LATR: 3D Lane Detection from Monocular Images with Transformer

Yueru Luo[1,2] Chaoda Zheng[1,2] Xu Yan[1,2] Tang Kun[3] Chao Zheng[3]
Shuguang Cui[2,1] Zhen Li[2,1]*
[1] FNii, CUHK-Shenzhen [2] SSE, CUHK-Shenzhen [3] Tencent Map, T Lab

Abstract

*3D lane detection from monocular images is a fundamental yet challenging task in autonomous driving. Recent advances primarily rely on structural 3D surrogates (*e.g.*, bird's eye view) built from front-view image features and camera parameters. However, the depth ambiguity in monocular images inevitably causes misalignment between the constructed surrogate feature map and the original image, posing a great challenge for accurate lane detection. To address the above issue, we present a novel* ***LATR*** *model, an end-to-end 3D lane detector that uses 3D-aware front-view features without transformed view representation. Specifically, LATR detects 3D lanes via cross-attention based on query and key-value pairs, constructed using our lane-aware query generator and dynamic 3D ground positional embedding. On the one hand, each query is generated based on 2D lane-aware features and adopts a hybrid embedding to enhance the lane information. On the other hand, 3D space information is injected as positional embedding from an iteratively-updated 3D ground plane. LATR outperforms previous state-of-the-art methods on both synthetic Apollo and realistic OpenLane and ONCE-3DLanes by large margins (*e.g.*,* ***11.4*** *gain in terms of F1 score on OpenLane). Code will be released at* https://github.com/JMoonr/LATR.

1. Introduction

3D Lane Detection is critical for various applications in autonomous driving, such as trajectory planning and lane keeping [48]. Despite the remarkable progress of LiDAR-based methods in other 3D perception tasks [56, 13], recent advances in 3D lane detection prefer using a monocular camera since it owns desirable advantages compared to LiDARs. Apart from the low deployment cost, cameras offer a longer perception range compared to other sensors and can produce high-resolution images with rich textures, which are crucial for detecting slim and long-span lanes.

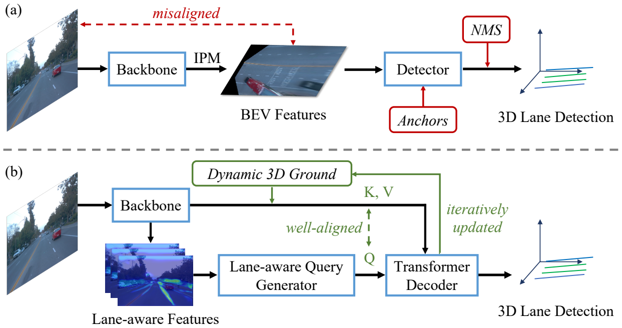

Figure 1. (a) Previous methods mainly utilize camera parameters and inverse perspective mapping (IPM) to transform the features into a surrogate space (*e.g.*, BEV), and further perform 3D lane detection through anchors and non-maximum suppression (NMS). (b) We propose the novel LATR, an anchor-free and NMS-free Transformer architecture, to perform 3D lane detection right on the front view. Using lane-aware queries and dynamic 3D ground positional embedding, our model produces well-aligned 3D features and achieves superior 3D lane detection.

Due to the lack of depth information, detecting 3D lanes from monocular images is challenging. A straightforward solution is to reconstruct the 3D lane layout based on 2D segmentation results and per-pixel depth estimation, as proposed in SALAD [49]. However, this method requires high-quality depth data for training and heavily relies on the precision of the estimated depth. Alternatively, CurveFormer [1] employs polynomials to model the 3D lane from the front view. While it avoids indefinite view transformation, the polynomial form adopted in their design restricts the flexibility of capturing diverse lane shapes. In contrast, current mainstream methods favor the utilization of 3D surrogate representations [7, 5, 8, 23, 3, 14]. These surrogate representations are constructed based on front-view image features and camera parameters, with no reliance on depth information. Since lanes inherently reside on the road, most of these methods build the 3D surrogate by projecting the image features into a bird's eye view (BEV) via inverse perspective mapping (IPM) [31]. However, IPM is strictly based on the flat ground assumption, thereby introducing misalignment between the 3D surrogate and the original image in many real driving scenarios (*e.g.*, uphill/downhill and

[*] Corresponding author

bumps). This misalignment, entangled with distortions, inevitably hinders the accurate estimation of the road structure and endangers driving safety. Despite attempts [3] made to relieve this issue by introducing deformable attention [57], the problem of misalignment remains unresolved.

Based on the above observation, we aim to improve 3D lane detection by *directly locating 3D lanes from the front view without any intermediate 3D surrogates through lane-aware queries*. Inspired by the 2D object detector DETR [2], we streamline lane detection as an end-to-end set prediction problem, forming **LA**ne detection **TR**ansformer (**LATR**). LATR detects 3D lanes from front view images using *lane-aware queries* and *dynamic 3D ground positional embedding*. We devise a lane representation scheme to describe lane queries, better capturing the properties of 3D lanes. Besides, we utilize lane-aware features to offer queries rich semantic and spatial priors. Since pure front-view features lack awareness of 3D space, we inject 3D positional information from a hypothetical 3D ground into the front-view features. This hypothetical ground, initialized as a horizontal grid, undergoes iterative optimization to fit the ground truth road. Finally, the lane-aware queries interact with 3D-aware features through a transformer decoder, followed by MLPs to produce 3D lane predictions.

Our main contributions are the following:

- We propose **LATR**, an end-to-end 3D lane detection framework based on Transformer. By directly detecting 3D lanes from the front view, without using any 3D surrogate representations, LATR offers efficiency and avoids feature misalignment present in prior methods.
- We introduce a lane-aware query generator that uses dynamically extracted lane-aware features to initialize query embeddings. Moreover, a dynamic positional embedding is proposed to bridge 3D space and 2D images, which derives from a constructed 3D ground that is iteratively updated under supervision.
- We conduct thorough experiments on the benchmark datasets of OpenLane, Apollo, and ONCE-3DLanes. Our proposed LATR outperforms previous SoTA methods by large margins (**+11.4** improvement on OpenLane, **+4.3** on Apollo, and **+6.26** on ONCE-3DLanes *w.r.t.* F1 score).

2. Related Work

2.1. 2D Lane Detection

Benefiting from deep learning, 2D lane detection has made significant progress. Primary approaches formulate the problem in four manners. *1) Segmentation-based* methods primarily resort to pixel-wise classification to generate lane masks [34, 33, 54], which are further post-processed to form a set of lanes through specific designs. *2) Anchor-based* approaches mainly leverage line-like anchors to regress the offset relative to targets [15, 42, 55]. To break the limitations of line-shape anchors, [11] employs eigenlane space to generate diverse lane candidates. Besides, row-based anchors are heuristically devised to classify row-wise pixels into lanes [51, 36, 20]. These row anchors are further extended to hybrid (row and column) anchors in [37], attempting to mitigate the localization errors for side lanes. *3) Keypoint-based* approaches are proposed to more flexibly model lane positions [12, 38, 45, 11], which first estimate locations of points and then group points into distinct lanes using different schemes. *4) Curve-based* approaches [44, 43, 23, 6] aim to fit lane curves via various curve formulations. Despite these promising advancements in 2D lane detection, a non-negligible gap persists between the 2D results and the requirements of real-world application scenarios, *i.e.*, accurate 3D positions.

2.2. 3D Lane Detection

To acquire accurate 3D positions in realistic scenarios, 3D lane detection was proposed to drive further related studies. Compared to the 2D lane task, nonetheless, there are fewer studies to explore the model design. Prevalent methods [7, 5, 8, 23, 3, 14] attempt to transform the 2D features into a surrogate 3D space based on IPM, operating under the flat road assumption. However, this assumption is easily broken when encountering uphill/downhill, which is common in driving scenarios. Thus, the latent feature representation becomes entangled with unexpected distortions caused by variations in road height, which impairs the model's reliability and jeopardizes traffic safety.

To tackle this problem, SALAD [49] employs front-view image segmentation, utilizing the obtained results to generate 3D lanes with the aid of depth estimation. However, their training demands dense depth annotation, and their performance hinges on the accuracy of the estimated depth. GP [14] employs 3D lane reconstruction to achieve detection. To preserve the 3D lane structure, they explicitly impose geometric constraints on intra-lane and inter-lane relationships under sophisticatedly designed supervision. M^2-3Dlane [30], instead, introduces LiDAR data to facilitate the monocular 3D lane detection. By leveraging LiDAR point clouds, they lift image features into 3D space and further fuse multi-modal features in BEV space. A concurrent work CurveFormer [1] also formulates detection within the perspective space under an end-to-end paradigm. However, it represents lanes parametrically and forces the model to implicitly learn 3D positions. While the polynomial formulation could provide a compact lane structure, it suffers from parameter sensitivity and lacks the flexibility to capture diverse lanes in realistic driving scenarios.

Different from previous works, LATR relies only on the front view and the monocular image to achieve end-to-end 3D lane detection. It establishes both intra-lane and inter-lane geometric relationships without imposing hard con-

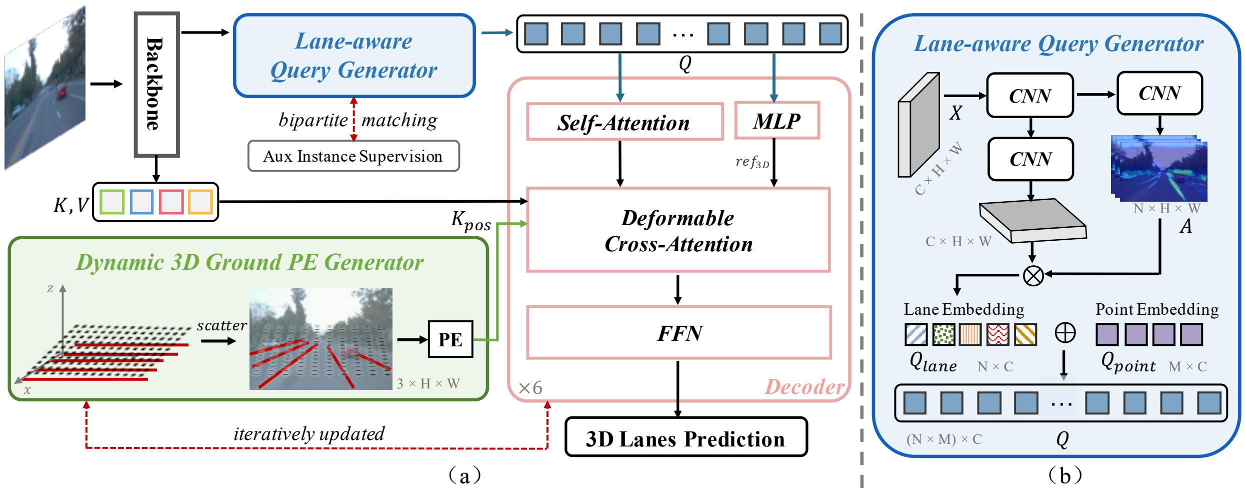

Figure 2. **The overall architecture.** LATR is a novel 3D lane detection framework that utilizes a Transformer-based approach, as shown in part (a). Specifically, the front-view image is first processed by the backbone network. Subsequently, a Lane-aware Query Generator generates queries that incorporate lane-level and point-level embeddings, as illustrated in (b). In addition, dynamic 3D ground positional embeddings are obtained through iterative refinement of a hypothetical 3D ground plane which is used to capture 3D information.

straints [14]. Our queries are endowed with lane-aware information from a global perspective, which further integrates with dynamic 3D ground features to predict 3D lanes.

2.3. Object Detection with Transformers

DETR [2] stands as a pioneer in adopting the end-to-end Transformer architecture for object detection, eliminating the need for heuristic designs in previous CNN-based methods, *e.g*., anchors, NMS, and instead employing learnable queries to target potential objects. Recently, a number of subsequent studies have embraced the end-to-end framework. To bridge 3D space and 2D image, monocular methods usually require depth estimation for the entire image [10, 35], or foreground [53], or object centers [29], while multi-view ones involve dense 3D space reconstruction [16, 17, 28] or the generation of 3D frustums [25, 26].

Furthermore, existing query-based object detection efforts [2, 57, 16, 17, 28] initialize queries either randomly, or by integrating tailored anchor designs [47, 24] for 2D objects. For 3D objects, approaches involve generating 3D frustum embeddings [25, 26] to provide localization information. However, applying these techniques directly to 3D lane detection brings certain limitations. First, the optimization of ambiguous learnable queries is challenging [41] and adversely affects query feature aggregation [47, 24]. Moreover, the static nature of the learned queries hampers their ability to generalize effectively. Second, due to the significant disparity between compact objects and slender lanes, those tailored designs [47, 24] are not suitable (*e.g*., anchor box-based query) or sub-optimal [52, 25] (*e.g*., frustum in 3D space) for lane detection.

3. Method

Given an input image $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$, 3D lane detection aims to predict the 3D position of lanes within it. Lanes are represented by a collection of 3D points, denoted as $\mathbf{Y} = \{\mathbf{L}_i | i \in 1, ..., N\}$, where N is the number of lanes in the image, and $\mathbf{L}_i$ denotes the i-th lane. Each lane $\mathbf{L}_i = (\mathbf{P}_i, \mathcal{C}_i)$ consists of a set of points $\mathbf{P}_i = \{(x_i^j, y_i^j, z_i^j)\}_{j=1}^M$ that construct the lane, where M is a predetermined cardinality of the output point set and $\mathcal{C}_i$ indicates the category. Normally, $y^{\{*\}}$ is set as a predefined longitudinal coordinate $\mathbf{Y}_{ref} = \{y^i\}_{i=1}^M$ [7, 8, 3].

3.1. Overall Architecture

The overall architecture of LATR is illustrated in Fig. 2. We first extract the feature map $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$ from the input image using a 2D backbone. Afterward, we generate lane-aware queries $\mathbf{Q} \in \mathbb{R}^{(N\times M)\times C}$ using the Lane-aware Query Generator, with N denoting the number of lanes and each lane being depicted by M points. Here, $(N \times M)$ indicates flattened channels along corresponding dimensions. Subsequently, the lane-aware queries $\mathbf{Q}$ aggressively interact with the feature map $\mathbf{X}$ via deformable attention [57]. Without constructing any misaligned 3D surrogates, we propose the Dynamic 3D Ground Positional Embedding (PE) Generator to enhance 2D features with 3D awareness during the deformable attention. Finally, we apply a prediction head on the updated queries to obtain the final lane prediction. Details of each component will be given in the following subsections.

3.2. Lane-aware Query Generator

Instead of adopting fixed learned features as queries in previous methods [2, 17, 25, 57], we propose a dynamic scheme to generate lane-aware query embeddings, which provide queries with valuable 2D lane priors derived from image features. Moreover, to offer queries more flexibility in portraying lanes and implicitly modeling intra-lane and inter-lane relations, we represent the query embedding with diverse granularities, lane-level and point-level.

The lane-level embedding captures the entire structure of each lane, while the point-level embedding gathers local features situated at $y^i \in \mathbf{Y}_{ref}$ as stated above. We then concert these two levels as our final query embeddings. This hybrid embedding scheme encodes queries with discriminative features from different lanes, and enables each query to capture shared patterns at a particular depth by sharing the point-level embedding. The intuition here is that points located at the same depth across lanes will undergo a uniform scaling factor during image projection. Besides, sharing the same road gives rise to shared properties among the lanes in 3D space, *e.g.*, identical elevation. The inner structure of this module is shown in Fig. 2 part (b).

1) Lane-level embedding encodes features of N lane instances from the image feature map $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$. Concretely, we utilize a lane-aware feature aggregator to gather features of distinct lanes, based on a set of instance activation maps (IAMs) $\mathbf{A} \in \mathbb{R}^{N\times(H\times W)}$ [4]. The IAMs are dynamically generated and can be formulated as:

$$\mathbf{A} = \sigma(\mathcal{F}([\mathbf{X}, \mathbf{S}])),$$

where $\mathcal{F}$ is implemented by several convolution layers, σ is a sigmoid function, $[\,,\,]$ denotes concatenation and $\mathbf{S}$ is a two-channel feature map which represents the 2D spatial localization of pixels [22]. With IAMs, the lane-level embedding $\mathbf{Q}_{lane} \in \mathbb{R}^{N\times C}$ can be obtained via:

$$\mathbf{Q}_{lane} = \mathbf{A} \cdot \mathbf{X}^{\mathsf{T}}.$$

During training, we add an auxiliary segmentation head [4] on top of $\mathbf{Q}_{lane}$ to predict 2D lane instance masks, which are supervised by projected 3D annotations. Following [4], we assign ground-truth labels to N instance masks using dice-based bipartite matching. We also use the same matched results to assign labels to our final lane predictions. Please refer to our supplementary for more details.

2) Point-level embedding expresses how points relate to each other in a lane. Instead of deriving it from the image features, we represent it as a set of learnable weights $\mathbf{Q}_{point} \in \mathbb{R}^{M\times C}$, where each $\boldsymbol{q}^i_{point} \in \mathbb{R}^{1\times C}$ corresponds to the embedding of a y^i in the predefined $\mathbf{Y}_{ref}$. These embeddings will be learned during training.

3) Lane-aware query embedding $\mathbf{Q} \in \mathbb{R}^{(N\times M)\times C}$ can be obtained using the following equation:

$$\mathbf{Q} = \mathbf{Q}_{lane} \oplus \mathbf{Q}_{point},$$

where $\oplus$ denotes the broadcasting summation. The broadcast operation enables the model to distinguish different instances and group points based on lane-level embedding. Note that, sharing point-level embedding can help model relative relation and common features among points across different lanes without introducing geometric priors, *e.g.*, point-distance constrain, and sophisticated supervision as done in previous methods [14].

3.3. Dynamic 3D Ground Positional Embedding

As discussed in Sec. 2.2, existing methods primarily either utilize a surrogate 3D space to estimate 3D lane positions [3, 8, 7, 5] or implicitly force the model to learn 3D positions [1]. Differently, we propose to use the prior that all lanes are located on the ground in the real world and construct a 3D plane to model the ground. Although there have been several attempts to leverage ground prior to facilitating 3D object detection, some make strong hypotheses like fixed ground [27], while others introduce extra prediction tasks, *e.g.*, dense depth estimation [46, 39, 35], horizon line detection, and contact points detection [50]. However, predicting these extra tasks poses a significant challenge due to their high degree of freedom (DOF), and inaccurate predictions can inevitably undermine performance due to accumulative errors. In this paper, we tackle this problem by restricting the plane to only two DOF. Specifically, we encode the plane as the *key*'s positional embedding in each deformable attention module. As follows, we will delve into how to generate the *3D ground positional embedding* and update the hypothetical plane to dynamically approach the real ground with only 3D lane annotations.

Positional Embedding Generation. We first construct a 3D plane $\mathbf{P} \in \mathbb{R}^{P\times 3}$ represented by 3D grids with P points $\mathbf{P} = \{(x_i, y_i, z_i)|i \in 1, ..., P\}$ and project all points into the 2D image using camera parameters T as:

$$d * [u, v, 1]^{\mathsf{T}} = T \cdot [x, y, z, 1]^{\mathsf{T}}. \quad (1)$$

We initialize the grid as a horizontal plane with z empirically set to -1.5. Based on Eq. (1), we scatter all projected points into a 2D canvas $\mathbf{M}_p \in \mathbb{R}^{3\times H\times W}$, which preserves 3D position for each projected point as:

$$\mathbf{M}_p[:, v, u] = (x, y, z), \quad (2)$$

where (u, v) and (x, y, z) denotes 2D and 3D coordinates defined in Eq. (1). For those pixels without projected points, we simply set them to 0s. Afterward, we obtain the 3D ground positional embedding $\mathbf{PE} \in \mathbb{R}^{C\times H\times W}$ via an MLP.

Dynamic Plane Update. In order to dynamically update the plane to approach the real ground, we predict a residual transformation matrix with two DOF (*i.e.*, $\Delta\theta_x$, Δh) in each decoder layer, using image features and projected canvas,

$$[\Delta\theta_x, \Delta h] = \mathrm{MLP}(\mathrm{AvgPool}(\mathcal{G}([\mathbf{X}, \mathbf{M}_p]))), \tag{3}$$

where $[\,,\,]$ represents concatenation, $\mathcal{G}$ is two convolution layers, $\mathbf{X}$ denotes 2D features from backbone and $\mathbf{M}_p$ is the projected canvas that encodes 3D positions from Eq. (2). Further, the transformation matrix can be formulated as:

$$D = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\Delta\theta_x & -\sin\Delta\theta_x & 0 \\ 0 & \sin\Delta\theta_x & \cos\Delta\theta_x & \Delta h \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4}$$

Then, we iteratively update the plane layer by layer via:

$$\widetilde{\mathbf{P}}_l^{\mathsf{T}} = D \cdot \widetilde{\mathbf{P}}_{l-1}^{\mathsf{T}}, \tag{5}$$

where $\widetilde{\mathbf{P}}$ is the homogeneous representation of $\mathbf{P}$ and l indexes the decoder layer. An algorithm is provided in our Appendix to summarize this procedure.

Supervision. To supervise the predicted transformation matrix D with two DOF, $\Delta\theta_z$, Δh. we project 3D lane annotations onto images using the camera parameters T via: $l_{u,v} = T \cdot l_{x,y,z}$, where $l_{x,y,z}$ is 3D lane annotations and $l_{u,v}$ is the corresponding location on the 2D image. Similar to Eq. (2), we scatter all projected lanes into $\mathbf{M}_l$. Thus, we use projected 3D lane annotations to sparsely supervise the projected plane. Given $\mathbb{P}$, which represents the set of all pixels projected from the plane, and $\mathbb{L}$, the set of all projected 3D lane annotations, our loss can be formulated as:

$$\mathcal{L}_{plane} = \sum_{u,v \in \mathbb{P} \cap \mathbb{L}} ||\mathbf{M}_p[:, u, v] - \mathbf{M}_l[:, u, v]||_2.$$

We use $\mathcal{L}_{plane}$ to supervise the update of the constructed 3D plane, allowing it to approach the real ground and obtain accurate 3D ground positional information. An illustration demonstrating the efficacy of $\mathcal{L}_{plane}$ is shown in Fig. 3.

3.4. Decoder

We build our decoder with L decoder layers following standard Transformer-based methods [2, 57]. In each layer, we use the query to predict 3D lane locations (x, y, z) as 3D reference points ref_{3D} and project each 3D point into the 2D image termed as ref_{2D} following Eq. (1). Then, we formulate the message exchange process using the lane-aware query embedding $\mathbf{Q} \in \mathbb{R}^{(N \times M) \times C}$ and the 3D ground positional embedding $\mathbf{PE} \in \mathbb{R}^{C \times (H \times W)}$ as follows:

$$\mathbf{Q}_l = \mathrm{DeformAttn}(\mathbf{Q}_{l-1}, \mathbf{X}^{\mathsf{T}} + \mathbf{PE}^{\mathsf{T}}, \mathbf{X}^{\mathsf{T}}, ref_{2D}),$$

where $\mathbf{X} \in \mathbb{R}^{C \times (H \times W)}$ represents the extracted image feature map, l is the layer index, and DeformAttn is a standard deformable attention module [57]. As shown in Fig. 2, we also estimate the residual transformation matrix of our constructed plane as illustrated in Sec. 3.3 and iteratively adjust its locations similar to reference points. With this iterative refinement mechanism, LATR can progressively update its knowledge about 3D ground and improve its localizability.

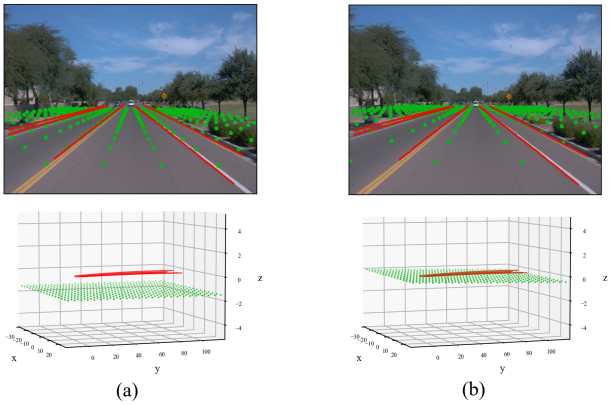

Figure 3. **An illustration of dynamic ground update.** Green points are sampled from our constructed 3D ground plane, and red lines represent ground truth lanes. (a) The left column demonstrates the initial 3D position of the constructed plane (green) and the ground truth (red). (b) The right column shows the updated position, where our constructed ground learned to find the real road position, *i.e.*, they are nearly in the same plane.

3.5. Prediction Head and Losses

Prediction Head. We apply a prediction head on top of the query to generate our final predictions. For the 3D positions estimation, we use an MLP which can be formulated as:

$$[\boldsymbol{\Delta x}, \boldsymbol{\Delta z}, \boldsymbol{v}] = \mathrm{MLP}_{reg}(\mathbf{Q}),$$

where $\boldsymbol{\Delta x}, \boldsymbol{\Delta z} \in \mathbb{R}^{N \times M \times 1}$ represents the offsets *w.r.t.* the corresponding reference points (Sec. 3.4) in last decoder layer. And $\boldsymbol{v} \in \mathbb{R}^{N \times M \times 1}$ denotes the visibility of each predicted lane point, which indicates whether the projected point is valid in the image. Together with the predefined longitudinal coordinate $\mathbf{Y}_{ref} \in \mathbb{R}^{M \times 1}$, we obtain N point sets as ref_{3D}. For the lane category, we adopt max-pooling along the point dimension followed by a per-instance MLP, formulated as:

$$\mathcal{C} = \mathrm{MLP}_{cls}(\mathrm{MaxPool}(\mathbf{Q})),$$

where $\mathcal{C} \in \mathbb{R}^{N \times K}$ denotes the class-logits and K is the number of possible classes. Lanes classified as "background" will be discarded in our final predictions. Here we apply the same bipartite matching results as our auxiliary segmentation in Sec. 3.2 for 3D lane label assignment. This strategy ensures consistent 2D segmentation and 3D lane detection supervision for each query.

Overall Loss. Given matched ground truth labels for the lane queries, we calculate the corresponding loss for each matched pair. Concretely, our total loss consists of three parts: the instance segmentation auxiliary loss $\mathcal{L}_{seg}$ (Sec. 3.2), the 3D ground-aware plane update loss $\mathcal{L}_{plane}$

Methods	F1 ↑	Category Accuracy ↑	X error (m) ↓		Z error (m) ↓	
			near	*far*	*near*	*far*
3DLaneNet [7]	44.1	-	0.479	0.572	0.367	0.443
GenLaneNet [8]	32.3	-	0.593	0.494	0.140	0.195
Cond-IPM	36.6	-	0.563	1.080	0.421	0.892
Persformer* [3]	50.5	89.5	0.319	0.325	0.112	0.141
CurveFormer [1]	50.5	-	0.340	0.772	0.207	0.651
Persformer-Res50†	53.0	89.2	0.321	0.303	0.085	0.118
LATR-Lite	61.5	91.9	0.225	0.249	0.073	0.106
LATR	**61.9** ↑11.4	**92.0** ↑2.5	**0.219** ↓0.100	**0.259** ↓0.066	**0.075** ↓0.037	**0.104** ↓0.037

Table 1. **Comparison with other 3D lane detection methods on the OpenLane validation dataset.** $*$ denotes the officially released results. † denotes the results reproduced by Persformer with ResNet50 and input images with a shape of 720×960 for a fair comparison. The ↓ in the head row indicates that lower metric values correspond to better performance, and vice versa. **Bold** numbers denote the best results and underline ones denote the previous best results. Performance gains are highlighted with blue arrows.

Methods	All	Up & Down	Curve	Extreme Weather	Night	Intersection	Merge & Split
3DLaneNet [7]	44.1	40.8	46.5	47.5	41.5	32.1	41.7
GenLaneNet [8]	32.3	25.4	33.5	28.1	18.7	21.4	31.0
Persformer* [3]	50.5	42.4	55.6	48.6	46.6	40.0	50.7
CurveFormer [1]	50.5	45.2	56.6	49.7	49.1	42.9	45.4
Persformer-Res50†	53.0	45.7	58.5	53.8	47.5	41.3	51.5
LATR-Lite (ours)	61.5	55.2	67.9	**57.6**	55.1	52.1	60.3
LATR (ours)	**61.9**	**55.2**	**68.2**	57.1	**55.4**	**52.3**	**61.5**
Improvement	↑11.4	↑10.0	↑11.6	↑7.4	↑6.3	↑9.4	↑10.8

Table 2. **Comparison with other 3D lane detection methods on OpenLane dataset under different scenarios.** $*$ denotes the latest official results. † denotes the results reproduced by Persformer [3] with ResNet50 [9] using the same input size as ours for a fair comparison. Black bold ones denote the best results. Blue arrows denote the performance gain. **LATR-Lite** is the lite version of our LATR, which comprises only two decoder layers, while it surpasses all previous SoTA methods and is on par with LATR. ***improvement*** is calculated by comparing LATR and the previous best results.

(Sec. 3.3), and the 3D lane prediction loss $\mathcal{L}_{lane}$. Formally, we have:

$$\mathcal{L}_{lane} = w_x\mathcal{L}_x + w_z\mathcal{L}_z + w_v\mathcal{L}_v + w_c\mathcal{L}_c,$$
$$\mathcal{L} = w_s\mathcal{L}_{seg} + w_p\mathcal{L}_{plane} + w_l\mathcal{L}_{lane},$$

where $w_{[*]}$ represent different loss weights: w_s=5.0, w_x=2.0, w_z=10.0, w_c=10.0, and the rest are set to 1.0. We use L1 loss for $\mathcal{L}_x, \mathcal{L}_z$ and use BCE loss for $\mathcal{L}_v$. For classification, we adopt focal loss [19] with γ=2.0 and α=0.25, same as [17, 25].

4. Experiments

We evaluate our method on three 3D lane benchmarks: OpenLane [3], Apollo [8] and ONCE-3DLanes [49].

4.1. Datasets and Metrics

OpenLane [3] is a comprehensive, large-scale benchmark for 3D lane detection, built on the Waymo dataset [40]. This dataset consists of 1000 segments, including 200K frames captured under various weather, terrain, and brightness conditions at a resolution of 1280×1920. OpenLane contains 880K lane annotations, which are spread across a total of 14 categories, providing a realistic and diverse set of challenges for 3D lane detection algorithms.

Apollo Synthetic [8] is generated using a game engine. It comprises over 10k images covering three distinct scenes: 1) *Balanced scenes*, 2) *Rarely observed scenes*, and 3) *Scenes with visual variations*. The dataset includes diversified terrain structures such as highways, urban, and residential areas, as well as various illumination conditions.

ONCE-3DLanes [49] is a real-world 3D lane dataset, built on the ONCE [32] dataset. It contains 211K images comprising different locations, lighting conditions, weather conditions and slop scenes. Notably, camera extrinsics are not accessible in ONCE-3DLanes.

Evaluation Metrics. We follow official evaluation metrics to investigate our model on the above two datasets. The evaluation is formulated as a matching problem based on minimum-cost flow, where the lane matching cost is obtained by taking the square root of the squared sum of pointwise distances over predefined ys. A lane prediction is con-

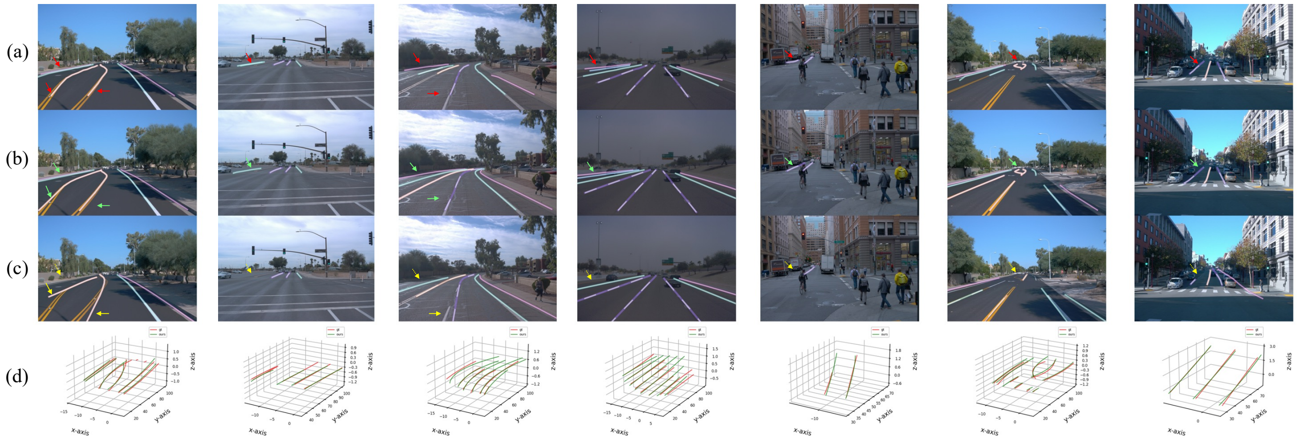

Figure 4. **Qualitative evaluation on OpenLane *val* set.** The rows (a), (b), (c) illustrate ground truth 3D lanes, prediction from LATR and Persformer [3] with 2D projection, respectively. Here, different colors indicate specific categories. Row (d) demonstrates the ground truth (red) and prediction of LATR (green) in 3D space. Best viewed in color (zoom in for details).

sidered matched if at least 75% of its points have a point-wise distance less than a predefined threshold of 1.5 meters [8, 3]. Errors are measured in the near range [0, 40]m and far range [40, 100]m along the heading direction. Moreover, the Average Precision (AP) metric is used to evaluate the performance on the Apollo Synthetic dataset [8].

4.2. Experimental Settings

Implementation Details. We use an input shape of 720 × 960 and adopt ResNet-50 [9] as our backbone to extract feature maps from three scales with spatial reduction ratios of $[\frac{1}{8}, \frac{1}{16}, \frac{1}{32}]$. After that, we exploit FPN [18] to generate multi-scale features, constructing a four-layer feature pyramid. Specifically, we apply dilation convolutions to upsample small feature maps into the largest scale, *i.e.*, the $\frac{1}{8}$ ratio, and aggregate all features as the input of our decoder. For the decoder, we employ deformable attention [57] with 4 attention heads, 8 sample points, and 256-D embeddings. Aligning with the common setting in object detection [52], we use a six-layer decoder in LATR as our default version and a two-layer design as our lite version, LATR-Lite.

Training. All our experiments are trained with the AdamW optimizer with a weight decay of 0.01. We set the learning rate to be 2×10^{-4} and use a cosine annealing scheduler. We use batch size 32 and trained models on A100 GPUs. We train the models for 24 epochs on OpenLane and ONCE-3DLanes, and 100 epochs on Apollo dataset.

4.3. Main Results

Results on OpenLane. We present the main results on OpenLane *val* set in Tab. 1. Several key trends can be observed: **1)** LATR surpasses all previous methods by substantial margins. It achieves a remarkable improvement of 11.4 in F1 score and 2.5 in category accuracy compared to the previous state-of-the-art method, Persformer [3]. **2)** LATR achieves significant reductions in both X and Z direction errors across both the near and far range, compared to the previous SOTA. Notably, the errors are reduced by 0.100m/0.066m and by 0.037m/0.037m in X and Z directions within the near/far range, respectively. **3)** LATR-Lite, albeit a lightweight version of LATR with only two decoder layers, attains comparable results to LATR. **4)** Although Persformer (Persformer-Res50) shows an increase in F1 from 50.5 to 53.0 by utilizing the same backbone and input shape as LATR, it still significantly lags behind LATR.

Different Scenarios in OpenLane. Apart from the main results on OpenLane, we also conducted comprehensive experiments across various scenarios within OpenLane *val* set. As shown in Tab. 2, our proposed LATR outperforms state-of-the-art methods by significant margins across six challenging scenarios. Specifically, we observed that our model performs more accurate lane detection under complex scenarios involving *curves* (+11.6 *w.r.t.* F1) and *merges/splits* (+10.8 *w.r.t.* F1), which benefits from our hybrid lane query embedding design. Additionally, our model improves significantly in the *up/down* scenario (+10.0 *w.r.t.* F1), suggesting that our dynamic 3D ground design enables the model to perceive the road better. These findings demonstrate the effectiveness of our proposed approach in handling diverse driving scenarios. In Fig. 4, we present a qualitative comparison of LATR and Persformer [3], where our method performs more accurate predictions in several challenging scenarios.

However, we also observed that relatively modest improvements (+7.4/+6.3) are achieved under challenging lighting conditions, *extreme weather* and *night*, compared to other scenarios. This may be attributed to the inherent constraints of the vision-centric method, which heavily relies on visual cues for perception.

Results on Apollo. Tab. 3 summarizes the results of our ex-

Scene	Methods	F1 ↑	AP ↑	X error (m) ↓		Z error (m) ↓	
				near	*far*	*near*	*far*
Balanced Scene	3DLaneNet [7]	86.4	89.3	0.068	0.477	0.015	**0.202**
	Gen-LaneNet [8]	88.1	90.1	0.061	0.496	0.012	0.214
	CLGo [21]	91.9	94.2	0.061	0.361	0.029	0.250
	PersFormer [3]	92.9	-	0.054	0.356	0.010	0.234
	GP [14]	91.9	93.8	0.049	0.387	0.008	0.213
	CurveFormer [1]	95.8	97.3	0.078	0.326	0.018	0.219
	LATR-Lite	96.5	97.8	0.035	0.283	0.012	0.209
	LATR	**96.8** ↑1.0	**97.9** ↑0.6	**0.022** ↓0.027	**0.253** ↓0.073	**0.007** ↓0.001	**0.202** ↓0.011
Rare Subset	3DLaneNet [7]	74.6	72.0	0.166	0.855	0.039	**0.521**
	Gen-LaneNet [8]	78.0	79.0	0.139	0.903	0.030	0.539
	CLGo [21]	86.1	88.3	0.147	0.735	0.071	0.609
	PersFormer [3]	87.5	-	0.107	0.782	0.024	0.602
	GP [14]	83.7	85.2	0.126	0.903	0.023	0.625
	CurveFormer [1]	95.6	97.1	0.182	0.737	0.039	0.561
	LATR-Lite	95.8	97.2	0.060	0.618	0.020	0.538
	LATR	**96.1** ↑0.5	**97.3** ↑0.2	**0.050** ↓0.057	**0.600** ↓0.135	**0.015** ↓0.008	0.532 ↑0.011
Visual Variations	3DLaneNet [7]	74.9	72.5	0.115	0.601	0.032	0.230
	Gen-LaneNet [8]	85.3	87.2	0.074	0.538	0.015	0.232
	CLGo [21]	87.3	89.2	0.084	0.464	0.045	0.312
	PersFormer [3]	89.6	-	0.074	0.430	0.015	0.266
	GP [14]	89.9	92.1	0.060	0.446	**0.011**	0.235
	CurveFormer [1]	90.8	93.0	0.125	0.410	0.028	0.254
	LATR-Lite	94.0	95.6	0.048	0.352	0.018	0.231
	LATR	**95.1** ↑4.3	**96.6** ↑3.6	**0.045** ↓0.015	**0.315** ↓0.095	0.016 ↑0.005	**0.228** ↓0.002

Table 3. **Comparison with other 3D lane detection methods on Apollo 3D Synthetic dataset with three different scenes.** LATR achieves the best performance over all metrics and across three scenes with notable margins. Moreover, we implement a lite version, dubbed LATR-Lite, which consists of two decoder layers and achieves comparable results as LATR. Blue arrows denote gains, while red ones denote deterioration.

periments on the Apollo dataset. We evaluate our method on three different scenes and study the F1 score, AP and errors, following the literature [8]. Our LATR demonstrates superiority over all scenes and metrics, despite the performance being close to saturation. Notably, our design significantly boosts the performance by 4.3 points in F1 and 3.6 points in AP under the visual variations scenario, which suggests the effectiveness of our dynamic ground design. Moreover, we observe that our model achieves comparable results with both two-layer and six-layer decoders.

Method	**F1(%)↑**	**P(%)↑**	**R(%)↑**	**CD Error(m)↓**
3D-LaneNet [7]	44.73	61.46	35.16	0.127
Gen-LaneNet [8]	45.59	63.95	35.42	0.121
SALAD [49]	64.07	75.90	55.42	0.098
PersFormer [3]	74.33	80.30	69.18	0.074
LATR	80.59	86.12	75.73	0.052
improvement	↑ 6.26	↑ 5.82	↑ 6.55	↓ 0.022

Table 4. Comparison with state-of-the-art methods on ONCE-3DLanes *val* set. “P” and “R” denote precision and recall, respectively. Blue numbers indicate improvements.

Results on ONCE-3DLanes. We present the experimental results on ONCE-3DLanes [49] dataset in Tab. 4, where we employ a similar camera setting as [3]. Despite inaccurate camera parameters, our method outperforms existing benchmarks notably on this dataset. Compared with PersFormer, LATR obtains a higher F1 score (**+6.26**) and reduces CD error by 0.022m, under a criterion of $\tau_{CD} = 0.3$ [49] and $prob = 0.5$. This outcome exhibits the efficacy of our LATR over different datasets.

Overall, these experiments on both realistic and synthetic datasets demonstrate the generalization and robustness of our proposed method, indicating its potential to be applied in real-world scenarios.

4.4. Model Analysis

We conduct a thorough analysis to validate the effectiveness of our design choices in LATR using the OpenLane-300 dataset, following the literature [3]. Further details concerning model complexity, and extra ablation studies (including decoder layers and input size variations) are provided in our Appendix.

Module Ablations. The first and third rows in Tab. 5 show the results of using learnable weights to replace the lane-level embedding. Using the learnable embedding, without prior knowledge of image features, performs much worse than its lane-aware counterpart (61.5 *vs*. 70.4 and 45.5 *vs*. 67.9 in F1 score). Furthermore, no matter how the lane-level embedding is obtained, noticeable performance drops are observed when the dynamic 3D ground PE is dropped from LATR. The gap becomes even more significant when learnable weights replace the lane-level embedding (61.5→45.5). Overall, when singly applied, each component in Tab. 5 significantly improves the performance over the baseline. Employing both designs further boost performance and achieved the best results.

Lane Embed	Ground Embed	F1 / *C.Acc.*	X error (m) *near*	X error (m) *far*	Z error (m) *near*	Z error (m) *far*
		45.5 / 78.6	0.644	0.491	0.109	0.147
✓		67.9 / 90.3	0.281	0.349	0.102	0.139
	✓	61.5 / 87.7	0.352	0.381	0.101	0.140
✓	✓	**70.4 / 92.9**	**0.241**	**0.321**	**0.097**	**0.132**

Table 5. **Ablation studies on designed modules.** "Lane Embed" and "Ground Embed" denote lane-level embedding in Lane-aware Query Generator and Dynamic 3D ground PE Generator, respectively. *C.Acc.* means category accuracy.

Point-level Query vs. Lane-level Query. In LATR , each query is enhanced with the point-level embedding, corresponding to a single point in the final lane prediction. Every M points that belong to the same lane-level embedding are grouped together to form a complete lane. In this part, we explore another choice, using lane-level embedding $\mathbf{Q}_{lane} \in \mathbb{R}^{N\times C}$ as the final query embeddings. Unlike the point-level query approach, this setup requires predicting M different points for each lane query. As shown in Tab. 6, using pure lane-level queries incurs a noticeable performance drop (*e.g.*, 70.4→66.5 in terms of F1) when compared to our proposed setup.

Model	F1 / *C.Acc.*	X error (m) *near*	X error (m) *far*	Z error (m) *near*	Z error (m) *far*
Lane only	66.5 / 91.3	0.278	0.337	0.100	0.138
Lane + Point	**70.4 / 92.9**	**0.241**	**0.321**	**0.097**	**0.132**

Table 6. Different design choices for lane-aware queries.

Effect of Dynamic 3D Ground Positional Embedding. To evaluate the efficacy of our dynamic 3D ground positional embedding, we compared it with several alternatives. Specifically, we explored the assignment of 3D positions to image pixels using a fixed frustum [25] and a fixed ground plane, alongside our proposed method that employs an iteratively updated ground plane. As shown in Tab. 7, the incorporation of 3D positional information into image pixels yielded performance improvements across all evaluated methods. As expected, we observed that generating 3D positions via a frustum exhibits inferior results compared to using a plane. This is reasonable, given that lanes exist on the ground. Consequently, the adoption of a frustum introduced a significant proportion of points in the air, where lanes are non-existent. Besides, Tab. 7 shows that using a dynamically updated plane is better than a fixed one, demonstrating the effectiveness of our design choice.

Methods	F1 / *C.Acc.*	X error (m) *near*	X error (m) *far*	Z error (m) *near*	Z error (m) *far*
-	67.9 / 90.3	0.281	0.349	0.102	0.139
Fixed Frustum	69.1 / 91.3	0.277	0.343	0.101	0.137
Fixed Plane	69.2 / 91.7	0.263	**0.321**	0.100	0.133
Dynamic Plane	**70.4 / 92.9**	**0.241**	**0.321**	**0.097**	**0.132**

Table 7. **Effect of dynamic 3D ground PE.** We compare the results of different 3D positional embedding designs.

5. Discussion

While our model significantly improves performance on three public datasets, even surpasses the multi-modal method [30], it does encounter certain failure cases. As a vision-centric method, LATR is susceptible to the loss of critical visual cues (*e.g.,* glare or blur, invisible lanes in darkness, or severe shadows), which can be observed in Tab. 2. Incorporating inherent rich 3D geometric information in LiDAR may offer support in such challenging situations and potentially provide a more robust solution for 3D lane detection. The exploration of a multi-modal method is an open and intriguing avenue for future research.

6. Conclusion

In this work, we propose LATR, a simple yet effective end-to-end framework for 3D lane detection that achieves the best performance. It skips the surrogate view transformation and directly performs 3D lane detection on front-view features. We propose an effective lane-aware query generator to offer query informative prior and design a hybrid embedding to enhance query perception capability by aggregating lane-level and point-level features. Moreover, to build the 2D-3D connection, we devise a hypothetical 3D ground to encode 3D space information into 2D features. Extensive experiments show that LATR achieves remarkable performance. We believe that our work can benefit the community and inspire further research.

7. Acknowledge

This work was supported in part by Shenzhen General Program No.JCYJ20220530143600001, by the Basic Research Project No.HZQB-KCZYZ-2021067 of Hetao Shenzhen HK S&T Cooperation Zone, by Shenzhen-Hong Kong Joint Funding No.SGDX20211123112401002, by Shenzhen Outstanding Talents Training Fund, by Guangdong Research Project No.2017ZT07X152 and No.2019CX01X104, by the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No.2022B1212010001), The Chinese University of Hong Kong, Shenzhen, by the NSFC 61931024&81922046, by Tencent Open Fund.

References

[1] Yifeng Bai, Zhirong Chen, Zhangjie Fu, Lang Peng, Pengpeng Liang, and Erkang Cheng. Curveformer: 3d lane detection by curve propagation with curve queries and attention. *arXiv preprint arXiv:2209.07989*, 2022. 1, 2, 4, 6, 8

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 213–229. Springer, 2020. 2, 3, 4, 5

[3] Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, and Junchi Yan. Persformer: 3d lane detection via perspective transformer and the openlane benchmark. In *European Conference on Computer Vision (ECCV)*, 2022. 1, 2, 3, 4, 6, 7, 8

[4] Tianheng Cheng, Xinggang Wang, Shaoyu Chen, Wenqiang Zhang, Qian Zhang, Chang Huang, Zhaoxiang Zhang, and Wenyu Liu. Sparse instance activation for real-time instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4433–4442, 2022. 4

[5] Netalee Efrat, Max Bluvstein, Shaul Oron, Dan Levi, Noa Garnett, and Bat El Shlomo. 3d-lanenet+: Anchor free lane detection using a semi-local representation. *arXiv preprint arXiv:2011.01535*, 2020. 1, 2, 4

[6] Zhengyang Feng, Shaohua Guo, Xin Tan, Ke Xu, Min Wang, and Lizhuang Ma. Rethinking efficient lane detection via curve modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17062–17070, 2022. 2

[7] Noa Garnett, Rafi Cohen, Tomer Pe'er, Roee Lahav, and Dan Levi. 3d-lanenet: end-to-end 3d multiple lane detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2921–2930, 2019. 1, 2, 3, 4, 6, 8

[8] Yuliang Guo, Guang Chen, Peitao Zhao, Weide Zhang, Jinghao Miao, Jingao Wang, and Tae Eun Choe. Gen-lanenet: A generalized and scalable approach for 3d lane detection. In *European Conference on Computer Vision*, pages 666–681. Springer, 2020. 1, 2, 3, 4, 6, 7, 8

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6, 7

[10] Kuan-Chih Huang, Tsung-Han Wu, Hung-Ting Su, and Winston H Hsu. Monodtr: Monocular 3d object detection with depth-aware transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4012–4021, 2022. 3

[11] Dongkwon Jin, Wonhui Park, Seong-Gyun Jeong, Heeyeon Kwon, and Chang-Su Kim. Eigenlanes: Data-driven lane descriptors for structurally diverse lanes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17163–17171, 2022. 2

[12] Yeongmin Ko, Younkwan Lee, Shoaib Azam, Farzeen Munir, Moongu Jeon, and Witold Pedrycz. Key points estimation and point instance segmentation approach for lane detection. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):8949–8958, 2021. 2

[13] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12697–12705, 2019. 1

[14] Chenguang Li, Jia Shi, Ya Wang, and Guangliang Cheng. Reconstruct from top view: A 3d lane detection approach based on geometry structure prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4370–4379, 2022. 1, 2, 3, 4, 8

[15] Xiang Li, Jun Li, Xiaolin Hu, and Jian Yang. Line-cnn: End-to-end traffic line detection with line proposal unit. *IEEE Transactions on Intelligent Transportation Systems*, 21(1):248–258, 2019. 2

[16] Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. *arXiv preprint arXiv:2206.10092*, 2022. 3

[17] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*, pages 1–18. Springer, 2022. 3, 4, 6

[18] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 7

[19] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 6

[20] Lizhe Liu, Xiaohao Chen, Siyu Zhu, and Ping Tan. Condlanenet: a top-to-down lane detection framework based on conditional convolution. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3773–3782, 2021. 2

[21] Ruijin Liu, Dapeng Chen, Tie Liu, Zhiliang Xiong, and Zejian Yuan. Learning to predict 3d lane shape and camera pose from a single image via geometry constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1765–1772, 2022. 8

[22] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *Advances in neural information processing systems*, 31, 2018. 4

[23] Ruijin Liu, Zejian Yuan, Tie Liu, and Zhiliang Xiong. End-to-end lane shape prediction with transformers. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 3694–3702, 2021. 1, 2

[24] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic

anchor boxes are better queries for detr. *arXiv preprint arXiv:2201.12329*, 2022. 3

[25] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVII*, pages 531–548. Springer, 2022. 3, 4, 6, 9

[26] Yingfei Liu, Junjie Yan, Fan Jia, Shuailin Li, Qi Gao, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petrv2: A unified framework for 3d perception from multi-camera images. *arXiv preprint arXiv:2206.01256*, 2022. 3

[27] Yuxuan Liu, Yuan Yixuan, and Ming Liu. Ground-aware monocular 3d object detection for autonomous driving. *IEEE Robotics and Automation Letters*, 6(2):919–926, 2021. 4

[28] Zhijian Liu, Haotian Tang, Alexander Amini, Xinyu Yang, Huizi Mao, Daniela Rus, and Song Han. Bevfusion: Multi-task multi-sensor fusion with unified bird's-eye view representation. *arXiv preprint arXiv:2205.13542*, 2022. 3

[29] Yan Lu, Xinzhu Ma, Lei Yang, Tianzhu Zhang, Yating Liu, Qi Chu, Junjie Yan, and Wanli Ouyang. Geometry uncertainty projection network for monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3111–3121, 2021. 3

[30] Yueru Luo, Xu Yan, Chaoda Zheng, Chao Zheng, Shuqi Mei, Tang Kun, Shuguang Cui, and Zhen Li. Mˆ 2-3dlanenet: Multi-modal 3d lane detection. *arXiv preprint arXiv:2209.05996*, 2022. 2, 9

[31] Hanspeter A Mallot, Heinrich H Bülthoff, JJ Little, and Stefan Bohrer. Inverse perspective mapping simplifies optical flow computation and obstacle detection. *Biological cybernetics*, 64(3):177–185, 1991. 1

[32] Jiageng Mao, Minzhe Niu, Chenhan Jiang, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Jie Yu, Chunjing Xu, et al. One million scenes for autonomous driving: Once dataset. 2021. 6

[33] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards end-to-end lane detection: an instance segmentation approach. In *2018 IEEE intelligent vehicles symposium (IV)*, pages 286–291. IEEE, 2018. 2

[34] X Pan, J Shi, P Luo, X Wang, and X Tang. Spatial as deep: spatial cnn for traffic scene understanding. 2017. *Pan X Shi J Luo P Spatial As Deep: Spatial CNN for Traffic Scene Understanding*, 10, 2017. 2

[35] Zequn Qin and Xi Li. Monoground: Detecting monocular 3d objects from the ground. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3793–3802, 2022. 3, 4

[36] Zequn Qin, Huanyu Wang, and Xi Li. Ultra fast structure-aware deep lane detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*, pages 276–291. Springer, 2020. 2

[37] Zequn Qin, Pengyi Zhang, and Xi Li. Ultra fast deep lane detection with hybrid anchor driven ordinal classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 2

[38] Zhan Qu, Huan Jin, Yang Zhou, Zhen Yang, and Wei Zhang. Focus on local: Detecting lane marker from bottom up via key point. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14122–14130, 2021. 2

[39] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8555–8564, 2021. 4

[40] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020. 6

[41] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris M Kitani. Rethinking transformer-based set prediction for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3611–3620, 2021. 3

[42] Lucas Tabelini, Rodrigo Berriel, Thiago M Paixao, Claudine Badue, Alberto F De Souza, and Thiago Oliveira-Santos. Keep your eyes on the lane: Real-time attention-guided lane detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 294–302, 2021. 2

[43] Lucas Tabelini, Rodrigo Berriel, Thiago M Paixao, Claudine Badue, Alberto F De Souza, and Thiago Oliveira-Santos. Polylanenet: Lane estimation via deep polynomial regression. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6150–6156. IEEE, 2021. 2

[44] Wouter Van Gansbeke, Bert De Brabandere, Davy Neven, Marc Proesmans, and Luc Van Gool. End-to-end lane detection through differentiable least-squares fitting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 2

[45] Jinsheng Wang, Yinchao Ma, Shaofei Huang, Tianrui Hui, Fei Wang, Chen Qian, and Tianzhu Zhang. A keypoint-based global association network for lane detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1392–1401, 2022. 2

[46] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8445–8453, 2019. 4

[47] Yingming Wang, Xiangyu Zhang, Tong Yang, and Jian Sun. Anchor detr: Query design for transformer-based object detection. *arXiv preprint arXiv:2109.07107*, 3(6), 2021. 3

[48] Kyle R Williams, Rachel Schlossman, Daniel Whitten, Joe Ingram, Srideep Musuvathy, James Pagan, Kyle A Williams, Sam Green, Anirudh Patel, Anirban Mazumdar, et al. Trajectory planning with deep reinforcement learning in high-level action spaces. *IEEE Transactions on Aerospace and Electronic Systems*, 2022. 1

[49] Fan Yan, Ming Nie, Xinyue Cai, Jianhua Han, Hang Xu, Zhen Yang, Chaoqiang Ye, Yanwei Fu, Michael Bi Mi, and

Li Zhang. Once-3dlanes: Building monocular 3d lane detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17143–17152, 2022. 1, 2, 6, 8

[50] Fan Yang, Xinhao Xu, Hui Chen, Yuchen Guo, Jungong Han, Kai Ni, and Guiguang Ding. Ground plane matters: Picking up ground plane prior in monocular 3d object detection. *arXiv preprint arXiv:2211.01556*, 2022. 4

[51] Seungwoo Yoo, Hee Seok Lee, Heesoo Myeong, Sungrack Yun, Hyoungwoo Park, Janghoon Cho, and Duck Hoon Kim. End-to-end lane marker detection via row-wise classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 1006–1007, 2020. 2

[52] Renrui Zhang, Han Qiu, Tai Wang, Ziyu Guo, Ziteng Cui, Yu Qiao, Hao Dong, Peng Gao, and Hongsheng Li. Monodetr: Depth-guided transformer for monocular 3d object detection. 3, 7

[53] Renrui Zhang, Han Qiu, Tai Wang, Xuanzhuo Xu, Ziyu Guo, Yu Qiao, Peng Gao, and Hongsheng Li. Monodetr: Depth-aware transformer for monocular 3d object detection. *arXiv preprint arXiv:2203.13310*, 2022. 3

[54] Tu Zheng, Hao Fang, Yi Zhang, Wenjian Tang, Zheng Yang, Haifeng Liu, and Deng Cai. Resa: Recurrent feature-shift aggregator for lane detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3547–3554, 2021. 2

[55] Tu Zheng, Yifei Huang, Yang Liu, Wenjian Tang, Zheng Yang, Deng Cai, and Xiaofei He. Clrnet: Cross layer refinement network for lane detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 898–907, 2022. 2

[56] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. 1

[57] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2, 3, 4, 5, 7