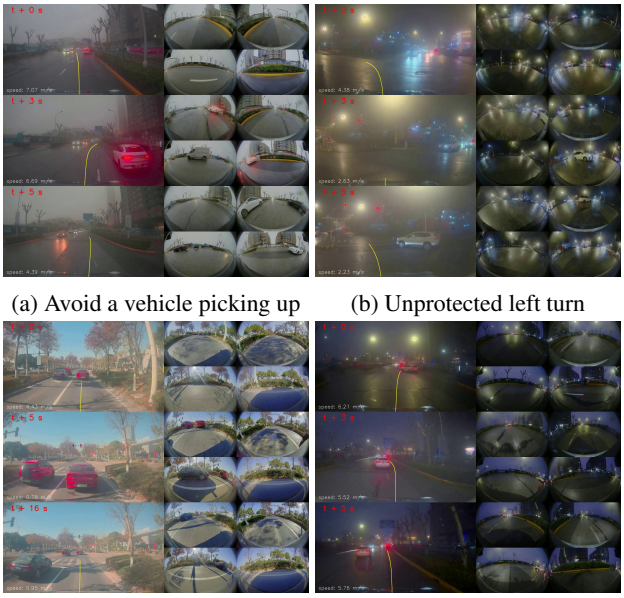
The experiment results on the private dataset shown in Table 2 further verify that route intention points significantly outperform cluster intention points. In addition, sampling points from both the primary and secondary route polylines further reduces the planning distance error. In detail, Fig. 3b and 3d show that route intention points sampled from secondary route polylines provide EA with additional routing options, especially when the primary route is blocked.

Effects of Integrated Prediction As shown in Table 4, experiments are conducted on Int2Planner by removing prediction task, to show the effects of integrated prediction. The results show that the planning performance deteriorates without the prediction task, which indicates that combining prediction and planning is necessary.

Effects of Decoder Iteration The value of K is the total number of decoder iterations. Table 5 shows the performance improvement with the increased number of iterations from 1 to 6. When further increasing the number of iterations to 9, the simulation results does not indicate obvious improvement. Therefore, we finally choose $K = 6$ for Int2Planner in the remaining experiments. Table 6 demonstrates the performance of the planning trajectories from the kth iteration (the K iteration is used by default). The results further verify the effectiveness of the iteration strategy.

Real-world Vehicle Test

We deploy Int2Planner in real-world autonomous driving vehicles. The test vehicles are the same type as those used to collect the private dataset. The perception and track results based on onboard sensors are utilized as input features of Int2Planner to extract historical states of SA. In addition, the route path is generated by a rule-based routing planner. The planning trajectory with the highest confidence of the multi-modal outputs of Int2Planner is passed to the control system to operate the autonomous driving vehicle.



(a) Avoid a vehicle picking up (b) Unprotected left turn

(c) Reactions to traffic lights (d) Avoid a moving vehicle

Figure 5: Real-world tests in urban areas. The front view image are combined with four surrounding view images and the output planning trajectories are projected as yellow lines.

We have conducted autonomous driving for hundreds of kilometers in urban areas in various scenes. During test experiments, each vehicle is equipped with onboard vehicle operators and a remote monitor system to prevent dangerous driving behaviors that may be caused by model failure. Fig. 5 shows several real-world test scenarios. The front view image and four surrounding view images are combined to show the entire environment, and for clarity, the planning trajectory of EA is projected onto the front view image. Each scenario is shown with three sub-images, with relative times noted in the top-left corner.

5 Conclusion

In this paper, we propose Int2Planner, an intention-based motion planner for integrated prediction and planning. We constrain the uncertainty of EA by route intention points, and multi-modal planning trajectories are generated and optimized based on route intention points. The experimental results show that our model achieves the state-of-the-art performance. Route intention points effectively improve planning performance and provide reasonable intentions for EA. Furthermore, we deploy Int2Planner in real-world vehicles and the tests show that Int2Planner can continuously interact with the environment and output reasonable and safe planning trajectories. **Limitation and Future Work.** In closed-loop simulations and real-world vehicle tests, the output planning trajectory with the highest confidence is currently utilized, but this trajectory is not necessarily the optimal one, especially in complex traffic scenarios. In future work, we will focus on handling multi-modal planning to further improve performance.

Acknowledgements

The work was in part supported by Shanghai Municipal Science and Technology Project under Grant 22Z510901584, 2021SHZDZX0102.

References

Caesar, H.; Kabzan, J.; Tan, K. S.; Fong, W. K.; Wolff, E.; Lang, A.; Fletcher, L.; Beijbom, O.; and Omari, S. 2021. Nuplan: A closed-loop ML-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv:2106.11810*.

Casas, S.; Luo, W.; and Urtasun, R. 2018. Intentnet: Learning to predict intention from raw sensor data. In *Conference on Robot Learning*, 947–956. PMLR.

Casas, S.; Sadat, A.; and Urtasun, R. 2021. MP3: A Unified Model To Map, Perceive, Predict and Plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 14403–14412.

Cheng, J.; Chen, Y.; Mei, X.; Yang, B.; Li, B.; and Liu, M. 2023. Rethinking Imitation-based Planner for Autonomous Driving. *arXiv preprint arXiv:2309.10443*.

Dauner, D.; Hallgarten, M.; Geiger, A.; and Chitta, K. 2023. Parting with Misconceptions about Learning-based Vehicle Motion Planning. *arXiv preprint arXiv:2306.07962*.

Deo, N.; Wolff, E.; and Beijbom, O. 2022. Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning*, 203–212. PMLR.

Fang, L.; Jiang, Q.; Shi, J.; and Zhou, B. 2020. Tpnet: Trajectory proposal network for motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6797–6806.

Gilles, T.; Sabatini, S.; Tsishkou, D.; Stanciulescu, B.; and Moutarde, F. 2022. GOHOME: Graph-oriented heatmap output for future motion estimation. In *2022 International Conference on Robotics and Automation (ICRA)*, 9107–9114.

Gu, J.; Sun, C.; and Zhao, H. 2021. DenseTNT: End-to-end Trajectory Prediction from Dense Goal Sets. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 15283–15292.

Hagedorn, S.; Hallgarten, M.; Stoll, M.; and Condurache, A. 2023. Rethinking Integration of Prediction and Planning in Deep Learning-Based Automated Driving Systems: A Review. *arXiv preprint arXiv:2308.05731*.

Hallgarten, M.; Stoll, M.; and Zell, A. 2023. From Prediction to Planning With Goal Conditioned Lane Graph Traversals. *arXiv preprint arXiv:2302.07753*.

Hu, M.; Liao, Y.; Wang, W.; Li, G.; Cheng, B.; and Chen, F. 2017. Decision tree-based maneuver prediction for driver rear-end risk-avoidance behaviors in cut-in scenarios. *Journal of Advanced Transportation*, 2017.

Hu, Y.; Yang, J.; Chen, L.; Li, K.; Sima, C.; Zhu, X.; Chai, S.; Du, S.; Lin, T.; Wang, W.; Lu, L.; Jia, X.; Liu, Q.; Dai, J.; Qiao, Y.; and Li, H. 2023a. Planning-Oriented Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 17853–17862.

Hu, Y.; Yang, J.; Chen, L.; Li, K.; Sima, C.; Zhu, X.; Chai, S.; Du, S.; Lin, T.; Wang, W.; Lu, L.; Jia, X.; Liu, Q.; Dai, J.; Qiao, Y.; and Li, H. 2023b. Planning-Oriented Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 17853–17862.

Huang, Z.; Liu, H.; and Lv, C. 2023. GameFormer: Game-theoretic Modeling and Learning of Transformer-based Interactive Prediction and Planning for Autonomous Driving. *arXiv preprint arXiv:2303.05760*.

Jia, X.; Gao, Y.; Chen, L.; Yan, J.; Liu, P. L.; and Li, H. 2023a. DriveAdapter: Breaking the Coupling Barrier of Perception and Planning in End-to-End Autonomous Driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 7953–7963.

Jia, X.; Wu, P.; Chen, L.; Liu, Y.; Li, H.; and Yan, J. 2023b. Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Jiang, B.; Chen, S.; Xu, Q.; Liao, B.; Chen, J.; Zhou, H.; Zhang, Q.; Liu, W.; Huang, C.; and Wang, X. 2023. VAD: Vectorized Scene Representation for Efficient Autonomous Driving. *ICCV*.

Li, J.; Dai, B.; Li, X.; Li, C.; and Di, Y. 2017. A real-time and predictive trajectory-generation motion planner for autonomous ground vehicles. In *2017 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, volume 2, 108–113.

Li, K.; Eiffert, S.; Shan, M.; Gomez-Donoso, F.; Worrall, S.; and Nebot, E. 2021. Attentional-GCNN: Adaptive pedestrian trajectory prediction towards generic autonomous vehicle use cases. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 14241–14247.

Lin, L.; Li, W.; Bi, H.; and Qin, L. 2022. Vehicle Trajectory Prediction Using LSTMs With Spatial–Temporal Attention Mechanisms. *IEEE Intelligent Transportation Systems Magazine*, 14(2): 197–208.

Liu, Y.; Zhang, J.; Fang, L.; Jiang, Q.; and Zhou, B. 2021. Multimodal motion prediction with stacked transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 7577–7586.

Mo, X.; Huang, Z.; Xing, Y.; and Lv, C. 2022. Multi-Agent Trajectory Prediction With Heterogeneous Edge-Enhanced Graph Attention Network. *IEEE Transactions on Intelligent Transportation Systems*, 23(7): 9554–9567.

Nayakanti, N.; Al-Rfou, R.; Zhou, A.; Goel, K.; Refaat, K. S.; and Sapp, B. 2023. Wayformer: Motion forecasting via simple & efficient attention networks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2980–2987.

Ngiam, J.; Vasudevan, V.; Caine, B.; Zhang, Z.; Chiang, H.-T. L.; Ling, J.; Roelofs, R.; Bewley, A.; Liu, C.; Venugopal, A.; Weiss, D. J.; Sapp, B.; Chen, Z.; and Shlens, J. 2022. Scene Transformer: A unified architecture for predicting future trajectories of multiple agents. In *International Conference on Learning Representations (ICLR)*.

Nilsson, J.; Ali, M.; Falcone, P.; and Sjöberg, J. 2013. Predictive manoeuvre generation for automated driving. In *16th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 418–423.

Renz, K.; Chitta, K.; Mercea, O.-B.; Koepke, A.; Akata, Z.; and Geiger, A. 2022. Plant: Explainable planning transformers via object-level representations. *arXiv preprint arXiv:2210.14222*.

Rhinehart, N.; McAllister, R.; Kitani, K.; and Levine, S. 2019. Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2821–2830.

Sadat, A.; Casas, S.; Ren, M.; Wu, X.; Dhawan, P.; and Urtasun, R. 2020. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 414–430. Springer.

Schmerling, E.; Leung, K.; Vollprecht, W.; and Pavone, M. 2018. Multimodal probabilistic model-based planning for human-robot interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 3399–3406.

Shi, S.; Jiang, L.; Dai, D.; and Schiele, B. 2022. Motion transformer with global intention localization and local movement refinement. *Advances in Neural Information Processing Systems*, 35: 6531–6543.

Sun, Q.; Huang, X.; Gu, J.; Williams, B.; and Zhao, H. 2022. M2i: From factored marginal trajectory prediction to interactive prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 6543–6552.

Treiber, M.; Hennecke, A.; and Helbing, D. 2000. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2): 1805.

Van Hoek, R.; Ploeg, J.; and Nijmeijer, H. 2021. Cooperative Driving of Automated Vehicles Using B-Splines for Trajectory Planning. *IEEE Transactions on Intelligent Vehicles*, 6(3): 594–604.

Varadarajan, B.; Hefny, A.; Srivastava, A.; Refaat, K. S.; Nayakanti, N.; Cornman, A.; Chen, K.; Douillard, B.; Lam, C. P.; Anguelov, D.; and Sapp, B. 2022. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, 7814–7821.

Xie, G.; Gao, H.; Huang, B.; Qian, L.; and Wang, J. 2018. A driving behavior awareness model based on a dynamic Bayesian network and distributed genetic algorithm. *International Journal of Computational Intelligence Systems*, 11(1): 469–482.

Xie, G.; Gao, H.; Qian, L.; Huang, B.; Li, K.; and Wang, J. 2017. Vehicle trajectory prediction by integrating physics- and maneuver-based approaches using interactive multiple models. *IEEE Transactions on Industrial Electronics*, 65(7): 5999–6008.

Ye, M.; Cao, T.; and Chen, Q. 2021. TPCN: Temporal point cloud networks for motion forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 11318–11327.

Ye, T.; Jing, W.; Hu, C.; Huang, S.; Gao, L.; Li, F.; Wang, J.; Guo, K.; Xiao, W.; Mao, W.; Zheng, H.; Li, K.; Chen, J.; and Yu, K. 2023. Fusionad: Multi-modality fusion for prediction and planning tasks of autonomous driving. *arXiv preprint arXiv:2308.01006*.

Zhao, H.; Gao, J.; Lan, T.; Sun, C.; Sapp, B.; Varadarajan, B.; Shen, Y.; Shen, Y.; Chai, Y.; Schmid, C.; Li, C.; and Anguelov, D. 2021. TNT: Target-driven Trajectory Prediction. In *Proceedings of the 2020 Conference on Robot Learning*, volume 155, 895–904.

Zhou, Z.; Wang, J.; Li, Y.-H.; and Huang, Y.-K. 2023. Query-centric trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17863–17873.