Planning-oriented Autonomous Driving
Supplementary Material

Yihan Hu[1,2*], Jiazhi Yang[1*], Li Chen[1*†], Keyu Li[1*], Chonghao Sima[1], Xizhou Zhu[3,1]
Siqi Chai[2], Senyao Du[2], Tianwei Lin[2], Wenhai Wang[1], Lewei Lu[3], Xiaosong Jia[1]
Qiang Liu[2], Jifeng Dai[1], Yu Qiao[1], Hongyang Li[1†]

[1] OpenDriveLab and OpenGVLab, Shanghai AI Laboratory
[2] Wuhan University [3] SenseTime Research

[*]Equal contribution [†]Project lead

https://github.com/OpenDriveLab/UniAD

Contents

A. Task Definition

Detection and tracking. Detection and tracking are two crucial perception tasks for autonomous driving, and we focus on representing them in the 3D space to facilitate downstream usage. 3D Detection is responsible for locating surrounding objects (coordinates, length, width, height, *etc.*) at each time stamp; tracking aims at finding the correspondences between different objects across time stamps and associating them temporally (*i.e.*, assigning a consistent track ID for each agent). In the paper, we use multi-object tracking in some cases to denote the detection and tracking process. The final output is a series of associated 3D boxes in each frame, and their corresponding features Q_A are forwarded to the motion module. Additionally, note that we have one special query named *ego-vehicle query* for downstream tasks, which would not be included in the prediction-ground truth matching process and it regresses the location of ego-vehicle accordingly.

Online mapping. Map intuitively embodies the geometric and semantic information of the environment, and online mapping is to segment meaningful road elements with onboard sensor data (multi-view images in our case) as a substitute for offline annotated high-definition (HD) maps. In UniAD, we model the online map into four categories: lanes, drivable area, dividers and pedestrian crossings, and we segment them in bird's-eye-view (BEV). Similar to Q_A, the map queries Q_M would be further utilized in the motion forecasting module to model the agent-map interaction.

Motion forecasting. Bridging perception and planning, prediction plays an important role in the whole autonomous driving system to ensure final safety. Typically, motion forecasting is an independently developed module that predicts agents' future trajectories with detected bounding boxes and HD maps. And the bounding boxes are ground truth annotations in most current motion datasets [20], which is not realistic in onboard scenarios. While in this paper, the motion forecasting module takes previously encoded sparse queries (*i.e.*, Q_A and Q_M) and dense BEV features B as inputs, and forecasts $\mathcal{K}$ plausible trajectories in future T timesteps for each agent. Besides, to be compatible with our end-to-end and scene-centric scenarios, we predict trajectories as offset according to each agent's current position. The agent features before the last decoding MLPs, which have encoded both the historical and future information will be sent to the occupancy module for scene-level future understanding. For the *ego-vehicle query*, it predicts future ego-motion as well (actually providing a coarse planning estimation), and the feature is employed by the planner to generate the ultimate goal.

Occupancy prediction. Occupancy grid map is a discretized BEV representation where each cell holds a belief indicating whether it is occupied, and the occupancy prediction task is designed to discover how the grid map changes in the future for T_o timesteps with multiple agent dynamics. Complementary to motion forecasting which is conditioned on sparse agents, occupancy prediction is densely represented in the whole-scene level. To investigate how the scene evolves with sparse agent knowledge, our proposed occupancy module takes as inputs both the observed BEV feature B and agent features G^t. After the multi-step agent-scene interaction (detailedly described in Appendix E), the instance-level probability map $\hat{O}_A^t \in \mathbb{R}^{N_a \times H \times W}$ is generated via matrix multiplication between occupancy feature and dense scene feature. To form whole-scene occupancy with agent identity preserved $\hat{O}^t \in \mathbb{R}^{H \times W}$ which is used for occupancy evaluation and downstream planning, we simply merge the instance-level probability at each timestep using pixel-wise argmax as in [5].

Planning. As an ultimate goal, the planning module takes all upstream results into consideration. Traditional planning methods in the industry often are rule-based, formulated by "if-else" state machines conditioned on various scenarios which are described with prior detection and prediction results. In our learning-based model, we take the upstream *ego-vehicle query*, and the dense BEV feature B as input, and predict one trajectory $\hat{\tau}$ for total T_p timesteps. Then, the trajectory $\hat{\tau}$ is optimized with the upstream predicted future occupancy $\hat{O}$ to avoid collision and ensure final safety.

B. The Necessity of Each Task

In terms of perception, *tracking* in the loop as does in PnPNet [43] and ViP3D [23] is proven to complement spatial-temporal features and provide history tracks for occluded agents, refraining from catastrophic decisions for downstream planning. With the aid of HD maps [23, 43, 58, 70] and motion forecasting, planning becomes more accurate toward higher-level intelligence. However, such information is expensive to construct and prone to be outdated, raising the demand for online *mapping* without HD maps. As for prediction, *motion* forecasting [7, 22, 32, 33, 76] generates long-term future behaviors and preserves agent identity in form of sparse waypoint outputs. Nonetheless, there exists the challenge to integrate non-differentiable box representation into subsequent planning module [23, 43]. Some recent literature investigates another type of prediction task named *occupancy* [62] prediction to assist end-to-end planning, in form of cost maps. However, the lack of agent identity and dynamics in occupancy makes it impractical to model social interactions for safe planning. The large computational consumption of modeling multi-step dense features also leads to a much shorter temporal horizon compared to *motion* forecasting. Therefore, to benefit from the two complementary types of prediction tasks for safe *planning*, we incorporate both agent-centric motion and whole-scene occupancy in UniAD.

C. Related Work

C.1. Joint perception and prediction

Joint learning of perception and prediction is proposed to avoid the cascading error in traditional modular-independence pipelines. Similar to the motion forecasting task alone, it usually has two types of output representations: agent-level bounding boxes and scene-level occupancy grid maps. Pioneering work FaF [49] predicts boxes in the future and aggregates past information to produce tracklets. IntentNet [7] extends it to reason about intentions and [19, 21] further predict future states in a refinement fashion. Some exploit detection first and utilize agent features in the second prediction stage [6, 39, 53]. Noticing that history information is ignored, PnPNet [43] enriches it by estimating tracking association scores to avert the non-differentiable optimization process as adopted by the tracking-by-detection paradigm [40, 47, 60, 68]. Yet, all these methods rely on non-maximum suppression (NMS) in detection which still leads to information loss. ViP3D [23] which is closely related to our work, employs agent queries in [73] to forecast, taking HD map as another input. We follow the philosophy of [23, 73] in agent track queries, but also develop non-linear optimization on target trajectories to alleviate the potential inaccurate perception problem. Moreover, we introduce an ego-vehicle query for better capturing the ego behaviors in the dynamic environment, and incorporate online mapping to prevent the localization risk or high construction cost with HD map.

The alternative representation, namely the occupancy grid map, discretizes the BEV map into grid cells which holds a belief indicating if it is occupied. Wu *et al*. [66] estimate a dense motion field, while it could not capture multimodal behaviors. Fishing Net [25] also predicts deterministic future BEV semantic segmentation with multiple sensors. To address this, P3 [58] proposes non-parametric distribution of future semantic occupancy and FIERY [27] devises the first paradigm for multi-view cameras. A few methods improve the performance of FIERY with more sophisticated uncertainty modeling [1, 29, 74]. Notably, this representation could easily extend to motion planning for collision avoidance [8,29,58], while it loses the agent identity characteristic and takes a heavy burden to computation which may constrain the prediction horizon. In contrast, we leverage agent-level information for occupancy prediction and ensure accurate and safe planning by unifying these two modes.

C.2. Joint prediction and planning

PRECOG [57] proposes a recurrent model that conditions forecasting on the goal position of the ego vehicle, while PiP [61] generates agents' motion considering complete presumed planning trajectories. However, producing a rough future trajectory is still challenging in the real world, toward which [46] presents a deep structured model to derive both prediction and planning from the same set of learnable costs. [30, 31] couple the prediction model with classic optimization methods. Meanwhile, some motion forecasting methods implicitly include the planning task by producing their future trajectories simultaneously [9, 34, 51]. Similarly, we encode possible behaviors of the ego vehicle in the scene-centric motion forecasting module, but the interpretable occupancy map is utilized to further optimize the plan to stay safe.

C.3. End-to-end motion planning

End-to-end motion planning has been an active research domain since Pomerleau [54] uses a single neural network that directly predicts control signals. Subsequent studies make great advances especially in closed-loop simulation with deeper networks [3], multi-modal inputs [2, 16, 55], multi-task learning [15, 67], reinforcement learning [10, 11, 35, 44, 63] and distillation from certain privilege knowledge [13, 72, 75]. However, for such methods of directly generating control outputs from sensor data, the transfer from the synthetic environment to realistic application remains a problem considering their robustness and safety assurance [17, 29]. Thus researchers aim at explicitly designing the intermediate representations of the network to prompt safety, where predicting how the scene evolves attracts broad interest. Some works [14,26,59] jointly decode planning and BEV semantic predictions to enhance interpretability, while PLOP [4] adopts a polynomial formulation to provide smooth planning results for both ego vehicle and neighbors. Cui *et al*. [18] introduce a contingency planner with diverse sets of future predictions and LAV [12] trains the planner with all vehicles' trajectories to provide richer training data. NMP [70] and its variant [65] estimate a cost volume to select the plan with minimal cost besides deterministic future perception. Though they risk producing inconsistent results between two modules, the cost map design is intuitive to recover the final plan in complex scenarios. Inspired by [70], most recent works [8,28,29,58,71] propose models that construct costs with both learned occupancy prediction and hand-crafted penalties. However, their performances heavily rely on the tailored cost based on human experience and the distribution from where trajectories are sampled [36]. Contrary to these approaches, we leverage the ego-motion information without sophisticated cost design and present the first attempt that incorporates the tracking module along with two genres of prediction representations simultaneously in an end-to-end model.

D. Notations

We provide a lookup table of notations and their shapes mentioned in this paper in Table 1 for reference.

E. Implementation Details

E.1. Detection and Tracking

We inherit most of the detection designs from BEVFormer [41] which takes a BEV encoder to transform image features into BEV feature B and adopts a Deformable DETR head [77] to perform detection on B. To further conduct end-to-end tracking without heavy post association, we introduce another group of queries named track queries as in MOTR [69] which continuously tracks previously observed instances according to its assigned track ID. We introduce the tracking process in detail below.

Training stage: At the beginning (*i.e*., first frame) of each training sequence, all queries are considered detection queries and predict all newborn objects, which is actually the same as BEVFormer. Detection queries are matched to the ground truth by the Hungarian algorithm [5]. They will be stored and updated via the query interaction module (QIM) for the next timestamp serving as track queries following MOTR [69]. In the next timestamp, track queries will be directly matched with a part of ground-truth objects according to the corresponding track ID, and detection queries will be matched with the remaining ground-truth objects (newborn objects). To stabilize training, we adopt the 3D IoU metric to filter the matched queries. Only those predictions having the 3D IoU with ground-truth boxes larger than a certain threshold (0.5 in practice) will be stored and updated.

Notation	Shape & Params.	Description
Q_o	900	number of initial object queries
D	256	embed dimensions
B	$200 \times 200 \times 256$	BEV feature encoded by a multi-view framework
N	6	number of transformer decoder layers for TrackFormer
N	6	number of transformer decoder layers for MapFormer
N	4	number of mask decoder layers for MapFormer
N	3	number of transformer decoder layers for MotionFormer
N	5	number of transformer decoder layers for OccFormer
N	3	number of transformer decoder layers for Planner
N_a	*dynamic*	number of agents from TrackFormer
N_m	300	number of map queries from MapFormer
Q_A	$N_a \times 256$	agent features from TrackFormer
P_A	$N_a \times 256$	agent positions from TrackFormer
Q_M	$N_m \times 256$	map features from MapFormer
$\mathcal{K}$	6	number of forecasting modality in MotionFormer
$\tilde{\mathbf{x}}$	$T \times 2$	ground truth for one agent's motion forecasting
$\hat{\mathbf{x}}$	$N_a \times T \times 2$	prediction of motion forecasting
T	12	length of prediction timestamps in MotionFormer
Q_{pos}	$N_a \times \mathcal{K} \times 256$	query position in MotionFormer
Q_{ctx}	$N_a \times \mathcal{K} \times 256$	query context in MotionFormer
Q_a	$N_a \times \mathcal{K} \times 256$	motion query after agent-agent interaction in MotionFormer
Q_m	$N_a \times \mathcal{K} \times 256$	motion query after agent-map interaction in MotionFormer
Q_g	$N_a \times \mathcal{K} \times 256$	motion query after agent-goal point interaction in MotionFormer
l	-	index of decoder layer
PE	-	sinusoidal position encoding function
I^s	$\mathcal{K} \times T \times 2$	scene-level anchor position in MotionFormer
I^a	$\mathcal{K} \times T \times 2$	agent-level anchor position in MotionFormer
Φ	-	kinematic cost function set
T_o	5	length of prediction timestamps in OccFormer
G^t	$N_a \times 256$	agent feature input
F^t	$200 \times 200 \times 256$	future state output
Q_X	$N_a \times 256$	motion query (max-pooled on modality level) from the last layer of MotionFormer
F^t_{ds}	$25 \times 25 \times 256$	downscaled dense feature
F^t_{dec}	$200 \times 200 \times 256$	decoded dense feature after convolutional decoder
D^t_{ds}	$25 \times 25 \times 256$	agent-aware dense feature after pixel-agent interaction
$\hat{O}^t_A$	$N_a \times 200 \times 200$	instance-level probability map
$\hat{O}^t$	200×200	classical instance-agnostic occupancy map merged from $\hat{O}^t_A$ for planning
O^t_m	200×200	attention mask for pixel-agent interaction
M^t	$N_a \times 256$	mask feature
U^t	$N_a \times 256$	occupancy feature
T_p	6	length of planning timestamps in Planner
$\hat{\tau}$	$T_p \times 2$	planned trajectory before the optimization with occupancy prediction
τ^*	$T_p \times 2$	ultimate plan output
λ	-	hyperparameters in cost functions, target functions, *etc*.

Table 1. **Lookup table of notations and hyperparameters** in the paper. The superscript t in certain notations denotes the t^{th} block of OccFormer, and is omitted in descriptions for simplicity.

Inference stage: Different from the training stage, each frame of a sequence is sent to the network sequentially, meaning that track queries could exist for a longer horizon than the training time. Another difference emerging in the inference stage is about query updating, that we use classification scores to filter the queries (0.4 for detection queries and 0.35 for track queries in practice) instead of the 3D IoU metric since the ground truth is not available. Besides, to avoid the interruption of tracklets caused by short-time occlusion, we use a lifecycle mechanism for the tracklets in the inference stage. Specifically, for each track query, it will be considered to disappear completely and be removed only when its corresponding classification score is smaller than 0.35 for a continuous period ($2s$ in practice).

E.2. Online Mapping

Following [42], we decompose the map query set into thing queries and stuff queries. The thing queries model instance-wise map elements (*i.e.*, lanes, boundaries, and pedestrian crossings) and are matched with ground truth via bipartite matching, while the stuff query is only in charge of semantic elements (*i.e.*, drivable area) and is processed with a class-fixed assignment. We set the total number of thing queries to 300 and only 1 stuff query for the drivable area. Also, we stack 6 location decoder layers and 4 mask decoder layers (we follow the structure of those layers as in [42]). We empirically choose thing queries after the location decoder as our map queries Q_M for downstream tasks.

E.3. Motion Forecasting

To better illustrate the details, we provide a diagram as shown in Fig. 1. Our MotionFormer takes I_T^a, I_T^s, $\hat{x}_0$, $\hat{x}_T^{l-1} \in \mathbb{R}^{\mathcal{K}\times 2}$ to embed query position, and takes Q_{ctx}^{l-1} as query context. Specifically, the anchors are clustered among training data of all agents by the k-means algorithm, and we set $\mathcal{K}=6$ which is compatible with our output modalities. To embed the scene-level prior, the anchor I_T^a is rotated and translated into the global coordinate frame according to each agent's current location and heading angle, which is denoted as I_T^s, as shown in Eq. (1),

$$I_{i,T}^s = R_i I_T^a + T_i, \tag{1}$$

where i is the index of the agent, and it is omitted later for brevity. To facilitate the coarse-to-fine paradigm, we also adopt the goal point predicted from the previous layer $\hat{x}_T^{l-1}$. In the meantime, the agent's current position is broadcast across the modality, denoted as $\hat{x}_0$. Then, MLPs and sinusoidal positional embeddings are applied for each of the prior positional knowledge and we summarize them as the query position $Q_{\text{pos}} \in \mathbb{R}^{\mathcal{K}\times\mathcal{D}}$, which is of the same shape as the query context Q_{ctx}. Q_{pos} and Q_{ctx} together build up our motion query. We set $\mathcal{D}$ to 256 throughout MotionFormer.

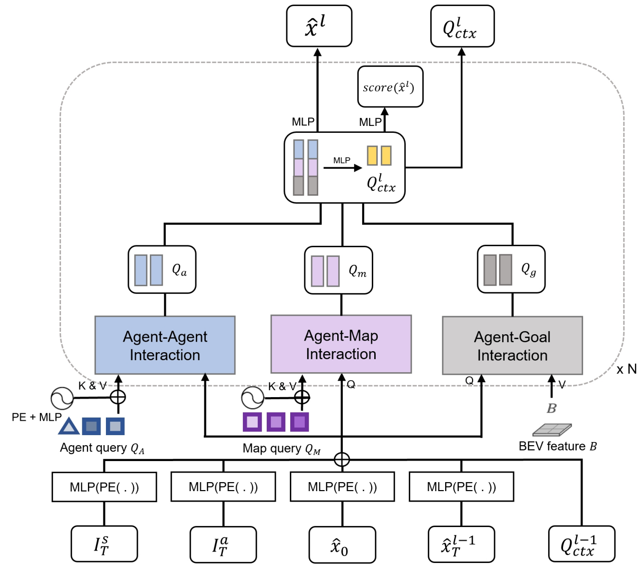

Figure 1. **MotionFormer.** It consists of N stacked agent-agent, agent-map, and agent-goal interaction transformers. The agent-agent, and agent-map interaction modules are built with standard transformer decoder layers. The agent-goal interaction module is constructed upon the deformable cross-attention module [77]. I_T^s: the end point of scene-level anchor, I_T^a: the end point of clustered agent-level anchor, $\hat{x}_0$: the agent's current position, $\hat{x}_T^{l-1}$: the predicted goal point from the previous layer, Q_{ctx}^{l-1}: query context from the previous layer.

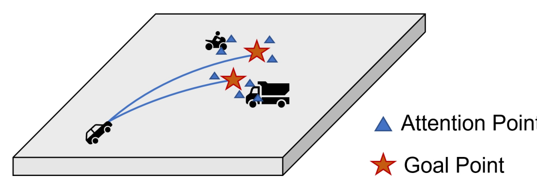

Figure 2. **Illustration of agent-goal interaction Module.** The BEV visual feature is sampled near each agent's goal points with deformable cross-attention.

As shown in Fig. 1, our MotionFormer consists of three major transformer blocks, *i.e.*, agent-agent, agent-map and agent-goal interaction modules. The agent-agent, agent-map interaction modules are built with standard transformer decoder layers, which are composed of a multi-head self-attention (MHSA) layer and a multi-head cross-attention (MHCA) layer, a feed-forward network (FFN) and several residual and normalization layers in between [5]. Apart from the agent queries Q_A and map queries Q_M, we also add the positional embeddings to those queries with sinusoidal positional embedding followed by MLP layers. The agent-goal interaction module is built upon deformable cross-attention module [77], where the goal point from the previously predicted trajectory ($R_i\hat{x}_{i,T}^{l-1} + T_i$) is adopted as the reference point, as shown in Fig. 2. Specifically, we set the number of sampled points to 4 per trajectory, and 6 trajectories per agent as we mention above. The output features of each interaction module are concatenated and projected with MLP layers to dimension $\mathcal{D} = 256$. Then, we use Gaussian Mixture Model to build each agent's trajectories, where $\hat{x}_l \in \mathcal{R}^{\mathcal{K}\times T\times 5}$. We set the prediction time horizon T to 12 (6 seconds) in UniAD. Note that we only take the first two of the last dimension (*i.e.*, x and y) as final output trajectories. Besides, the scores of each modality are also predicted ($score(\hat{x}_l) \in \mathcal{R}^{\mathcal{K}}$). We stack the overall modules for N times, and N is set to 3 in practice.

E.4. Occupancy Prediction

Given the BEV feature from upstream modules, we first downsample it by /4 with convolutional layers for efficient multi-step prediction, then pass it to our proposed OccFormer. OccFormer is composed of T_o sequential blocks shown in Fig. 3, where $T_o = 5$ is the temporal horizon (including current and future frames) and each block is responsible for generating occupancy of one specific frame. Different from prior works which are short of agent-level knowledge, our proposed method incorporates both dense scene features and sparse agent features when unrolling the future representations. The dense scene feature is from the output of the last block (or the observed feature for current frame) and it's further downscaled (/8) by a convolution layer to reduce computation for pixel-agent interaction. The sparse agent feature is derived from the concatenation of track query Q_A, agent positions P_A, and motion query Q_X, and it is then passed to a temporal-specific MLP for temporal sensitivity. We conduct pixel-level self-attention to model the long-term dependency required in some rapidly changing scenes, then perform scene-agent incorporation by attending each pixel of the scene to corresponding agents. To enhance the location alignment between agents and pixels, we restrict the cross-attention with an attention mask which is generated by a matrix multiplication between mask feature and downscaled scene feature, where the mask feature is produced by encoding agent feature with an MLP. We then upsample the attended dense feature to the same resolution as input F^{t-1} (/4) and add it with F^{t-1} as a residual connection for stability. The resulting feature F^t is both sent to the next block and a convolutional decoder for predicting occupancy at the original BEV resolution (/1). We reuse the mask feature and pass it to another MLP to form occupancy feature, and the instance-level occupancy is therefore generated by a matrix multiplication between occupancy feature and decoded dense feature F^t_{dec} (/1). Note that the MLP layer for mask feature, the MLP layer for occupancy feature, and the convolutional decoder are shared across all T_o blocks while other components are independent in each block. Dimensions of all dense features and agent features are 256 in OccFormer.

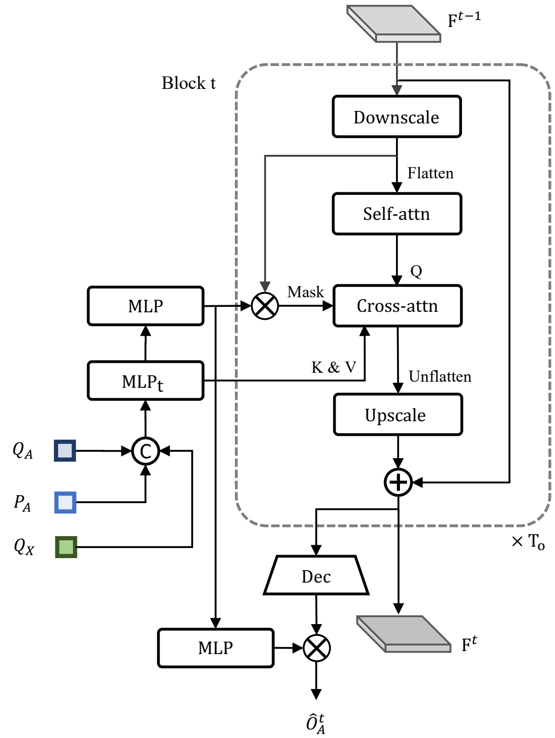

Figure 3. **OccFormer.** It is composed of T_o sequential blocks where T_o is the temporal horizon (including current and future frames) and each block is responsible for generating occupancy of one specific frame. We incorporate both dense scene features and sparse agent features, which are encoded from upstream track query Q_A, agent position P_A and motion query Q_X, to inject agent-level knowledge into future scene representations. We form instance-level occupancy $\hat{O}^t_A$ via a matrix multiplication between agent-level feature and decoded dense feature at the end of each block.

E.5. Planning

As shown in Fig. 4, our planner takes the ego-vehicle query generated from the tracking and motion forecasting module, which is symbolized with the blue triangle and yellow rectangle respectively. These two queries, along with the command embedding, are encoded with MLP layers followed by a max-pooling layer across the modality dimension, where the most salient modal features are selected and aggregated. The BEV feature interaction module is built with standard transformer decoder layers, and it is stacked for N layers, where we set $N = 3$ here. Specifically, it cross-attends the dense BEV feature with the aggregated plan query. More qualitative results can be found in Appendix F.5 showing the effectiveness of this module. To embed location information, we fuse the plan-query with learned position embedding and the BEV feature with sinusoidal positional embedding. We then regress the planning trajectory with MLP layers, which is denoted

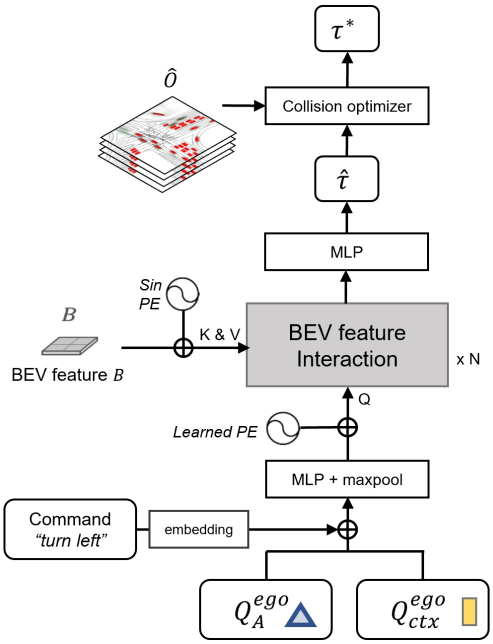

Figure 4. **Planner.** Q_A^{ego} and $Q_{\text{ctx}}^{\text{ego}}$ are the *ego-vehicle query* from the tracking and motion forecasting modules, respectively. Along with the high-level command, they are encoded with MLP layers followed by a max-pooling layer across the modality dimension, where the most salient modal features are selected and aggregated. The BEV feature interaction module is built with standard transformer decoder layers, and it is stacked for N layers.

as $\hat{\tau} \in \mathcal{R}^{T_p \times 2}$. Here we set $T_p = 6$ (3 seconds). Finally, we devise the collision optimizer for obstacle avoidance, which takes the predicted occupancy $\hat{O}$ and trajectory $\hat{\tau}$ as input as Eq. (10) in the main paper. We set $d = 5$, $\sigma = 1.0$, $\lambda_{\text{coord}} = 1.0$, $\lambda_{\text{obs}} = 5.0$.

E.6. Training Details

Joint learning. UniAD is trained in two stages which we find more stable. In **stage one**, we pre-train perception tasks including tracking and online mapping to stabilize perception predictions. Specifically, we load corresponding pre-trained BEVFormer [41] weights to UniAD for fast convergence including image backbone, FPN, BEV encoder and detection decoder except for object query embeddings (due to the additional *ego-vehicle query*). We stop the gradient back-propagation in the image backbone to reduce memory cost and train UniAD for 6 epochs with tracking and online mapping losses as follows:

$$L_1 = L_{\text{track}} + L_{\text{map}}. \tag{2}$$

In **stage two**, we keep the image backbone frozen as well, and additionally freeze BEV encoder, which is used for view transformation from image view to BEV, to further reduce memory consumption with more downstream modules. UniAD now is trained with all task losses including tracking, mapping, motion forecasting, occupancy prediction, and planning for 20 epochs (for various ablation studies in main paper, it's trained for 8 epochs for efficiency):

$$L_2 = L_{\text{track}} + L_{\text{map}} + L_{\text{motion}} + L_{\text{occ}} + L_{\text{plan}}. \tag{3}$$

Detailed losses and hyperparameters within each term of L_1 and L_2 are described below individually. The length of each training sequence (at each step) for tracking and BEV feature aggregation [41] in both stages is 5 (3 in ablation studies for efficiency).

Detection&tracking loss. Following BEVFormer [41], the *Hungarian loss* is adopted for each paired result, which is a linear combination of a Focal loss [45] for class labels and an l_1 for 3D boxes localization. In terms of the matching strategy, candidates from newborn queries are paired with ground truth objects through bipartite matching, and predictions from track queries inherit the assigned ground truth index from previous frames. Specifically, $L_{\text{track}} = \lambda_{\text{focal}} L_{\text{focal}} + \lambda_{l_1} L_{l_1}$, where $\lambda_{\text{focal}} = 2$ and $\lambda_{l_1} = 0.25$.

Online mapping loss. As in [42], this includes thing losses for lanes, dividers, and contours, also a stuff loss for the drivable area, where Focal loss is responsible for classification, L1 loss is responsible for thing bounding boxes, Dice loss and GIoU loss [56] account for segmentation. Detailedly, $L_{\text{map}} = \lambda_{\text{focal}} L_{\text{focal}} + \lambda_{l_1} L_{l_1} + \lambda_{\text{giou}} L_{\text{giou}} + \lambda_{\text{dice}} L_{\text{dice}}$, with $\lambda_{\text{focal}} = \lambda_{\text{giou}} = \lambda_{\text{dice}} = 2$ and $\lambda_{l_1} = 0.25$.

Motion forecasting loss. Like most of the prior methods, we model the multimodal trajectories as gaussian mixtures, and use the multi-path loss [9, 64], which includes a classification score loss L_{cls} and a negative log-likelihood loss term L_{nll}, and λ denotes the corresponding weight: $L_{\text{motion}} = \lambda_{\text{cls}} L_{\text{cls}} + \lambda_{\text{reg}} L_{\text{nll}}$, where $\lambda_{\text{cls}} = \lambda_{\text{reg}} = 0.5$. To ensure the temporal smoothness of trajectories, we predict agents' speed at each timestep first and accumulate it across time to obtain their final trajectories [32].

Occupancy prediction loss. The output of instance-level occupancy prediction is a binary segmentation of each agent, therefore we adopt binary cross-entropy and Dice loss [50] as the occupancy loss. Formally, $L_{\text{occ}} = \lambda_{\text{bce}} L_{\text{bce}} + \lambda_{\text{dice}} L_{\text{dice}}$, with $\lambda_{\text{bce}} = 5$ and $\lambda_{\text{dice}} = 1$ here. Additionally, since the attention mask in the pixel-agent interaction module could be seen as a coarse prediction, we employ an auxiliary occupancy loss with the same form to supervise it.

Planning loss. Safety is the most crucial factor in planning. Therefore, apart from the naive imitation l_2 loss, we employ another collision loss which keeps the planned trajectory away from obstacles as follows:

$$L_{\text{col}}(\hat{\tau}, \delta) = \sum_{i,t} \texttt{IoU}(box(\hat{\tau}_t, w + \delta, l + \delta), b_{i,t})), \quad (4)$$

$$L_{\text{plan}} = \lambda_{\text{imi}} |\hat{\tau}, \tilde{\tau}|_2 + \lambda_{\text{col}} \sum_{(\omega,\delta)} \omega L_{\text{col}}(\hat{\tau}, \delta), \quad (5)$$

where $\lambda_{\text{imi}} = 1$, $\lambda_{\text{col}} = 2.5$, (ω, δ) is a weight-value pair considering additional safety distance, $box(\hat{\tau}_t, w+\delta, l+\delta)$ represents the ego bounding box with an increased size at timestamp t to keep a larger safe distance, and $b_{i,t}$ indicates each agent forecasted in the scene. In practice, we set (ω, δ) to $(1.0, 0.0), (0.4, 0.5), (0.1, 1.0)$.

F. Experiments

F.1. Protocols

We follow most of the basic training settings as in BEVFormer [41] for both two stages with a batch size of 1, a learning rate of $2{\times}10^{-4}$, learning rate multiplier of the backbone 0.1 and AdamW optimizer [48] with a weight decay of 1×10^{-2}. The default size of BEV size is 200×200, covering BEV ranges of [-51.2m, 51.2m] for both X and Y axis with the interval as $0.512m$. More hyperparameters related to feature dimensions are shown in Table 1. Experiments are conducted with 16 NVIDIA Tesla A100 GPUs.

F.2. Metrics

Multi-object tracking. Following the standard evaluation protocols, we use **AMOTA** (Average Multi-object Tracking Accuracy), **AMOTP** (Average Multi-object Tracking Precision), **Recall**, and **IDS** (Identity Switches) to evaluate the 3D tracking performance of UniAD on nuScenes dataset. AMOTA and AMOTP are computed by integrating MOTA (Multi-object Tracking Accuracy) and MOTP (Multi-object Tracking Precision) values over all recalls:

$$\text{AMOTA} = \frac{1}{n-1} \sum_{r \in \{\frac{1}{n-1}, \frac{2}{n-1}, \ldots, 1\}} \text{MOTA}_r, \quad (6)$$

$$\text{MOTA}_r = \max(0, 1 - \frac{\text{FP}_r + \text{FN}_r + \text{IDS}_r - (1-r)\text{GT}}{r\text{GT}}), \quad (7)$$

where FP_r, FN_r, and IDS_r represent the number of false positives, false negatives and identity switches computed at the corresponding recall r, respectively. GT stands for the number of ground truth objects in this frame. AMOTP can be defined as:

$$\text{AMOTP} = \frac{1}{n-1} \sum_{r \in \{\frac{1}{n-1}, \frac{2}{n-1}, \ldots, 1\}} \frac{\sum_{i,t} d_{i,t}}{\text{TP}_r}, \quad (8)$$

where $d_{i,t}$ denotes the position error (in x and y axis) of matched track i at time stamp t, and TP_r is the number of true positives at the corresponding recall r.

Online mapping. We have four categories for the online mapping task, *i.e.*, lanes, boundaries, pedestrian crossings and drivable area. We calculate the intersection-over-union (**IoU**) metric for each class between the network outputs and ground truth maps.

Motion forecasting. On one hand, following the standard motion prediction protocols, we adopt conventional metrics, including **minADE** (minimum Average Displacement Error), **minFDE** (minimum Final Displacement Error) and **MR** (Miss Rate). Similar to the prior works [43, 49, 53], these metrics are only calculated within matched TPs, and we set the matching threshold to $1.0m$ in all of our experiments. As for the MR, we set the miss FDE threshold to $2.0m$. On the other hand, we also employ recently proposed end-to-end metrics, *i.e.*, **EPA** (End-to-end Prediction Accuracy) [23] and **minFDE-AP** [53]. For EPA, we use the same setting as in ViP3D [23] for a fair comparison. For minFDE-AP, we do not separate ground truth into multiple bins (static, linear, and non-linearly moving sub-categories) for simplicity. Specifically, only when an object's perception location and its min-FDE are within the distance threshold ($1.0m$ and $2.0m$ respectively), it would be counted as a TP for the AP (average precision) calculation. Similarly to the prior works, we merge the car, truck, construction vehicle, bus, trailer, motorcycle, and bicycle as the vehicle category, and all the motion forecasting metrics provided in the experiments are evaluated on the vehicle category.

Occupancy prediction. We evaluate the quality of predicted occupancy in both whole-scene level and instance-level following [27, 74]. Specifically, The **IoU** measures the whole-scene categorical segmentations which is instance-agnostic, while the Video Panoptic Quality (**VPQ**) [37] takes into account each instance's presence and consistency over time. The VPQ metric is calculated as follows:

$$\text{VPQ} = \sum_{t=0}^{H} \frac{\sum_{(p_t,q_t)\in \text{TP}_t} \texttt{IoU}(p_t, q_t)}{|\text{TP}_t| + \frac{1}{2}|\text{FP}_t| + \frac{1}{2}|\text{FN}_t|}, \quad (9)$$

where H is the future horizon and we set $H = 4$ (which leads to $T_o = 5$ including the current timestamp) as in [27, 74], covering $2.0s$ consecutive data at 2Hz. TP_t, FP_t, and FN_t are the set of true positives, false positives, and false

Methods	Encoder	Tracking			Mapping		Motion Forecasting				Occupancy Prediction				Planning	
		AMOTA↑	AMOTP↓	IDS↓	IoU-lane↑	IoU-road↑	minADE↓	minFDE↓	MR↓	EPA↑	IoU-n.↑	IoU-f.↑	VPQ-n.↑	VPQ-f.↑	avg.L2↓	avg.Col.↓
UniAD-S	R50	0.241	1.488	958	0.315	0.689	0.788	1.126	0.156	0.381	59.4	35.6	49.2	28.9	1.04	0.32
UniAD-B	R101	0.359	1.320	**906**	0.313	0.691	**0.708**	**1.025**	**0.151**	0.456	63.4	40.2	54.7	33.5	1.03	0.31
UniAD-L	V2-99*	**0.409**	**1.259**	1583	**0.323**	**0.709**	0.723	1.067	0.158	**0.508**	**64.1**	**42.6**	**55.8**	**36.9**	**1.03**	**0.29**

Table 2. **Comparisons between three variations of UniAD.** $*$: pre-trained with extra depth data [52].

ID	Det.	Track	Map	Motion	Occ.	Plan	#Params	FLOPs	FPS
0 [74]	✓		✓		✓		102.5M	1921G	-
1	✓						65.9M	1324G	4.2
2	✓	✓					68.2M	1326G	2.7
3	✓	✓	✓				95.8M	1520G	2.2
4	✓	✓	✓	✓			108.6M	1535G	2.1
5	✓	✓	✓	✓	✓		122.5M	1701G	2.0
6	✓	✓	✓	✓	✓	✓	125.0M	1709G	1.8

Table 3. **Computational complexity and runtime** with different modules incorporated. ID.1 is similar to original BEVFormer [41], and ID. 0 (BEVerse-Tiny) [74] is an MTL framework.

negatives at timestamp t respectively. Both two metrics are evaluated under two different BEV ranges, **near** ("-n.") for $30m \times 30m$ and **far** ("-f.") for $100m \times 100m$ around the ego vehicle. We evaluate the results of the current step ($t = 0$) and the future 4 steps together on both metrics.

Planning. We adopt the same metrics as in ST-P3 [29], *i.e.*, **L2 error** and **collision rate** at various timestamps.

F.3. Model complexity and Computational cost

We measure the complexity of UniAD and runtime on an Nvidia Tesla A100 GPU, as depicted in Table 3. Though the decoder part of tasks brings a certain amount of parameters, the computational complexity mainly comes from the encoder part, compared to the original BEVFormer detector (ID. 1). We also provide a comparison with the recent BEVerse [74]. UniAD owns more tasks, achieves superior performance, and has *lower* FLOPs - indicating affordable budget to additional computation cost.

F.4. Model scale

We provide three variations of UniAD under different model scales as shown in Table 2. The chosen image backbones for image-view feature extraction are ResNet-50 [24], ResNet-101 and VoVNet 2-99 [38] for UniAD-S, UniAD-B and UniAD-L respectively. Since the model scale (image encoder) mainly influences the BEV feature quality, we could observe that the perceptual scores improve with a larger backbone, which further could lead to better prediction and planning performance.

F.5. Qualitative results

Attention mask visualization. To investigate the internal mechanism and show its explainability, we visualize the attention mask of the cross-attention module in the planner. As shown in Fig. 5, the predicted tracking bounding boxes, planned trajectory, and the ground truth HD Map are rendered for reference, and the attention mask is overlayered on top. From left to right, we show two consecutive frames in a time sequence but with different navigation commands. We can observe that the planned trajectory varies largely according to the command. Also, much attention is paid to the goal lane as well as the critical agents that are yielding to our ego vehicle.

Visualization of different scenarios. We provide visualizations for more scenarios, including cruising around the urban areas (Fig. 6), critical cases (Fig. 7), and obstacle avoidance scenarios (Fig. 8). One promising evidence for our planning-oriented design is shown in Fig. 9, where inaccurate results occur in prior modules while the later tasks could still recover. Similarly, we show results for all tasks in surround-view images, BEV, as well as the attention mask from the planner. A demo video[1] is also provided for reference.

Failure cases are essential for an autonomous driving algorithm to understand its weakness and guide future work, and here we present some failure cases of UniAD. The failure cases of UniAD are mainly under some long-tail scenarios where all modules are affected, as depicted in Fig. 10 and Fig. 11.

[1] https://opendrivelab.github.io/UniAD/

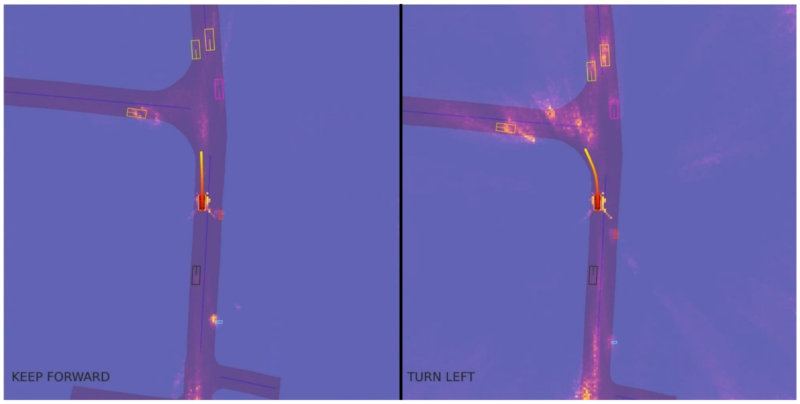

Figure 5. **Effectiveness of navigation command and attention mask visualization.** Here we demonstrate how attention is paid in accordance with the navigation command. We render the attention mask from the BEV interaction module in the planning module, the predicted tracking bounding boxes as well as the planned trajectory. The navigation command is printed on the bottom left, and the HD Map is rendered only for reference. From left to right, we show two consecutive frames in a time sequence but with different navigation commands. We can observe that the planned trajectory varies largely according to the command. Also, much attention is paid to the goal lane as well as the critical agents that are yielding to our ego vehicle.

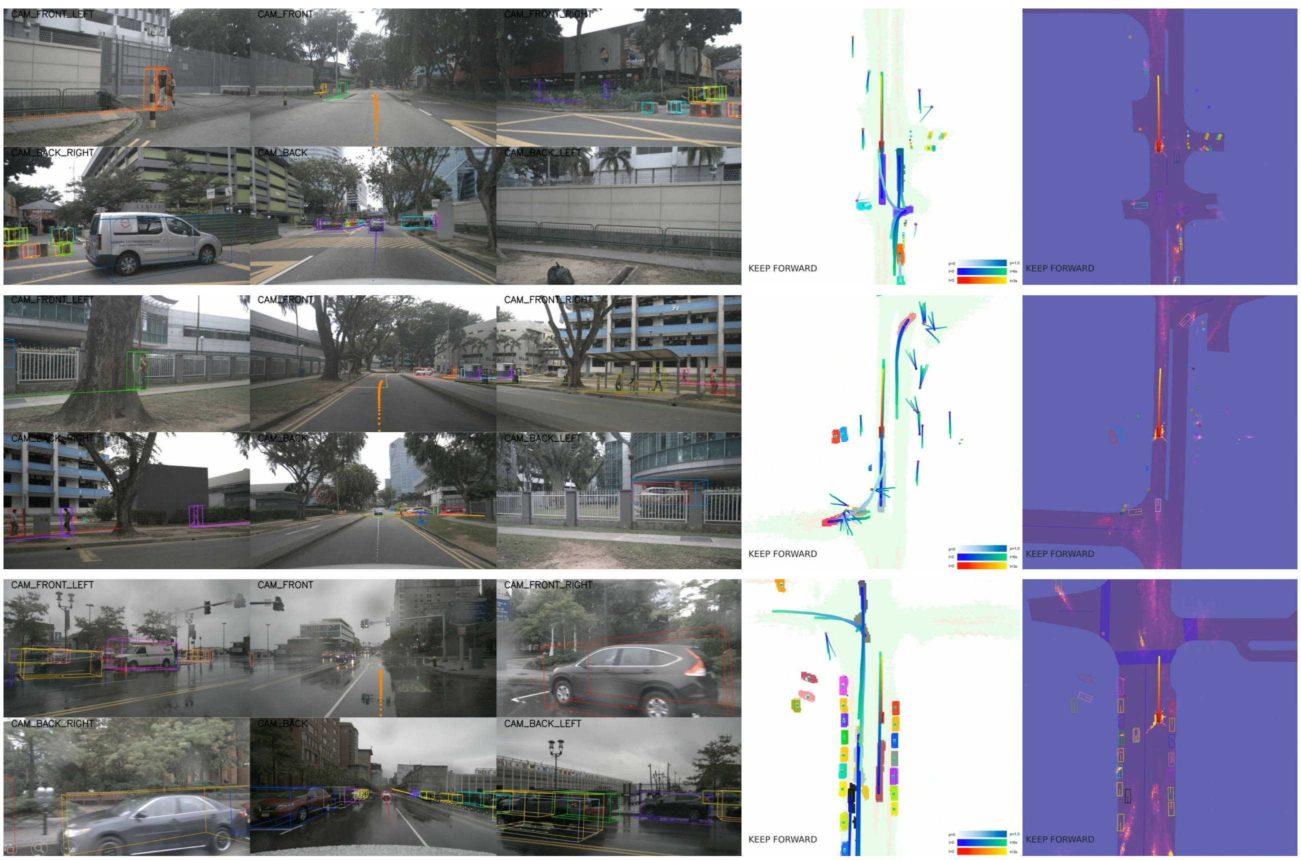

Figure 6. **Visualization for cruising around the urban areas.** UniAD can generate high-quality interpretable perceptual and prediction results, and make a safe maneuver. The first three columns show six camera views, and the last two columns are the predicted results and the attention mask from the planning module respectively. Each agent is illustrated with a unique color. Only top-1 and top-3 trajectories from motion forecasting are selected for visualization on images-view and BEV respectively.

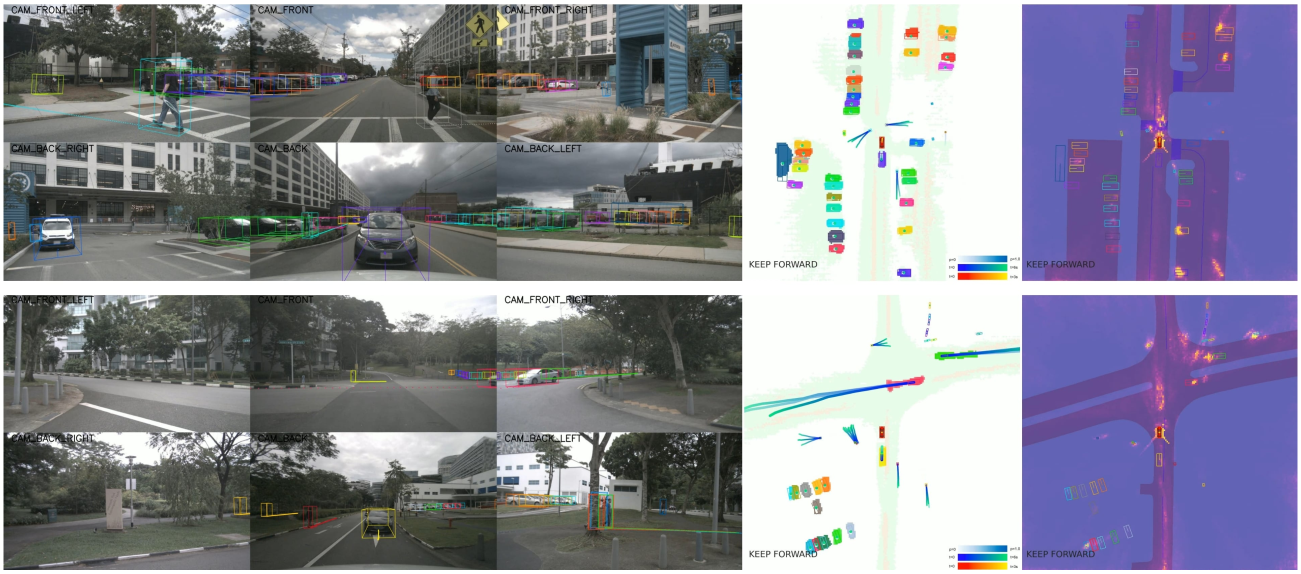

Figure 7. **Critical case visualization.** Here we demonstrate two critical cases. The first scenario (top) shows that the ego vehicle is yielding to two pedestrians crossing the street, and the second scenario (down) shows that the ego vehicle is yielding to a fast-moving car and waiting to go straight without protection near an intersection. We can observe that much attention is paid to the most critical agents, *i.e.*, pedestrians and fast-moving vehicles, as well as the intended goal location.

Figure 8. **Obstacles avoidance visualization.** In these two scenarios, the ego vehicle is changing lanes attentively to avoid the obstacle vehicle. From the attention mask, we can observe that our method focuses on the obstacles as well as the road in the front and back.

Figure 9. **Visualization for planning recovering from perception failures.** We show an interesting case where inaccurate results occur in prior modules while the later tasks could still recover. The top row and the down row represent two consecutive frames from the same scenario. The vehicle in the red circle is moving from a far distance toward the intersection at a high speed. It is observed that the tracking module misses it at first, then captures it at the latter frame. The blue circles show a stationary car yielding to the traffic, and it is missed in both frames. Interestingly, both vehicles show strong reactions to the attention masks of the planner, even though they are missed in the prior modules. It means that our planner still pays attention to those critical though missed agents, which is intractable in previous fragmented and non-unified driving systems, and demonstrates the robustness of UniAD.

Figure 10. **Failure cases 1.** Here we present a long-tail scenario, where a large trailer with a white container occupies the entire road. We can observe that our tracking module fails to detect the accurate size of the front trailer and heading angles of vehicles beside the road.

Figure 11. **Failure cases 2.** In this case, the planner is over-cautious about the incoming vehicle in the narrow street. The dark environment is one critical type of long-tail scenarios in autonomous driving. Applying smaller collision loss weight and more regulation regarding the boundary might mitigate the problem.

References

[1] Adil Kaan Akan and Fatma Güney. StretchBEV: Stretching future instance prediction spatially and temporally. In *ECCV*, 2022. 3

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

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

[4] Thibault Buhet, Émilie Wirbel, and Xavier Perrotton. PLOP: Probabilistic polynomial objects trajectory planning for autonomous driving. In *CoRL*, 2020. 3

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2, 3, 5

[6] Sergio Casas, Cole Gulino, Renjie Liao, and Raquel Urtasun. Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data. In *ICRA*, 2020. 2

[7] Sergio Casas, Wenjie Luo, and Raquel Urtasun. Intentnet: Learning to predict intention from raw sensor data. In *CoRL*, 2018. 2

[8] Sergio Casas, Abbas Sadat, and Raquel Urtasun. Mp3: A unified model to map, perceive, predict and plan. In *CVPR*, 2021. 3

[9] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. In *CoRL*, 2020. 3, 7

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

[11] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. Learning to drive from a world on rails. In *ICCV*, 2021. 3

[12] Dian Chen and Philipp Krähenbühl. Learning from all vehicles. In *CVPR*, 2022. 3

[13] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *CoRL*, 2020. 3

[14] Kashyap Chitta, Aditya Prakash, and Andreas Geiger. NEAT: Neural attention fields for end-to-end autonomous driving. In *ICCV*, 2021. 3

[15] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE TPAMI*, 2022. 3

[16] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *ICRA*, 2018. 3

[17] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *ICCV*, 2019. 3

[18] Alexander Cui, Sergio Casas, Abbas Sadat, Renjie Liao, and Raquel Urtasun. Lookout: Diverse multi-future prediction and planning for self-driving. In *ICCV*, 2021. 3

[19] Nemanja Djuric, Henggang Cui, Zhaoen Su, Shangxuan Wu, Huahua Wang, Fang-Chieh Chou, Luisa San Martin, Song Feng, Rui Hu, Yang Xu, Alyssa Dayan, Sidney Zhang, Brian C. Becker, Gregory P. Meyer, Carlos Vallespi-Gonzalez, and Carl K. Wellington. Multixnet: Multiclass multistage multimodal motion prediction. In *IV*, 2021. 2

[20] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, Zoey Yang, Aurélien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *ICCV*, 2021. 2

[21] Sudeep Fadadu, Shreyash Pandey, Darshan Hegde, Yi Shi, Fang-Chieh Chou, Nemanja Djuric, and Carlos Vallespi-Gonzalez. Multi-view fusion of sensor data for improved perception and prediction in autonomous driving. In *WACV*, 2022. 2

[22] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *CVPR*, 2020. 2

[23] Junru Gu, Chenxu Hu, Tianyuan Zhang, Xuanyao Chen, Yilun Wang, Yue Wang, and Hang Zhao. ViP3D: End-to-end visual trajectory prediction via 3d agent queries. In *CVPR*, 2023. 2, 8

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 9

[25] Noureldin Hendy, Cooper Sloan, Feng Tian, Pengfei Duan, Nick Charchut, Yuesong Xie, Chuang Wang, and James Philbin. Fishing net: Future inference of semantic heatmaps in grids. *arXiv preprint arXiv:2006.09917*, 2020. 3

[26] Anthony Hu, Gianluca Corrado, Nicolas Griffiths, Zak Murez, Corina Gurau, Hudson Yeo, Alex Kendall, Roberto Cipolla, and Jamie Shotton. Model-based imitation learning for urban driving. In *NeurIPS*, 2022. 3

[27] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. FIERY: Future instance prediction in bird's-eye view from surround monocular cameras. In *ICCV*, 2021. 3, 8

[28] Peiyun Hu, Aaron Huang, John Dolan, David Held, and Deva Ramanan. Safe local motion planning with self-supervised freespace forecasting. In *CVPR*, 2021. 3

[29] Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. ST-P3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *ECCV*, 2022. 3, 9

[30] Zhiyu Huang, Haochen Liu, Jingda Wu, and Chen Lv. Differentiable integrated motion prediction and planning with learnable cost function for autonomous driving. *arXiv preprint arXiv:2207.10422*, 2022. 3

[31] Boris Ivanovic, Amine Elhafsi, Guy Rosman, Adrien Gaidon, and Marco Pavone. MATS: An interpretable trajectory forecasting representation for planning and control. In *CoRL*, 2021. 3
[32] Xiaosong Jia, Li Chen, Penghao Wu, Jia Zeng, Junchi Yan, Hongyang Li, and Yu Qiao. Towards capturing the temporal dynamics for trajectory prediction: a coarse-to-fine approach. In *CoRL*, 2022. 2, 7
[33] Xiaosong Jia, Liting Sun, Masayoshi Tomizuka, and Wei Zhan. Ide-net: Interactive driving event and pattern extraction from human data. *IEEE RA-L*, 2021. 2
[34] Alexey Kamenev, Lirui Wang, Ollin Boer Bohan, Ishwar Kulkarni, Bilal Kartal, Artem Molchanov, Stan Birchfield, David Nistér, and Nikolai Smolyanskiy. Predictionnet: Real-time joint probabilistic traffic prediction for planning, control, and simulation. In *ICRA*, 2022. 3
[35] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *ICRA*, 2019. 3
[36] Tarasha Khurana, Peiyun Hu, Achal Dave, Jason Ziglar, David Held, and Deva Ramanan. Differentiable raycasting for self-supervised occupancy forecasting. In *ECCV*, 2022. 3
[37] Dahun Kim, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Video panoptic segmentation. In *CVPR*, 2020. 8
[38] Youngwan Lee, Joong-won Hwang, Sangrok Lee, Yuseok Bae, and Jongyoul Park. An energy and gpu-computation efficient backbone network for real-time object detection. In *CVPR Workshop*, 2019. 9
[39] Lingyun Luke Li, Bin Yang, Ming Liang, Wenyuan Zeng, Mengye Ren, Sean Segal, and Raquel Urtasun. End-to-end contextual perception and prediction with interaction transformer. In *IROS*, 2020. 2
[40] Yanwei Li, Yilun Chen, Xiaojuan Qi, Zeming Li, Jian Sun, and Jiaya Jia. Unifying voxel-based representation with transformer for 3d object detection. In *NeurIPS*, 2022. 2
[41] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. BEVFormer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*, 2022. 3, 7, 8, 9
[42] Zhiqi Li, Wenhai Wang, Enze Xie, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, Ping Luo, and Tong Lu. Panoptic segformer: Delving deeper into panoptic segmentation with transformers. In *CVPR*, 2022. 5, 7
[43] Ming Liang, Bin Yang, Wenyuan Zeng, Yun Chen, Rui Hu, Sergio Casas, and Raquel Urtasun. Pnpnet: End-to-end perception and prediction with tracking in the loop. In *CVPR*, 2020. 2, 8
[44] Xiaodan Liang, Tairui Wang, Luona Yang, and Eric Xing. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. In *ECCV*, 2018. 3
[45] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 7
[46] Jerry Liu, Wenyuan Zeng, Raquel Urtasun, and Ersin Yumer. Deep structured reactive planning. In *ICRA*, 2021. 3
[47] Zhijian Liu, Haotian Tang, Alexander Amini, Xingyu Yang, Huizi Mao, Daniela Rus, and Song Han. BEVFusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. In *ICRA*, 2023. 2
[48] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2018. 8
[49] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In *CVPR*, 2018. 2, 8
[50] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, 2016. 7
[51] Jiquan Ngiam, Benjamin Caine, Vijay Vasudevan, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, David Weiss, Ben Sapp, Zhifeng Chen, and Jonathon Shlens. Scene transformer: A unified multi-task model for behavior prediction and planning. In *ICLR*, 2022. 3
[52] Dennis Park, Rares Ambrus, Vitor Guizilini, Jie Li, and Adrien Gaidon. Is pseudo-lidar needed for monocular 3d object detection? In *ICCV*, 2021. 9
[53] Neehar Peri, Jonathon Luiten, Mengtian Li, Aljoša Ošep, Laura Leal-Taixé, and Deva Ramanan. Forecasting from lidar via future object detection. In *CVPR*, 2022. 2, 8
[54] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *NeurIPS*, 1988. 3
[55] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *CVPR*, 2021. 3
[56] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, 2019. 7
[57] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. PRECOG: Prediction conditioned on goals in visual multi-agent settings. In *ICCV*, 2019. 3
[58] Abbas Sadat, Sergio Casas, Mengye Ren, Xinyu Wu, Pranaab Dhawan, and Raquel Urtasun. Perceive, predict, and plan: Safe motion planning through interpretable semantic representations. In *ECCV*, 2020. 2, 3
[59] Hao Shao, Letian Wang, Ruobing Chen, Hongsheng Li, and Yu Liu. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In *CoRL*, 2022. 3
[60] Yining Shi, Jingyan Shen, Yifan Sun, Yunlong Wang, Jiaxin Li, Shiqi Sun, Kun Jiang, and Diange Yang. Srcn3d: Sparse r-cnn 3d surround-view camera object detection and tracking for autonomous driving. *arXiv preprint arXiv:2206.14451*, 2022. 2
[61] Haoran Song, Wenchao Ding, Yuxuan Chen, Shaojie Shen, Michael Yu Wang, and Qifeng Chen. Pip: Planning-informed trajectory prediction for autonomous driving. In *ECCV*, 2020. 3
[62] Sebastian Thrun and Arno Bücken. Integrating grid-based and topological maps for mobile robot navigation. In *AAAI*, 1996. 2

[63] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *CVPR*, 2020. 3

[64] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled S Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi Pang Lam, Dragomir Anguelov, and Benjamin Sapp. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. *arXiv preprint arXiv:2111.14973*, 2021. 7

[65] Bob Wei, Mengye Ren, Wenyuan Zeng, Ming Liang, Bin Yang, and Raquel Urtasun. Perceive, attend, and drive: Learning spatial attention for safe self-driving. In *ICRA*, 2021. 3

[66] Pengxiang Wu, Siheng Chen, and Dimitris N Metaxas. Motionnet: Joint perception and motion prediction for autonomous driving based on bird's eye view maps. In *CVPR*, 2020. 3

[67] Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. In *NeurIPS*, 2022. 3

[68] Jinrong Yang, En Yu, Zeming Li, Xiaoping Li, and Wenbing Tao. Quality matters: Embracing quality clues for robust 3d multi-object tracking. *arXiv preprint arXiv:2208.10976*, 2022. 2

[69] Fangao Zeng, Bin Dong, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. In *ECCV*, 2021. 3

[70] Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun. End-to-end interpretable neural motion planner. In *CVPR*, 2019. 2, 3

[71] Wenyuan Zeng, Shenlong Wang, Renjie Liao, Yun Chen, Bin Yang, and Raquel Urtasun. Dsdnet: Deep structured self-driving network. In *ECCV*, 2020. 3

[72] Jimuyang Zhang and Eshed Ohn-Bar. Learning by watching. In *CVPR*, 2021. 3

[73] Tianyuan Zhang, Xuanyao Chen, Yue Wang, Yilun Wang, and Hang Zhao. MUTR3D: A Multi-camera Tracking Framework via 3D-to-2D Queries. In *CVPR Workshop*, 2022. 2

[74] Yunpeng Zhang, Zheng Zhu, Wenzhao Zheng, Junjie Huang, Guan Huang, Jie Zhou, and Jiwen Lu. BEVerse: Unified perception and prediction in birds-eye-view for vision-centric autonomous driving. *arXiv preprint arXiv:2205.09743*, 2022. 3, 8, 9

[75] Zhejun Zhang, Alexander Liniger, Dengxin Dai, Fisher Yu, and Luc Van Gool. End-to-end urban driving by imitating a reinforcement learning coach. In *ICCV*, 2021. 3

[76] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning Chai, Cordelia Schmid, Congcong Li, and Dragomir Anguelov. TNT: Target-driven trajectory prediction. In *CoRL*, 2020. 2

[77] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2020. 3, 5