PersFormer: 3D Lane Detection via Perspective Transformer and the OpenLane Benchmark

Li Chen[1*†] CHENLI1@PJLAB.ORG.CN
Chonghao Sima[1,2*] SIMAC@PURDUE.EDU
Yang Li[1*] LIYANG@PJLAB.ORG.CN
Zehan Zheng[1] ZHENGZEHAN@PJLAB.ORG.CN
Jiajie Xu[3] JIAJIEX@ANDREW.CMU.EDU
Xiangwei Geng[4] GENGXIANGWEI@SENSEAUTO.COM
Hongyang Li[1,5†] LIHONGYANG@PJLAB.ORG.CN
Conghui He[1] HECONGHUI@PJLAB.ORG.CN
Jianping Shi[4] SHIJIANPING@SENSEAUTO.COM
Yu Qiao[1] QIAOYU@PJLAB.ORG.CN
Junchi Yan[5] YANJUNCHI@SJTU.EDU.CN

[1] *Shanghai AI Laboratory, Shanghai, China*
[2] *Purdue University, West Lafayette, IN, USA*
[3] *Carnegie Mellon University, Pittsburgh, PA, USA*
[4] *SenseTime Research, Beijing, China*
[5] *Shanghai Jiao Tong University, Shanghai, China*
[*] *equal contributions.* [†] *corresponding authors.*

Abstract

Methods for 3D lane detection have been recently proposed to address the issue of inaccurate lane layouts in many autonomous driving scenarios (uphill/downhill, bump, *etc.*). Previous work struggled in complex cases due to their simple designs of the spatial transformation between front view and bird's eye view (BEV) and the lack of a realistic dataset. Towards these issues, we present PersFormer: an end-to-end monocular 3D lane detector with a novel Transformer-based spatial feature transformation module. Our model generates BEV features by attending to related front-view local regions with camera parameters as a reference. PersFormer adopts a unified 2D/3D anchor design and an auxiliary task to detect 2D/3D lanes simultaneously, enhancing the feature consistency and sharing the benefits of multi-task learning. Moreover, we release one of the first large-scale real-world 3D lane datasets, which is called OpenLane, with high-quality annotation and scenario diversity. OpenLane contains 200,000 frames, over 880,000 instance-level lanes, 14 lane categories, along with scene tags and the closed-in-path object annotations to encourage the development of lane detection and more industrial-related autonomous driving methods. We show that PersFormer significantly outperforms competitive baselines in the 3D lane detection task on our new OpenLane dataset as well as Apollo 3D Lane Synthetic dataset, and is also on par with state-of-the-art algorithms in the 2D task on OpenLane. The project page is available at https://github.com/OpenPerceptionX/OpenLane.

Keywords: Lane detection, 3D vision, Bird's eye view, Transformer, Datasets and benchmarks, Autonomous driving

1. Introduction

Autonomous driving is one of the most successful applications for AI algorithms to deploy in recent years. Modern Advanced Driver Assistance Systems (ADAS) for either L2 or L4 routes provide functionalities such as Automated Lane Centering (ALC) and Lane Departure Warning (LDW), where the essential need for perception is a lane detector to generate robust and generalizable lane lines (Comma.ai, 2017). With the prosperity of deep learning, lane detection algorithms in the 2D image space has achieved impressive results (Tabelini et al., 2021; Liu et al., 2021a; Qu et al., 2021), where the task is formulated as a 2D segmentation problem given front view (perspective) image as input (Lee et al., 2017; Pan et al., 2018; Neven et al., 2018; Abualsaud et al., 2021). However, such a framework to perform lane detection in the perspective view, 2D space is not applicable for industry-level products where complicated scenarios dominate.

On one side, downstream modules as in planning and control *often* require the lane location to be in the form of the orthographic bird's eye view (BEV) instead of a front view representation. Representation in BEV is for better task alignment with interactive agents (vehicle, road marker, traffic light, *etc.*) in the environment and multi-modal compatibility with other sensors such as LiDAR and Radar. The conventional approaches to address such a demand are either to simply project perspective lanes to ones in the BEV space (Wang et al., 2014; Meyer et al., 2018), or more elegantly to cast perspective features to BEV by aid of camera in/extrinsic matrices (Garnett et al., 2019; Guo et al., 2020; Yu et al., 2020b). The latter solution is inspired by the spatial transformer network (STN) (Jaderberg et al., 2015) to generate a 1-1 correspondence from the image to BEV feature grids. By doing so, the quality of features in BEV depends solely on the quality of the *corresponding* feature in the front view. The predictions using these outcome features are not adorable as the blemish of scale variance in the front view, which inherits from the camera's pinhole model, remains.

On the other side, the height[1] of lane lines has to be considered when we project perspective lanes into BEV space. As illustrated in Fig. 1, the lanes would diverge/converge in case of uphill/downhill if the height is ignored, leading to improper action decisions as in the planning and control module. Previous literature (Wang et al., 2014; Neven et al., 2018; Su et al., 2021) inevitably hypothesize that lanes in the BEV space lie on a flat ground, *i.e.*, the height of lanes is zero. The planar assumption does not hold true in most autonomous driving scenarios, *e.g.*, uphill/downhill, bump, crush turn, *etc.* Since the height information is unavailable on public benchmarks or complicated to acquire accurate ground truth, 3D lane detection is ill-posed. There are some attempts to address this issue by creating 3D synthetic benchmarks (Garnett et al., 2019; Guo et al., 2020). Their performance still needs improvement in complex, realistic scenarios nonetheless (c.f. (b-c) in Fig. 1). Moreover, the domain adaption between simulation and real data is not well-studied (Garnett et al., 2020).

To address these bottlenecks aforementioned, we propose Perspective Transformer, shortened as **PersFormer**, which has a spatial feature transformation module to optimize fea-

1. We define the height of lane line z to be the relative height concerning the zero point in the ego vehicle coordinate system (x, y, z) in BEV 3D space. The coordinate of the perspective (front view) 2D space in the image plane is referred to as (u, v).

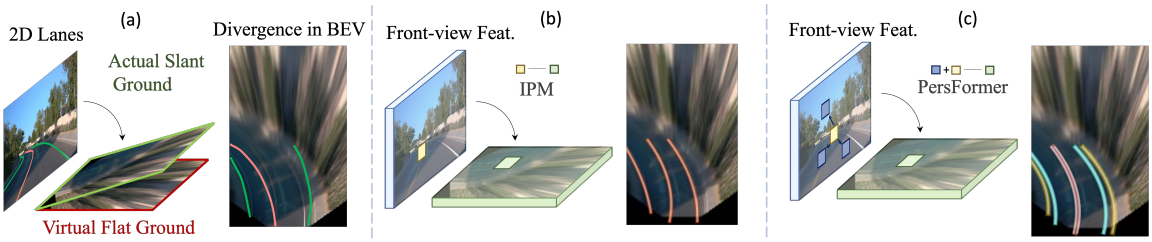

Figure 1: Motivation of performing lane detection from 2D in (a) to BEV in (b); and the superiority of our method in (c) versus (b). Lanes would diverge/converge in projected BEV on planar assumption, and a 3D solution with height to be considered can accurately predict the parallel topology in this case

tures and generate better BEV representations for the task. The proposed framework unifies 2D/3D lane detection tasks, and substantiates performance on the proposed large-scale realistic lane dataset.

First, we model the spatial feature transformation as a learning procedure that has an attention mechanism to capture the interaction both among local region in the front view feature and between two views (front view to BEV), consequently being able to generate a fine-grained BEV feature representation. Inspired by (Vaswani et al., 2017; Carion et al., 2020), we construct a Transformer-based module to realize this, while the deformable attention mechanism (Zhu et al., 2021) is adopted to remarkably reduce the computational memory requirement and dynamically adjust keys through the cross-attention module to capture prominent feature among the local region. Compared with direct 1-1 transformation via Inverse Perspective Mapping (IPM), the resultant features would be more representative and robust as it attends to the surrounding local context and aggregates relevant information. We further aim at unifying 2D and 3D lane detection tasks to benefit from the co-learning optimization. Second, we release the first real-world, large-scale 3D lane dataset and corresponding benchmark, OpenLane, to support research into the problem. OpenLane contains 200,000 annotated frames and over 880,000 lanes - each with one of 14 category labels (single white dash, double yellow solid, left/right curbside, *etc.*), which exceeds all of the existing lane datasets. It also has some distinguishing elements such as scenes, weather, and closed-in-path-object (CIPO) for other research topics in autonomous driving.

The main contributions of our work are three-fold: 1) Perspective Transformer, a novel Transformer-based architecture to realize spatial transformation of features; **2)** An architecture to simultaneously unify 2D and 3D lane detection, which is feasibly needed in the application. Experiments show that our PersFormer outperforms state-of-the-art 3D lane detection algorithms; **3)** The OpenLane dataset, the first large-scale realistic 3D lane dataset with high-quality labeling and vast diversity. The dataset, baselines, as well as the whole suite of codebase, will be released to facilitate the research in this area.

2. Related Work

2.1 Vision Transformers in Bird's-Eye-View (BEV)

Projecting features to BEV and performing downstream tasks in it has become more dominant and ensured better performance recently (Liu, 2021). Compared with conventional CNN structure, the cross attention scheme in Vision Transformers (Vaswani et al., 2017; Dosovitskiy et al., 2021; Carion et al., 2020; Liu et al., 2021c; Zhu et al., 2021) is naturally introduced to serve as a learnable transformation of features across different views in an elegant spirit (Liu, 2021). Instead of simply projecting features via IPM, the successful application of Transformers in view transformation has demonstrated great success in various domains, including 3D object detection (Yin et al., 2020; Wang et al., 2022; Guan et al., 2022), prediction (Gao et al., 2020; Gu et al., 2021; Ngiam et al., 2021), planning (Prakash et al., 2021; Chitta et al., 2021), *etc.*

Previous work (Garnett et al., 2019; Yang et al., 2021; Wang et al., 2022; Saha et al., 2021; Can et al., 2021) bring the BEV philosophy into pipeline, and yet they do not consider attention mechanism and/or 3D vision geometry (in this case, camera parameters). For instance, 3D-LaneNet (Garnett et al., 2019) is set up with camera in/extrinsic matrices; the IPM process generates a virtual BEV representation from front view features. As discussed in Section 1, such a direct transformation depends on the precision of camera parameters, and the scale of feature might be distorted due to the lack of accurate per-pixel depth information. DETR3D (Wang et al., 2022) also considers camera geometry and formulates a learnable 3D-to-2D query search with some attention scheme. However, there is no explicit BEV modelling for robust feature representation; the aggregated features might not be properly represented in 3D space. To address these shortcomings, our proposed PersFormer takes into account both the effect of camera parameters to generate BEV features and the convenience of cross-attention mechanism to model view transformation, achieving better feature representation in the end.

2.2 Lane Detection Benchmarks

A large-scale, diverse dataset with high-quality annotation is a pivot for lane detection. Along with the progress of lane detection approaches, numerous datasets have been proposed (Lee et al., 2017; Huang et al., 2019; Yu et al., 2020a; TuSimple, 2017; Behrendt and Soussan, 2019; Pan et al., 2018; Xu et al., 2020). However, they usually fit into one or the other lane detection scenario. For example, (Lee et al., 2017; Huang et al., 2019; Yu et al., 2020a) annotate lanes and lane markings in pixel-level so they are best suitable for semantic segmentation task. TuSimple (2017); Behrendt and Soussan (2019) collect data on highways with light traffic only, which is not challenging and has a large gap between the evaluation and real-world performance for up-to-date algorithms. Pan et al. (2018); Xu et al. (2020) consider more scenarios under different weather and traffic conditions; however, no-segment character limits their applicability for future applications, such as lane tracking or temporal lane detection. The recently released VIL-1000 (Zhang et al., 2021) is specifically designed for video instance lane detection, and yet it does not provide tracking ID annotation across the segments. Due to the difficulty of collecting 3D information for lanes, current 3D lane detection algorithms mainly focus on synthetic data (Guo et al., 2020). It

is small-scale and exists the domain gap between simulation and realistic scenarios. Tab. 1 depicts more details of the existing benchmarks and their comparison with our proposed OpenLane dataset. OpenLane is the first large-scale, realistic 3D lane dataset. It equips with a wide span of diversity in both data distribution and task applicability.

2.3 2D Lane Detection

Early lane detection approaches rely on traditional computer vision techniques, such as filtering (Aly, 2008; Li et al., 2016), clustering (Wang et al., 2014), *etc.* With the advent of deep learning, CNN-based methods significantly outperform hand-crafted algorithms. A typical way is to treat lane detection as a semantic segmentation problem (Lee et al., 2017; Pan et al., 2018; Neven et al., 2018; Hou et al., 2019; Abualsaud et al., 2021). Binary segmentation (Neven et al., 2018) needs post-clustering process for lane instance discrimination, while multi-class segmentation (Lee et al., 2017; Pan et al., 2018; Hou et al., 2019) usually limits the maximum detection results in one frame. Moreover, the pixel-wise classification takes large computation resources. To overcome this, several work propose lightweight yet effective grid based (Qin et al., 2020; Liu et al., 2021a; Jayasinghe et al., 2021; Qu et al., 2021) or anchor based (Chen et al., 2019; Li et al., 2019; Xu et al., 2020; Su et al., 2021; Tabelini et al., 2021) methods. The grid-based approach detects lanes in a row-wise way, whose resolution is much lower than the segmentation map. The model outputs the probability for each cell if it belongs to a lane and a vertical post-clustering process is still needed to generate the lane instances. Anchor-based approaches adopt the idea from classical object detection, focusing on optimizing the offsets from predefined line anchors. In this circumstance, how to define anchors is a critical problem. Chen *et al.* (Chen et al., 2019) adopts vertical anchors, which cause great difficulty for curving lane prediction. Some work (Li et al., 2019; Tabelini et al., 2021; Su et al., 2021) design anchors as a tilt slender shape, while the huge amount of different anchors to improve the detection accuracy would influence the computational efficiency. Nevertheless, considering their incredible performance on public datasets, we adopt the anchor-based formulation and carefully re-design anchors to achieve both high accuracy and efficiency.

2.4 3D Lane Detection

As discussed in Section 1, planar assumption does not always reserve in some cases, *i.e.*, uphill/downhill, bump. Several approaches (Nedevschi et al., 2004; Benmansour et al., 2008; Bai et al., 2018) utilize multi-modal or multi-view sensors, such as a stereo camera or LiDAR, to get the 3D ground topology. However, these sensors have shortages of high cost in hardware and computation resources, confining their practical applications. Recently, some monocular methods (Garnett et al., 2019; Guo et al., 2020; Jin et al., 2021; Liu et al., 2021b) take a single image and employ IPM to predict lanes in 3D space. 3D-LaneNet (Garnett et al., 2019) is the pioneering work in this domain with one simple end-to-end neural network; it adopts STN (Jaderberg et al., 2015) to accomplish the spatial projection of features. Gen-LaneNet (Guo et al., 2020) builds on top of 3D-LaneNet and designs a two-stage network for decoupling the segmentation encoder and 3D lane prediction head. However, its performance extensively depends on the accuracy of binary segmentation in the first stage, where it would fail in some scenarios, such as extreme weather or night scenario. These two

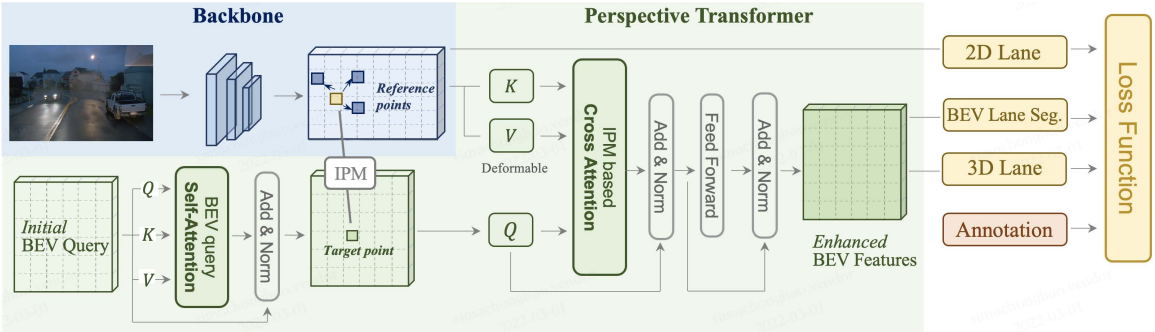

Figure 2: Our proposed PersFormer pipeline. The core is to learn a spatial feature transformation from front view to BEV space so that the generated BEV features at target point would be more representative by attending local context around reference point. PersFormer consists of the self-attention module to interact with its own BEV queries; the cross-attention module that takes the key-value pair from the IPM-based front view features to generate fine-grained BEV feature

approaches also suffer from improper feature transformation and unsatisfying performance in curving or crush turn cases. Confronted with the issues above, we bring in PersFormer to provide better feature representation and optimize anchor design to unify 2D and 3D lane detection simultaneously.

3. Methodology

In this section, we propose PersFormer, a unified 2D/3D lane detection framework with Transformer. We first describe the problem formulation, followed by an introduction to the overall structure in Section 3.2. In Section 3.3, we present Perspective Transformer, an explicit feature transformation module from front view to BEV space by the aid of camera parameters. In Section 3.4, we give details on the anchor design to unify 2D/3D tasks and in Section 3.5 we further elaborate on the auxiliary task and loss function to finalize our training strategy.

3.1 Problem Formulation

Given an input image $I_{org} \in \mathbb{R}^{H_{org} \times W_{org}}$, the goal of PersFormer is to predict a collection of 3D lanes $L_{3D} = \{l_1, l_2, \ldots, l_{N_{3D}}\}$ and 2D lanes $L_{2D} = \{l_1, l_2, \ldots, l_{N_{2D}}\}$, where N_{3D}, N_{2D} are the total number of 3D lanes in the pre-defined BEV range and 2D lanes in the original image space (front view) respectively. Mathematically, each 3D lane l_d is represented by an ordered set of 3D coordinates:

$$l_d = \left[(x_1, y_1, z_1), (x_2, y_2, z_2), \ldots, (x_{N_d}, y_{N_d}, z_{N_d})\right], \tag{1}$$

where d is the lane index, and N_d is the max number of sample points of this lane. The form of 2D lane is represented similarly with 2D coordinate (u, v) accordingly. Each lane has a categorical attribute $c_{\text{3D/2D}}$, indicating the type of this lane (*e.g.*, single-white dash line).

Also, for each point in a single 2D/3D lane, there exists an attribute property indicating whether the point is visible or not, denoted by $\mathbf{vis}_{\text{fv/bev}}$ as a vector for the lane.

3.2 Approach Overview

The overall structure, as illustrated in Fig. 2, consists of three parts: the backbone, the Perspective Transformer, and the lane detection post-processing. The backbone takes as input the resized image and generates multi-scale front view features. The popular ResNet variant (Tan and Le, 2019) is adopted as backbone. Note that these features might suffer from the defect of scale variance, occlusion, *etc.* - residing from the inherent feature extraction in the front view space. The Perspective Transformer takes the front view features as input and generates BEV features by the aid of camera intrinsic and extrinsic parameters. Instead of simply projecting the 1-1 feature correspondence from the front view to BEV, we introduce Transformer to attend local context and aggregate surrounding features to form a robust representation in BEV. By doing so, we learn the inverse perspective mapping from front view to BEV in an elegant manner with Transformer. Finally, the lane detection heads are responsible for predicting 2D and 3D coordinates as well as lane types. The 2D/3D detection heads are referred to as LaneATT (Tabelini et al., 2021) and 3D-LaneNet (Garnett et al., 2019), with some modification on the structure and anchor design.

3.3 Proposed Perspective Transformer

We present Perspective Transformer, a spatial transformation method that combines camera parameters and data-driven learning procedures. The general idea of Perspective Transformer is to use the coordinates transformation matrix from IPM as a reference to generate BEV feature representation, by attending related region (local context) in front view feature. On the assumption that the ground is flat and the camera parameters are given, a classical IPM approach calculates a set of coordinate mapping from front-view to BEV, where the BEV space is defined on the flat ground (see (Hartley and Zisserman, 2004), Section 8.1.1). Given a point p_{fv} with its coordinate (u, v) in the front-view feature $F_{\text{fv}} \in \mathbb{R}^{H_{\text{fv}} \times W_{\text{fv}} \times C}$, IPM maps the point p_{fv} to the corresponding point p_{bev} in BEV, where (x, y) is the coordinate in the BEV space $\mathbb{R}^{H_{\text{bev}} \times W_{\text{bev}} \times C}$. The transform is achieved with camera in/extrinsic and can be represented mathematically as:

$$\begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = \alpha_{f2b} \cdot R_\theta \cdot K^{-1} \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ -h \end{pmatrix}, \tag{2}$$

where α_{f2b} implies the scale factor between front-view and BEV, R_θ denotes the pitch rotation matrix from extrinsic, K is the intrinsic matrix, and h stands for camera height. Such a transformation in Eqn.(2) enframes a strong prior on the attention unit in Perspective Transformer to generate more representative features in BEV space.

The architecture of Perspective Transformer is inspired by popular approaches such as DETR (Carion et al., 2020), and consists of the self-attention module and cross-attention module (see Fig. 2). We differentiate from them in that the queries are not implicitly updated. However, instead, they are piloted by an explicit meaning - the physical location to detect objects or lanes in BEV. In the **self-attention** module, the output Q_{bev} descends

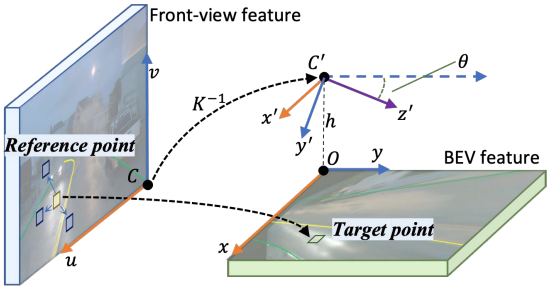

Figure 3: Generation of keys in the cross attention. Point (x, y) in BEV space casts the corresponding point (u, v) in front view through intermediate state $(x^{'}, y^{'})$; by learning offsets, the network learns target-reference points mapping from green rectangles to yellow and related blue rectangles as keys to Transformer

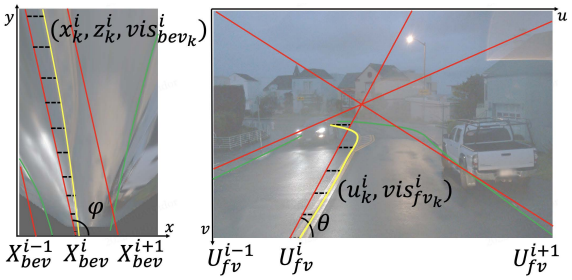

Figure 4: Unifying anchor design in 2D and 3D. We first put curated anchors (red) in the BEV space (left), then project them to the front view (right). Offset x_k^i and u_k^i (dashed line) are predicted to match ground truth (yellow and green) to anchors. The correspondence is thus built, and features are optimized together

from the triplet (key, value, query) input through their interaction. The formulation of such a self-attention can be described as:

$$Q_{\text{bev}} = \texttt{softmax}\left(\frac{QK^{\top}}{\sqrt{d_k}}\right)V, \tag{3}$$

where $K, Q, V \in \mathbb{R}^{(H_{\text{bev}} \times W_{\text{bev}} \times C)}$ are the same query that is pre-defined in BEV, $\sqrt{d_k}$ is the dimensional normalized factor.

In the **cross-attention** module, the input query $Q_{\text{bev}}^{'}$ is the outcome of several additional layers feeding the self-attention output Q_{bev} as input. Note that $Q_{\text{bev}}^{'}$ is an explicit feature representation as to which part in BEV should be paid more attention since the generation of queries is location-sensitive in BEV. This is quite different compared with queries that do not consider view transformation in most Vision Transformers (Wang et al., 2022; Guan et al., 2022; Zhu et al., 2021). Furthermore, the intuition behind employing Transformer to map features from front view to BEV is that such an attention mechanism would automatically attend which part of features contribute *most* towards the target point (query) in the destination view. The direct feature transformation would suffer from camera parameter noise or scale variance issues, as discussed and illustrated in Section 1. Note that the naive Transformer cannot be applied directly since the number of key-value pairs is huge and thus be confined by computational burden. Inspired by Deformable DETR (Zhu et al., 2021), we attend partial key-value pairs around the local region in a learnable manner to save cost and improve efficiency.

Fig. 3 depicts the feature transformation process and the generation of key-value pairs in cross-attention. Specifically, given a query point (x, y) in the target BEV map $Q_{\text{bev}}^{'}$, we project it to the corresponding point (u, v) in the front view via Eqn.(2). As does

similarly in (Zhu et al., 2021), we learn some offsets based on point (u, v) to generate a set of most related points around it. These learned points, together with (u, v) are defined as *reference points*. They contribute most to the query point (x, y), defined as *target point*, in BEV-space. The reference points serve as the surrounding context in the local region that contributes most to the feature representation from perspective view to BEV space. They are the desired keys we try to find, and their features are values for the cross attention module. Note that the initial locations of reference points from IPM are used as preliminary locations for the coordinate mapping; the location are adjusted gradually during the learning procedure, which is the core role of Deformable Attention.

Putting things together, the output of the cross-attention module can be formulated as:

$$F_{\text{bev}} = \texttt{DeformAttn}(Q'_{\text{bev}}, F_{\text{fv}}, p_{\text{fv2bev}}), \tag{4}$$

where $F_{\text{bev}} \in \mathbb{R}^{(H_{\text{bev}} \times W_{\text{bev}} \times C)}$ is the final desired features for the subsequent 3D head to get lane predictions, Q'_{bev} denotes the input queries, $F_{\text{fv}} \in \mathbb{R}^{(H_{\text{fv}} \times W_{\text{fv}} \times C)}$ indicates the front view features from backbone, and p_{fv2bev} is the IPM-inited coordinate mapping from front view to BEV space. Considering F_{fv} and p_{fv2bev} with the deformable unit, we get the explicit transformed BEV feature F_{bev}.

To sum up, Perspective Transformer extracts front-view features among the reference points to construct representative BEV features. As demonstrated in Section 5, such a feature transformation in an aggregation spirit via Transformer is proven to perform better than a direct IPM-based projection across views.

3.4 Simultaneous 2D and 3D Lane Detection

Although the main focus in this paper lies in 3D detection, we formulate the PersFormer framework to detect 2D and 3D lanes in one shot. On one side, 2D lane detection in the perspective view still draws interest in the community as part of the general high-level vision problems (Abualsaud et al., 2021; Tabelini et al., 2021; Liu et al., 2021a; Qu et al., 2021); on the other side, unifying 2D and 3D tasks are naturally feasible since the BEV features to predict 3D outputs descend from the counterpart in the 2D branch. An end-to-end unified framework would leverage features and benefit from the co-learning optimization process as proven in most multi-task literature (Liang et al., 2019; Vandenhende et al., 2021; Kumar et al., 2021).

Since our method is anchor-based detection, the core issue to achieve the unified framework is to integrate anchors in both 2D and 3D. Unfortunately, anchors in these two domains usually do not share similar distribution. For example, the popular 2D approach LaneATT (Tabelini et al., 2021) settles too many anchors, spanning different directions in the image; while the recent 3D work Gen-LaneNet (Guo et al., 2020) puts too few anchors, which are parallel and sparse in BEV. Based on these observations, we thereby design anchors such that the redesigned anchors could leverage the network to optimize shared features across *two* domains. We start with several groups of anchors (here, the group number is set to 7) sampled with different incline angles in the BEV space and then projected to the front view. Fig. 4 elaborates on the integration of 2D and 3D anchors. Below we describe how the lane line is modeled via anchors.

3.4.1 3D BEV ANCHOR DESIGN

To match ground truth lanes tightly, the anchors are placed approximately longitudinal along x-axis, with an incline angle φ. As denoted in Fig. 4(left), the initial line (equally spaced) with staring position along x-axis is denoted by X^i_{bev} for each anchor i. Similar to anchor regression in object detection, the network predicts the relative offset $\mathbf{x}^i$ w.r.t. the initial position X^i_{bev}; hence the resultant lane prediction along x-axis is $(\mathbf{x}^i + X^i_{\text{bev}})$. As indicated in Eqn.(1), each lane is represented as a number of N_d points. The prediction head generates three vectors related to lane shape as follows:

$$(\mathbf{x}^i, \mathbf{z}^i, \mathbf{vis}^i_{\text{bev}}) = \{(x^{(i,k)}, z^{(i,k)}, \text{vis}^{(i,k)}_{\text{bev}})\}_{k=1}^{N_d} \tag{5}$$

where $\mathbf{z}^i$ is the lane height in 3D sense, the binary $\text{vis}^{(i,k)}_{\text{bev}}$ denotes the visibility of each location k in lane i, which controls the endpoint or length of a lane. Note that the lane position along y-axis does not need to be predicted since each y value of the N_d samples in a lane is pre-defined - we predict the $x^{(i,k)}$ value at the corresponding (fixed) y location. To sum up, the description of a lane's location in the world coordinate system is denoted as $(\mathbf{x}^i + X^i_{\text{bev}}, \mathbf{y}, \mathbf{z}^i)$.

3.4.2 2D ANCHOR DESIGN

The anchor description and prediction are similar to those defined in 3D view, except that the (u, v) is in 2D space and there is no height (see Fig. 4(right)). We omit the detailed notations for brevity. It is worth mentioning that each 3D anchor X^i_{bev} with an incline angle φ corresponds to a specific 2D anchor U^i_{fv} with the incline angle θ; the connection is built via the projection in Eqn.(2). We achieve the goal of unifying 2D and 3D tasks simultaneously by setting the *same* set of anchors. Such a design would optimize features together and features being more aligned and representative across views.

3.5 Prediction Loss

3.5.1 INTERMEDIATE SUPERVISION

As do in many preceding work (Wei et al., 2016; Newell et al., 2016; Huang et al., 2019), adding more intermediate supervision into the network training would boost the performance of network. Since lane detection belongs to image segmentation and requires general large resolution, we concatenate a U-Net structure (Ronneberger et al., 2015) head on top of the generated BEV features. Such an auxiliary task is to predict lanes in BEV, but instead in a conventional 2D segmentation manner, aiming for better feature representation for the main task. The ground truth S_{gt} is a binary segmentation map projected from 3D lane ground truth to the BEV space. The prediction output is denoted by S_{pred} and owns the same size as S_{gt}.

3.5.2 LOSS FUNCTION

Equipped with the anchor representation and segmentation head aforementioned, we summarize the overall loss. Given an image input and its ground truth labels, it finally computes a sum of all anchors' loss; the loss is a combination of the 2D lane detection, 3D lane de-

Table 1: Comparison of OpenLane with existing benchmarks. "Avg. Length" denotes the average time duration of segments. "Inst. Anno." indicates whether lanes are annotated instance-wise (c.f. semantic-wise). "Track. Anno." implies if a lane has a unique tracking ID. Numbers in '#Frames' are the number of annotated frames over total frames respectively. Details of "Scenario" can be found in the Appendix. Compared datasets are: Caltech Lanes (Aly, 2008), TuSimple (TuSimple, 2017), 3D Synthetic (Guo et al., 2020), VIL-100 (Zhang et al., 2021), VPG (Lee et al., 2017), LLAMAS (Behrendt and Soussan, 2019), ApolloScape (Huang et al., 2019), BDD100K (Yu et al., 2020a), CULane (Pan et al., 2018), CurveLanes (Xu et al., 2020).

Dataset	#Segments	#Frames	Avg. Length	Inst. Anno.	Track. Anno.	Max #Lanes	Line Category	Scenario
Caltech Lanes	4	1224/1224	-	✓	✗	4	-	Easy
TuSimple	6.4K	6.4K/128K	1s	✓	✗	5	-	Easy
3D Synthetic	-	10K/10K	-	✓	-	6	-	Easy
VIL-100	100	10K/10K	10s	✓	✗	6	10	Medium
VPG	-	20K/20K	-	✗	-	-	7	Medium
LLAMAS	14	79K/100K	-	✓	✗	4	-	Easy
ApolloScape	235	115K/115K	16s	✗	✗	-	13	Medium
BDD100K	100K	100K/120M	40s	✗	✗	-	11	Medium
CULane	-	133K/133K	-	✓	-	4	-	Medium
CurveLanes	-	150K/150K	-	✓	-	9	-	Medium
OpenLane	1K	**200K/200K**	20s	✓	✓	**24**	**14**	**Hard**

tection and intermediate segmentation with learnable weights (α, β, γ) accordingly:

$$\mathcal{L} = \sum_i \alpha \mathcal{L}_{\text{2D}}(c^i_{\text{2D}}, \mathbf{u}^i, \mathbf{vis}^i_{\text{fv}}) + \beta \mathcal{L}_{\text{ 3D}}\ (c^i_{\text{3D}}, \mathbf{x}^i, \mathbf{z}^i, \mathbf{vis}^i_{\text{bev}}) + \gamma \mathcal{L}_{\text{seg}}(S_{\text{pred}}), \tag{6}$$

where $c^i_{(\cdot)}$ is the predicted lane category in 2D and 3D domain respectively. The loss input above shows the prediction part only; we omit the ground truth notation for brevity. The loss of lane category classification for the 2D/3D task is the cross-entropy; the loss of lane shape regression is the l_1 norm; the loss of lane visibility prediction is the binary cross-entropy loss. The loss of the auxiliary task is a binary cross-entropy loss between two segmentation maps.

4. OpenLane: A Large-scale Realistic 3D Lane Benchmark

4.1 Highlights over Previous Benchmarks

OpenLane is the *first* real world 3D lane dataset and the *largest* scale to date compared with existing benchmarks. We construct OpenLane on top of the influential Waymo Open dataset (Sun et al., 2020), following the same data format and evaluation pipeline - leveraging existent practice in the community so that users would not handle additional rules for a new benchmark. Tab. 1 compares OpenLane with existing counterparts in various aspects. In short, OpenLane owns 200K frames and over 880K carefully annotated lanes,

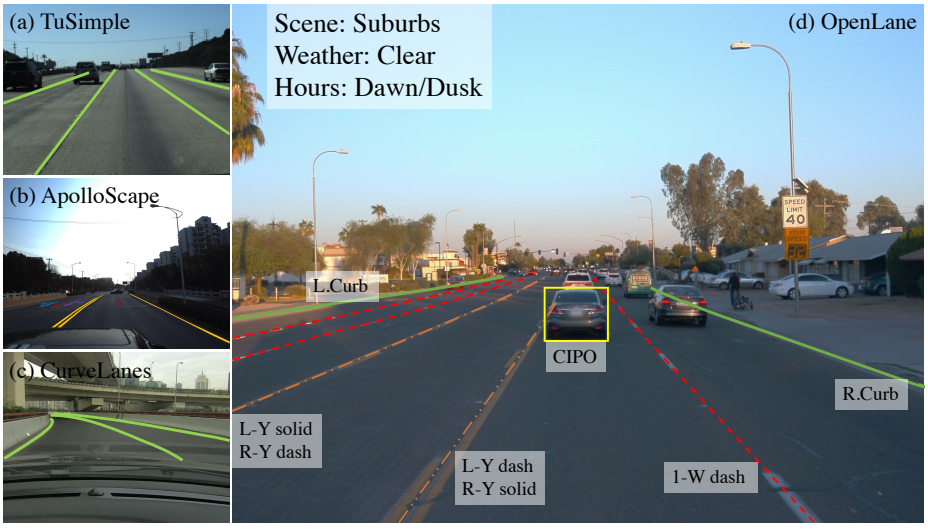

Figure 5: Annotation samples of OpenLane compared with other lane datasets. OpenLane is challenging with more lane categories per frame in average and has rich labels including scene, weather, hours, CIPO

33% and 35% more compared with existing largest lane dataset CurveLanes (Xu et al., 2020) respectively, with rich annotations.

We annotate all the lanes in each frame, including those in the *opposite* direction if no curbside exists in the middle. Due to the complicated lane topology, *e.g.*, intersection/roundabout, one frame could contain as many as **24** lanes in OpenLane. Statistically, about 25% frames of OpenLane have more than 6 lanes, which exceeds the maximum number in most lane datasets. **14** lane categories are annotated alongside to cover a wide range of lane types in most scenarios, including road edges. Double yellow solid lanes, single white solid and dash lanes take up almost 90% of total lanes. This is imbalanced, and yet it falls into a long-tail distribution problem, which is common in realistic scenarios. In addition to the lane detection task, we also annotate: (a) scene tags, such as weather and locations; (b) the closest-in-path object (CIPO), which is defined as the most concerned target w.r.t. ego vehicle; such a tag is quite pragmatic for subsequent modules as in planning/control, besides a whole set of objects from perception. An annotation example is provided in Fig. 5(d), along with some typical samples in existing 2D lane datasets in Fig. 5(a-c). The detailed statistics, annotation criterion and visualization can be found in the Supplementary.

4.2 Generation of High-quality Annotation

Building a real-world 3D lane dataset has challenges mainly in an accurate localization system and occlusions. We compare several popular sensor datasets (Chang et al., 2019; Sun et al., 2020; Caesar et al., 2020) by projecting 3D object annotations to image planes and constructing 3D scene maps using both learning-based (Teed and Deng, 2021) or SLAM algorithms (Shan et al., 2020, 2021). The reconstruction precision and scalability of Waymo

Open Dataset (Sun et al., 2020) outperforms other candidates, leading to employing it as our basis.

Primarily, we generate the necessary high-quality 2D lane labels. They contain the final annotations of tracking ID, category, and 2D points ground truth. Then for each frame, the point clouds are first filtered with the original 3D object bounding boxes and then projected back into the corresponding image. We further keep those points related to 2D lanes only with a certain threshold. However, the output directly after a static threshold filtering could lead to an unsatisfying ground truth due to the perspective scaling issue. To solve this and keep the slender shape of lanes, we use the filtered point clouds to interpolate the 3D position for each point in 2D annotations. Afterward, with the help of the localization and mapping system, 3D lane points in frames within a segment could be spliced into long, high-density lanes. This process could bring some unreasonable parts into the current frame; thus, points in one lane whose 2D projections are higher than the ending position of its 2D annotation are labeled as invisible. A smoothing step is ultimately deployed to filtrate any outliers and generate the 3D labeling results. We omit some technical details, such as how to deal with a large U-turn during smoothing, and we refer the audience to the Supplementary for more details.

5. Experiments

We examine PersFormer on two 3D lane benchmarks, the newly proposed real-world OpenLane dataset, and the synthetic Apollo dataset. We compare with previous SOTAs for both 3D and 2D lane detection tasks, namely 3D-LaneNet (Garnett et al., 2019), Gen-LaneNet (Guo et al., 2020), LaneATT (Tabelini et al., 2021), *etc.*

5.1 Evaluation Metrics and Implementation Details.

For both 3D lane datasets, we follow the evaluation metrics designed by Gen-LaneNet (Guo et al., 2020), with additional category accuracy on OpenLane dataset. For the 2D task, the classical metric in CULane (Pan et al., 2018) is adopted. We put correlated details in supplementary materials.

5.2 Results on OpenLane

We provide 3D and 2D evaluation results on the proposed OpenLane dataset. In order to evaluate the models thoroughly, we report F-Score on the entire validation set and different scenario sets. The scenario sets are selected from the entire validation set based on the scene tags of each frame, consisting of Up&Down case, Curve case, Extreme Weather case, Night case, Intersection case, and Merge&Split case. In Tab. 2, PersFormer gets the highest F-Score on the entire validation set and every scenario set, surpassing previous SOTA methods in varying degrees. In Tab. 3, PersFormer outperforms LaneATT (Tabelini et al., 2021), which is our baseline 2D method, by **11%** on the overall validation set. Comparison with more 2D methods is shown in supplementary materials. Detailed comparison with previous 3D SOTAs is presented in Tab. 4. PersFormer outperforms the previous best method in F-Score by **7.6%**, realizes satisfying accuracy on the classification of lane type, and presents the first baseline result. Note that PersFormer is not satisfying on the metrics of the near

Table 2: Comparison with other open-sourced 3D methods on OpenLane. PersFormer achieves the best F-Score on the entire validation set and every scenario set

Method	All	Up & Down	Curve	Extreme Weather	Night	Intersection	Merge & Split
3D-LaneNet	40.2	37.7	43.2	43.0	39.3	29.3	36.5
Gen-LaneNet	29.7	24.2	31.1	26.4	17.5	19.7	27.4
PersFormer (ours)	**47.8**	**42.4**	**52.8**	**48.7**	**46.0**	**37.9**	**44.6**

Table 3: Comparison with baseline 2D method on OpenLane. PersFormer achieves the best F-Score on the entire validation set and every scenario set

Method	All	Up & Down	Curve	Extreme Weather	Night	Intersection	Merge & Split
LaneATT-S	28.3	25.3	25.8	32.0	27.6	14.0	24.3
LaneATT-M	31.0	28.3	27.4	34.7	30.2	17.0	26.5
PersFormer (ours)	**42.0**	**40.7**	**46.3**	**43.7**	**36.1**	**28.9**	**41.2**

Table 4: Comprehensive 3D Lane evaluation under different metrics. On the strength of unified anchor design, PersFormer outperforms previous 3D methods on the metrics of far error while retains comparable results on near error (m)

Method	F-Score	Category Accuracy	X error near	X error far	Z error near	Z error far
3D-LaneNet	40.2	-	**0.278**	0.823	**0.159**	0.714
Gen-LaneNet	29.7	-	0.309	0.877	0.160	0.750
PersFormer (ours)	**47.8**	**92.3**	0.322	**0.778**	0.213	**0.681**

error on x-axis and z-axis; this is probably because the unified anchor design is more suitable in fitting the main body of a lane rather than the starting point. Qualitative results are shown in Fig. 6, indicating that PersFormer is good at catching dense and unapparent lanes in usual autonomous driving scenes (more can be found in supplementary materials). Overall, our proposed PersFormer reaches the best performance on 3D lane detection and gains remarkable improvement in 2D on OpenLane.

5.3 Results on Apollo 3D Synthetic

We evaluate PersFormer on Apollo 3D Lane Synthetic dataset (Guo et al., 2020). In Tab. 5, while limited by the scale of the dataset (10K frames), our PersFormer still achieves the best F-Score on every scene set. In terms of X/Z error, our model gets comparable results compared to previous methods.

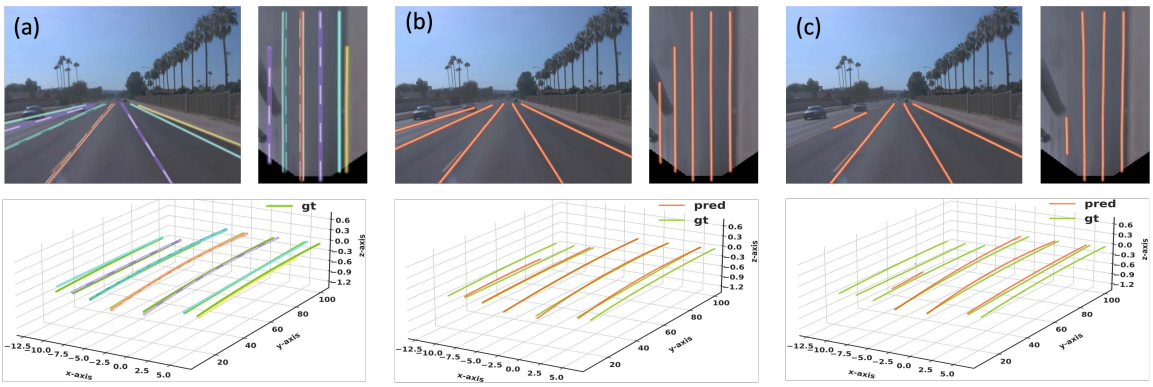

Figure 6: Qualitative results of PersFormer(a), 3D-LaneNet(b) (Garnett et al., 2019), and Gen-LaneNet(c) (Guo et al., 2020). Under a straight road scenario, PersFormer can provide lane-type information and even detect subtle curbside while other methods are missing it

Table 5: Comparison with previous 3D methods on Apollo 3D Lane Synthetic. PersFormer achieves best F-Score on every scene set with comparable X/Z error (m). Compared methods are: 3D-LaneNet (Garnett et al., 2019), Gen-LaneNet (Guo et al., 2020), 3D-LaneNet(l/att) (Jin et al., 2021), Gen-LaneNet(l/att) (Jin et al., 2021), CLGo (Liu et al., 2021b).

Scene	Method	F-Score	X error near	X error far	Z error near	Z error far
Balanced Scenes	3D-LaneNet	86.4	0.068	0.477	0.015	**0.202**
	Gen-LaneNet	88.1	0.061	0.496	0.012	0.214
	3D-LaneNet(l/att)	91.0	0.082	0.439	0.011	0.242
	Gen-LaneNet(l/att)	90.3	0.080	0.473	0.011	0.247
	CLGo	91.9	0.061	0.361	0.029	0.250
	PersFormer (ours)	**92.9**	**0.054**	**0.356**	**0.010**	0.234
Rarely Observed	3D-LaneNet	72.0	0.166	0.855	0.039	**0.521**
	Gen-LaneNet	78.0	0.139	0.903	0.030	0.539
	3D-LaneNet(l/att)	84.1	0.289	0.925	0.025	0.625
	Gen-LaneNet(l/att)	81.7	0.283	0.915	0.028	0.653
	CLGo	86.1	0.147	**0.735**	0.071	0.609
	PersFormer (ours)	**87.5**	**0.107**	0.782	**0.024**	0.602
Vivual Variants	3D-LaneNet	72.5	0.115	0.601	0.032	**0.230**
	Gen-LaneNet	85.3	0.074	0.538	0.015	0.232
	3D-LaneNet(l/att)	85.4	0.118	0.559	0.018	0.290
	Gen-LaneNet(l/att)	86.8	0.104	0.544	0.016	0.294
	CLGo	87.3	0.084	0.464	0.045	0.312
	PersFormer (ours)	**89.6**	**0.074**	**0.430**	**0.015**	0.266

Table 6: Ablative Study. Exp.1 is the baseline 3D method, growing with anchor design and multi-task learning (Exp.2-5). The performance culminates with our spatial feature transformation module and explicit BEV supervision (Exp.6,7)

Exp.	New Anchor	3D	2D	Perspective Transformer	Binary Seg	3D F-Score	2D F-Score
1		✓				41.77	-
2		✓	✓			43.49	32.33
3	✓		✓			-	34.90
4	✓	✓				42.75	-
5	✓	✓	✓			44.29	34.98
6	✓	✓	✓	✓		46.62	37.00
7	✓	✓	✓	✓	✓	47.79	42.00

5.4 Ablation Study

We present ablation studies on the anchor design, multi-task strategy, transformer-based view transformation, and auxiliary segmentation task. We mainly report the improvement on 3D lane detection and provide related results on 2D task.

5.4.1 Anchor design and multi-task

Starting with a pure 3D lane detection framework (similar to 3D-LaneNet (Garnett et al., 2019)), PersFormer gains **1.7%** by adopting multi-task scheme (Exp.2) and **0.98%** with new anchor design (Exp.4) respectively. Note that the new anchor design also helps 2D task independently (Exp.3). By jointly using the new anchor and multi-task trick, PersFormer acquires an improvement of **2.5%** in 3D task and **2.6%** in 2D task (Exp.5).

5.4.2 Spatial feature transformation

The spatial feature transformation plays a vital role in 3D lane detection. By using Perspective Transformer with the new anchor design, the improvement increases to **4.9%** (Exp.6), almost doubling the previous improvement. Adding auxiliary binary segmentation task further brings an improvement to **6.02%** (Exp.7), which is our complete model. These ablations support our assumption that PersFormer indeed generates a fine-grained BEV feature, and the spatial feature transformation does illustrate its importance in 3D lane detection task. Surprisingly, a better BEV feature helps 2D task a lot as well, improving **9.7%** (Exp.7). We leave the deep root cause analysis for future work, which may further prompt new paradigm.

6. Conclusions and Outlook

In this paper, we have proposed Persformer, a novel Transformer-based 2D/3D lane detector, along with OpenLane, a large-scale realistic 3D lane dataset. We demonstrate experimentally that a fine-grained BEV feature with explicit prior and supervision can significantly improve the performance of lane detection. As OpenLane is built upon Waymo Open Dataset (Sun et al., 2020), a road-object joint detection framework is possible in the future. Moreover, BEV is the necessity in the future of autonomous driving, and how to design a

better BEV representation remains to be explored. The proposed PersFormer may also be adapted to new tasks.

Acknowledgments

We would like to acknowledge the great support from SenseBee labelling team at SenseTime Research, and the fruitful discussions and comments for this project from Zhiqi Li, Yuenan Hou, Yu Liu, Jing Shao, Jifeng Dai.

Changelog

v1.0 (Mar. 2022): initial manuscript release; OpenLane benchmark version 1.0

References

Hala Abualsaud, Sean Liu, David B Lu, Kenny Situ, Akshay Rangesh, and Mohan M Trivedi. Laneaf: Robust multi-lane detection with affinity fields. *IEEE Robotics and Automation Letters*, 6(4):7477–7484, 2021. 2, 5, 9

Mohamed Aly. Real time detection of lane markers in urban streets. In *2008 IEEE Intelligent Vehicles Symposium, IV*, pages 7–12. IEEE, 2008. 5, 11

Min Bai, Gellert Mattyus, Namdar Homayounfar, Shenlong Wang, Shrinidhi Kowshika Lakshmikanth, and Raquel Urtasun. Deep multi-sensor lane detection. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pages 3102–3109. IEEE, 2018. 5

Karsten Behrendt and Ryan Soussan. Unsupervised labeled lane markers using maps. In *Proceedings of the IEEE International Conference on Computer Vision, ICCV*, 2019. 4, 11

Nabil Benmansour, Raphaël Labayrade, Didier Aubert, and Sébastien Glaser. Stereovision-based 3d lane detection system: a model driven approach. In *2008 11th International IEEE Conference on Intelligent Transportation Systems*, pages 182–188. IEEE, 2008. 5

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, CVPR*, pages 11621–11631, 2020. 12

Yigit Baran Can, Alexander Liniger, Danda Pani Paudel, and Luc Van Gool. Structured bird's-eye-view traffic scene understanding from onboard images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV*, pages 15661–15670, 2021. 4

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision, ECCV*, pages 213–229. Springer, 2020. 3, 4, 7, 32

Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 8748–8757, 2019. 12

Zhenpeng Chen, Qianfei Liu, and Chenfan Lian. Pointlanenet: Efficient end-to-end cnns for accurate real-time lane detection. In *2019 IEEE Intelligent Vehicles Symposium, IV*, pages 2563–2568. IEEE, 2019. 5

Kashyap Chitta, Aditya Prakash, and Andreas Geiger. Neat: Neural attention fields for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV*, pages 15793–15803, 2021. 4

Comma.ai. Openpilot. https://github.com/commaai/openpilot, 2017. 2

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations, ICLR*, 2021. 4

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 11525–11533, 2020. 4

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, ICCV*, 2019. 2, 4, 5, 7, 13, 15, 16, 24, 31, 35, 36, 37

Noa Garnett, Roy Uziel, Netalee Efrat, and Dan Levi. Synthetic-to-real domain adaptation for lane detection. In *Proceedings of the Asian Conference on Computer Vision, ACCV*, 2020. 2

Junru Gu, Chen Sun, and Hang Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV*, pages 15303–15312, 2021. 4

Tianrui Guan, Jun Wang, Shiyi Lan, Rohan Chandra, Zuxuan Wu, Larry Davis, and Dinesh Manocha. M3detr: Multi-representation, multi-scale, mutual-relation 3d object detection with transformers. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, WACV*, pages 772–782, 2022. 4, 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, ECCV*, pages 666–681. Springer, 2020. 2, 4, 5, 9, 11, 13, 14, 15, 24, 30, 31, 35, 36, 37

R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004. 7

Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning lightweight lane detection cnns by self attention distillation. In *Proceedings of the IEEE/CVF international conference on computer vision, ICCV*, pages 1013–1021, 2019. 5

Xinyu Huang, Peng Wang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. The apolloscape open dataset for autonomous driving and its application. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2702–2719, 2019. 4, 10, 11

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems, NeurIPS*, 28, 2015. 2, 5

Oshada Jayasinghe, Damith Anhettigama, Sahan Hemachandra, Shenali Kariyawasam, Ranga Rodrigo, and Peshala Jayasekara. Swiftlane: Towards fast and efficient lane detection. In *2021 20th IEEE International Conference on Machine Learning and Applications, ICMLA*, pages 859–864. IEEE, 2021. 5

Yujie Jin, Xiangxuan Ren, Fengxiang Chen, and Weidong Zhang. Robust monocular 3d lane detection with dual attention. In *2021 IEEE International Conference on Image Processing, ICIP*, pages 3348–3352, 2021. 5, 15

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980. 32

Varun Ravi Kumar, Senthil Yogamani, Hazem Rashed, Ganesh Sitsu, Christian Witt, Isabelle Leang, Stefan Milz, and Patrick Mäder. Omnidet: Surround view cameras based multi-task visual perception network for autonomous driving. *IEEE Robotics and Automation Letters*, 6(2):2830–2837, 2021. 9

Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Tae-Hee Lee, Hyun Seok Hong, Seung-Hoon Han, and In So Kweon. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *Proceedings of the IEEE international conference on computer vision, ICCV*, pages 1947–1955, 2017. 2, 4, 5, 11

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. 5

Zuo-Quan Li, Hui-Min Ma, and Zheng-Yu Liu. Road lane detection with gabor filters. In *2016 International Conference on Information System and Artificial Intelligence, ISAI*, pages 436–440. IEEE, 2016. 5

Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 7345–7353, 2019. 9

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. 24

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, CVPR*, pages 3773–3782, 2021a. 2, 5, 9, 33

Patrick Langechuan Liu. Monocular bev perception with transformers in autonomous driving. https://towardsdatascience.com/monocular-bev-perception-with-transformers-in-autonomous-driving-c41e4a893944, 2021. 4

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. *arXiv preprint arXiv:2112.15351*, 2021b. 5, 15, 31

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV*, pages 10012–10022, 2021c. 4

Annika Meyer, N Ole Salscheider, Piotr F Orzechowski, and Christoph Stiller. Deep semantic lane segmentation for mapless driving. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, pages 869–875. IEEE, 2018. 2

Sergiu Nedevschi, Rolf Schmidt, Thorsten Graf, Radu Danescu, Dan Frentiu, Tiberiu Marita, Florin Oniga, and Ciprian Pocol. 3d lane detection system based on stereovision. In *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems (IEEE Cat. No. 04TH8749)*, pages 161–166. IEEE, 2004. 5

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, 5

Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision, ECCV*, pages 483–499. Springer, 2016. 10

Jiquan Ngiam, Vijay Vasudevan, Benjamin Caine, Zhengdong Zhang, Hao-Tien Lewis Chiang, Jeffrey Ling, Rebecca Roelofs, Alex Bewley, Chenxi Liu, Ashish Venugopal, et al. Scene transformer: A unified architecture for predicting future trajectories of multiple agents. In *International Conference on Learning Representations, ICLR*, 2021. 4

Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Spatial as deep: Spatial cnn for traffic scene understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence, AAAI*, 2018. 2, 4, 5, 11, 13, 31

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 32

Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 7077–7087, 2021. 4

Zequn Qin, Huanyu Wang, and Xi Li. Ultra fast structure-aware deep lane detection. In *European Conference on Computer Vision, ECCV*, pages 276–291. Springer, 2020. 5

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, CVPR*, pages 14122–14130, 2021. 2, 5, 9

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 10

Avishkar Saha, Oscar Mendez Maldonado, Chris Russell, and Richard Bowden. Translating images into maps. *arXiv preprint arXiv:2110.00966*, 2021. 4

Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Rus Daniela. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*. IEEE, 2020. 12

Tixiao Shan, Brendan Englot, Carlo Ratti, and Rus Daniela. Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In *IEEE International Conference on Robotics and Automation, ICRA*. IEEE, 2021. 12

Jinming Su, Chao Chen, Ke Zhang, Junfeng Luo, Xiaoming Wei, and Xiaolin Wei. Structure guided lane detection. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 997–1003, 2021. 2, 5

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, CVPR*, pages 2446–2454, 2020. 11, 12, 13, 16

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, CVPR*, pages 294–302, 2021. 2, 5, 7, 9, 13, 33

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning, ICML*, pages 6105–6114. PMLR, 2019. 7, 24

Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in Neural Information Processing Systems, NeurIPS*, 34, 2021. 12

TuSimple. https://github.com/TuSimple/tusimple-benchmark, 2017. 4, 11

Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. Multi-task learning for dense prediction tasks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021. 9

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems, NeurIPS*, 30, 2017. 3, 4

Jun Wang, Tao Mei, Bin Kong, and Hu Wei. An approach of lane detection based on inverse perspective mapping. In *17th International IEEE Conference on Intelligent Transportation Systems, ITSC*, pages 35–38. IEEE, 2014. 2, 5

Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Conference on Robot Learning, CoRL*, pages 180–191. PMLR, 2022. 4, 8, 32

Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, CVPR*, pages 4724–4732, 2016. 10

Hang Xu, Shaoju Wang, Xinyue Cai, Wei Zhang, Xiaodan Liang, and Zhenguo Li. Curvelane-nas: Unifying lane-sensitive architecture search and adaptive point blending. In *European Conference on Computer Vision, ECCV*, pages 689–704. Springer, 2020. 4, 5, 11, 12

Weixiang Yang, Qi Li, Wenxi Liu, Yuanlong Yu, Yuexin Ma, Shengfeng He, and Jia Pan. Projecting your view attentively: Monocular road scene layout estimation via cross-view transformation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 15536–15545, 2021. 4

Junbo Yin, Jianbing Shen, Chenye Guan, Dingfu Zhou, and Ruigang Yang. Lidar-based online 3d video object detection with graph-based message passing and spatiotemporal transformer attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR*, pages 11495–11504, 2020. 4

Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, CVPR*, pages 2636–2645, 2020a. 4, 11

Zhuoping Yu, Xiaozhou Ren, Yuyao Huang, Wei Tian, and Junqiao Zhao. Detecting lane and road markings at a distance with perspective transformer layers. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems, ITSC*, pages 1–6. IEEE, 2020b. 2

Yujun Zhang, Lei Zhu, Wei Feng, Huazhu Fu, Mingqian Wang, Qingxia Li, Cheng Li, and Song Wang. Vil-100: A new dataset and a baseline model for video instance lane detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision, ICCV*, pages 15681–15690, 2021. 4, 11

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations, ICLR*, 2021. 3, 4, 8, 9, 32, 33, 34

Appendix A. Algorithm

We summarize the details of PersFormer here. We introduce the backbone, overall structure and the unified anchor design. Later we break down the loss function into pieces.

A.1 Backbone

The backbone module is slightly different from previous work (Garnett et al., 2019; Guo et al., 2020), as we need to consider 2D/3D branches together. We use EfficientNet (Tan and Le, 2019) as our backbone, and extract a specific layer as our following module's input. Later we provide two designs, using FPN (Lin et al., 2017) or not. After using several convolution layers, the backbone module outputs 4 different scaled front-view feature maps. Their resolutions are 180×240, 90×120, 45×60, 22×30. Each front-view feature map is then transformed to BEV-space feature map with the help of Perspective Transformer, resulting in 4 BEV feature maps.

A.2 Anchor Details

In this section, we present details of our anchor design, including angles, numbers of anchors and how we associate ground truth lanes with anchors in 2D and 3D. As introduced in the main body of the paper, we first set anchors in BEV space. Following Gen-LaneNet (Guo et al., 2020), the starting positions X^i_{bev} are evenly placed along x-axis with the spacing of 8 pixels. However, we differentiate it from the incline angle φ. Gen-LaneNet sets straightforward (parallel to y-axis) only, which makes it hard to predict lanes with large curvatures or perpendicular lanes. Towards this problem, we put 7 anchors at each X^i_{bev} with different angles, *i.e.*, $\varphi \in \{\pi/2, \arctan(\pm 0.5), \arctan(\pm 1), \arctan(\pm 2)\}$. Note that the angles are in terms of grid coordinates, which is not equal to the absolute values when grids are not square. Moreover, we project all the BEV anchors to image space with average camera height and pitch angle of the dataset, leading to corresponding 2D anchors.

The association between ground truth lanes and anchors is based on the average distance similar to the loss calculation process, instead of assigning the closest anchor at Y_{ref} to ground truths as (Garnett et al., 2019; Guo et al., 2020). The Y_{ref} is set very close to ego-vehicle, *i.e.*, $5m$ in Gen-LaneNet, which makes it better predict lanes in close area while having unsatisfactory performance in the far distance. In our experiments, we assign the anchor with minimum *edit distance* to ground truth lanes in both 2D and 3D tasks. The distance is calculated at fixed y positions: $(5, 10, 15, 20, 30, 40, 50, 60, 80, 100)$ for 3D anchors, and 72 equally sampled heights for 2D anchors.

A.3 Loss Function

We give the details of loss function here. As introduced in the main body of the paper, given the pre-defined y value of the N_d samples along y-axis, the 3D detection head outputs a set of points for each anchor i as following:

$$(\mathbf{x}^i, \mathbf{z}^i, \mathbf{vis}^i_{\text{bev}}) = \{(x^{(i,k)}, z^{(i,k)}, \text{vis}^{(i,k)}_{\text{bev}})\}_{k=1}^{N_d} \tag{7}$$

The y values are $(5, 10, 15, 20, 30, 40, 50, 60, 80, 100)$ in the BEV space, and the size of the BEV space is $20m \times 100m$. Similar to 3D setting, given the pre-defined v value of the N_d

samples along v-axis in front view, the 2D prediction is:

$$(\mathbf{u}^i, \mathbf{vis}^i_{\text{uv}}) = \{(u^{(i,k)}, \text{vis}^{(i,k)}_{\text{uv}})\}_{k=1}^{N_d} \tag{8}$$

The loss is a combination of the 2D lane detection, 3D lane detection and intermediate segmentation with learnable weights (α, β, γ) accordingly:

$$\mathcal{L} = \sum_i \alpha \mathcal{L}_{\text{2D}}(c^i_{\text{2D}}, \mathbf{u}^i, \mathbf{vis}^i_{\text{fv}}) + \beta \mathcal{L}_{\text{ 3D}}\ (c^i_{\text{3D}}, \mathbf{x}^i, \mathbf{z}^i, \mathbf{vis}^i_{\text{bev}}) + \gamma \mathcal{L}_{\text{seg}}(S_{\text{pred}}), \tag{9}$$

where $c^i_{(\cdot)}$ is the predicted lane category in 2D and 3D domain respectively. For $\mathcal{L}_{\text{ 3D}}$, it consists of classification loss, regression loss and visibility loss. The classification loss is a cross-entropy loss, which is as follow:

$$\mathcal{L}_{\text{3D-cls}} = \mathcal{L}_{CE}(c^i_{\text{3D-pred}}, c^i_{\text{3D-gt}}) \tag{10}$$

The regression loss is a L_1 loss, which is as follow:

$$\mathcal{L}_{\text{3D-reg}} = \mathcal{L}_{L_1}(\{\mathbf{x}^i, \mathbf{z}^i\}_{\text{pred}}, \{\mathbf{x}^i, \mathbf{z}^i\}_{\text{gt}}) \tag{11}$$

The visibility loss is a binary cross-entropy loss, which is as follow:

$$\mathcal{L}_{\text{3D-vis}} = \mathcal{L}_{BCE}(\mathbf{vis}^i_{\text{pred}}, \mathbf{vis}^i_{\text{gt}}) \tag{12}$$

The 2D loss functions are similar to the 3D ones, except they are in 2D form:

$$\begin{aligned} \mathcal{L}_{\text{2D-cls}} &= \mathcal{L}_{CE}(c^i_{\text{2D-pred}}, c^i_{\text{2D-gt}}) \\ \mathcal{L}_{\text{2D-reg}} &= \mathcal{L}_{L_1}(\{\mathbf{u}^i\}_{\text{pred}}, \{\mathbf{u}^i\}_{\text{gt}}) \\ \mathcal{L}_{\text{2D-vis}} &= \mathcal{L}_{BCE}(\mathbf{vis}^i_{\text{pred}}, \mathbf{vis}^i_{\text{gt}}) \end{aligned} \tag{13}$$

The segmentation loss is a binary cross-entropy loss as well, which is as follow:

$$\mathcal{L}_{\text{seg}} = \mathcal{L}_{BCE}(S_{\text{pred}}, S_{\text{gt}}) \tag{14}$$

Appendix B. Details on OpenLane Benchmark

In this section, we present more details on dataset statics, our annotation criterion, visualization examples, algorithms we adopted when generating the dataset.

B.1 Dataset Statistics

OpenLane has 1,150 segments with train/validation/test splits of 798/202/150, respectively. Since the test sets are kept for its online leaderboard evaluation, we annotate the other 1,000 segments, *i.e.*, 200K frames at a frequency of 10 FPS, and keep the original train/validation partition for fair comparison with other tasks, such as object detection.

We compute the statistics in OpenLane and visualize them. The overall number of segments with different scene tags is given in Tab. 7. It implies great diversity in data collection and raises higher requirements on the robustness of algorithms. The weather distribution is visually presented in Fig. 7. It shows the benchmark covers various weather

Table 7: Statics of scenario tags. Scene tags are annotated in terms of segments

Tags		Train	Val.	All
Weather	Clear	515	145	660
	Partly cloud	131	28	159
	Overcast	33	8	41
	Rainy	107	18	125
	Foggy	12	3	15
Scene	Residential	270	69	339
	Urban	234	56	290
	Suburbs	259	64	323
	Highway	30	6	36
	Parking lot	5	7	12
Hours	Daytime	653	167	820
	Night	88	22	110
	Dawn/Dusk	57	13	70

conditions and well holds the consistency in the train/validation split. The distribution of the number of lanes in each frame is shown in Fig. 8. About 25% frames of OpenLane have more than 6 lanes, which exceeds the maximum number in most lane datasets. Fig. 9 shows the distribution of lane categories. Single white solid and dash lanes, double yellow solid lanes take up almost 90% of the total lanes. This is imbalanced and yet it falls into a long-tail distribution problem, which is common in realistic scenarios. Fig. 10 presents the distribution of altitude difference per frame. Only around 20% frames are relatively flat with absolute height variation less than $0.5m$, whereas the difference is more than $1m$ in over 50% of OpenLane. This data further demonstrates the necessity of 3D lane detection. The above statistics and examples below demonstrate that OpenLane is the most challenging one compared to existing lane detection datasets.

B.2 Annotation Criterion

We aim at introducing how we annotate lanes, scene tags and CIPO levels in this section. Details such as data structures, folder hierarchy will be provided in the dataset releasing page in the future.

B.2.1 Lanes

Our principle for the 2D lane detection task is to find all visible lanes inside left and right road edges. Following this philosophy, we carefully annotate lanes in each frame. However, due to the complexity of scenarios, there exist some special cases we seek to illustrate here. (1) Lanes are often occluded by objects or invisible because of abrasion but they are still valuable for the real application. Thus we annotate lanes if parts of them are visible, meaning lanes with one side being occluded are extended or lanes with invisible intermediate parts are completed according to the context, as shown in Fig. 11. (2) It is very common that the number of lanes changes, especially when lanes have complex topologies such as fork lanes in merge and split cases. Traditional lane datasets usually omit these scenarios for simplicity, while we keep them all and further choose them out of the whole dataset for evaluation. Fork lanes are annotated as separate lanes with a common starting point (split) or ending point (merge) - two close adjacent lanes are desired for the lane detection

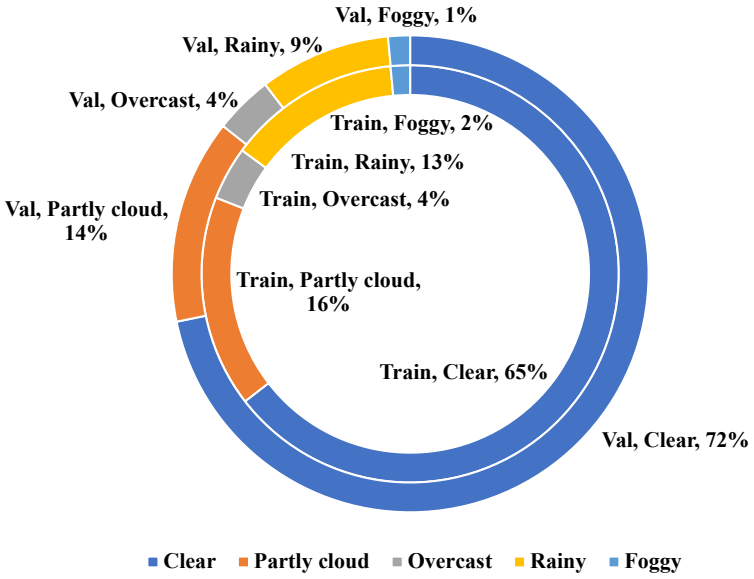

Figure 7: Distribution of weather tags in training and validation sets. The data is collected under different weathers and split into training and validation with great balance

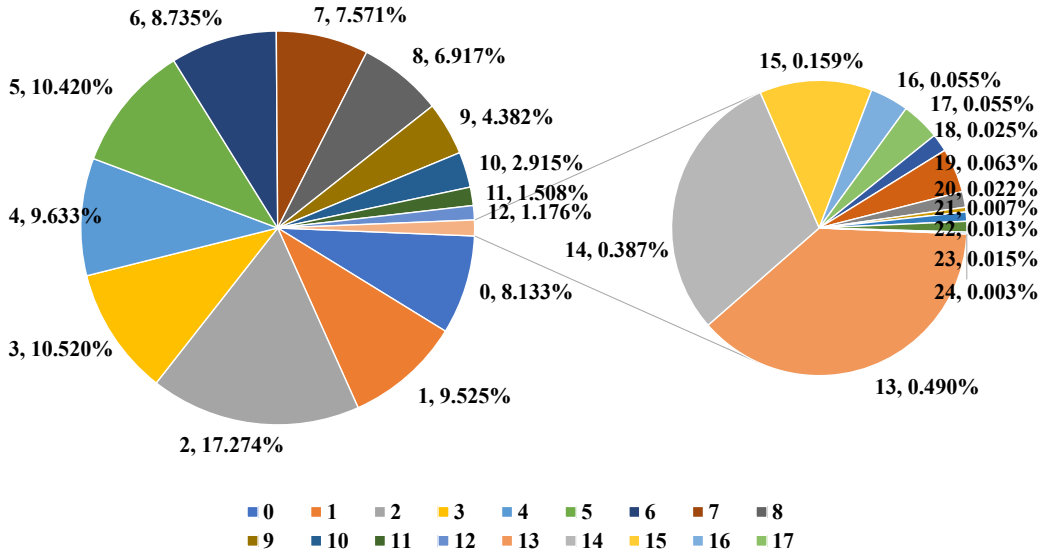

Figure 8: Distribution of lane numbers per frame. The maximum number is 24, and 25% frames have more than 6 lane

methods. (3) We further annotate each lane as one of the 14 lane categories, *i.e.*, single

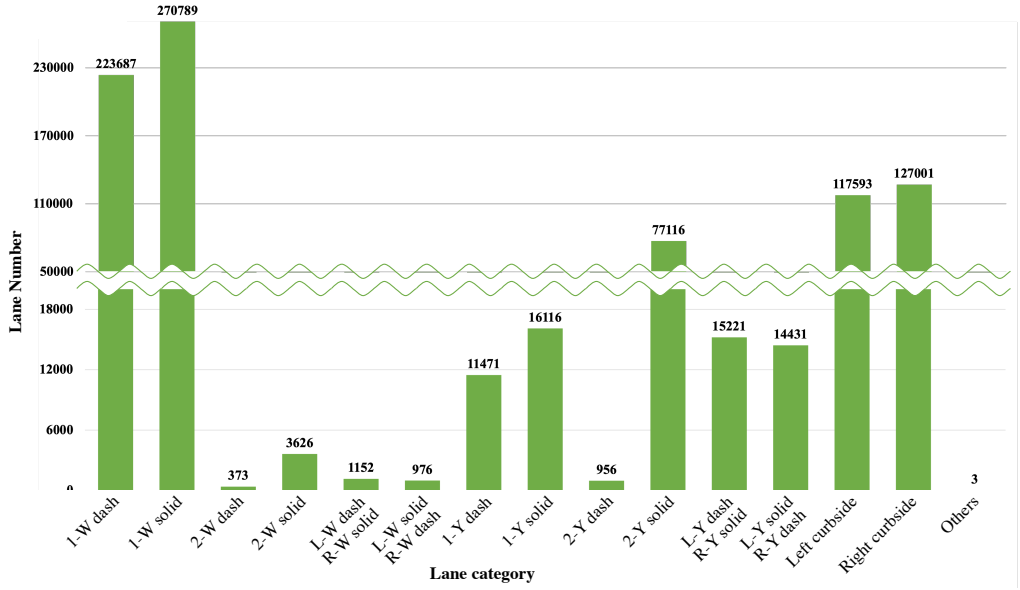

Figure 9: Distribution of the lane category. Here we abbreviate single in *1*, double in *2*, white in *W*, yellow in *Y*, left in *L*, and right in *R*. Thus *1-W dash* means the category of single white dash lanes

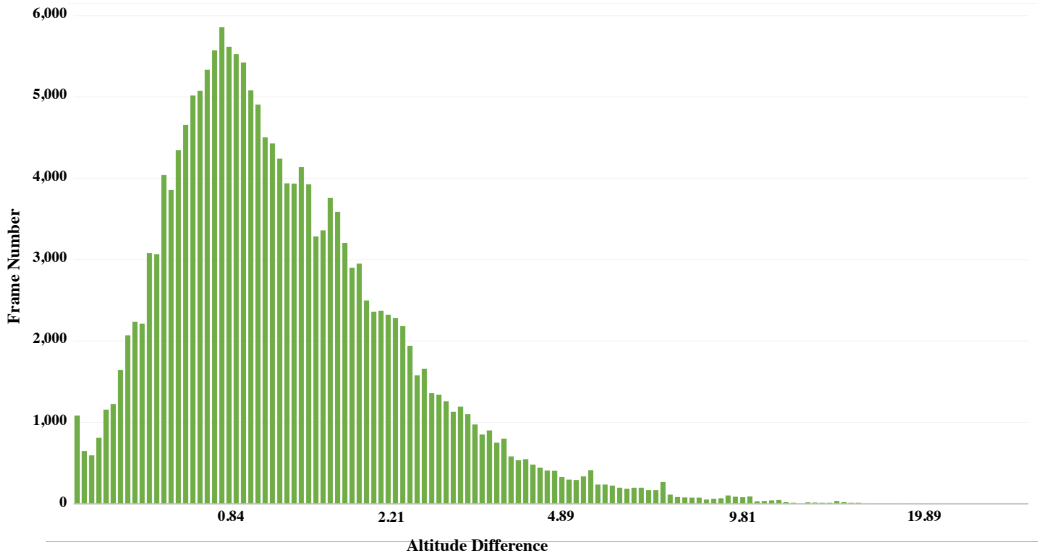

Figure 10: Altitude difference per frame. Note the x-axis is approximately in a log scale and its unit is m

white dash, single white solid, double white dash, double white solid, double white dash solid (left white dash with right white solid), double white solid dash (left white solid with

right white dash), single yellow dash, single yellow solid, double yellow dash, double yellow solid, double yellow dash solid (left yellow dash with right yellow solid), double yellow solid dash (left yellow solid with right yellow dash), left curbside, right curbside. Note that traffic bollards are considered as curbsides as well if they are not temporally placed. (4) Different from all the other lane datasets, we annotate a tracking ID for each lane which is unique across the whole segment. We believe this could be helpful for video lane detection or lane tracking tasks. We also assign a number in 1-4 to the most important 4 lanes based on their relative position to the ego-vehicle. Basically, the left-left lane is 1, the left lane is 2, the right lane is 3, and the right-right lane is 4.

All valid 2D ground truths are transformed to 3D annotations by the generation method in Sec. 4.2 of the main body (Generation of High-quality Annotation), except those without LiDAR points scanning through. Thus the criterion above applies to 3D lanes as well.

B.2.2 Scene tags

We label each segment with 3 scene tags, *i.e.*, weather, scene and hours. We hope these labels can help researchers to investigate the robustness of their models under various scenarios. The statics are shown in Tab. 7. Specifically, the dataset covers 5 different kinds of weather, clear, partly cloud, overcast, rainy and foggy. Note that we classify the video as partly cloud or foggy when there are clouds or fog in the sky respectively, otherwise it will be categorized as overcast. The scene, or the location, includes 5 categories, *i.e.*, residential, urban, suburbs, highway and parking lot. And the hours are divided into 3 parts: daytime, night, dawn/dusk.

B.2.3 Closest-in-path object (CIPO)

CIPO is usually defined as the closest object in ego lane, which refers to a single vehicle only. However, there are cases that vehicles on left/right lanes are intended to cut in which are crucial as well, or there may not be any qualified vehicles in ego lane. To cover the complex scenarios, we categorize objects, mainly including vehicles, pedestrians and cyclists, into 4 different CIPO levels. (1) The most important one, which is closest to ego vehicle within the required reaction distance and has over 50% part of it in the ego lane. Level 1 contains one object at most. (2) Objects are annotated as Level 2 when their bodies interact with the real or virtual lines of ego lane. They are typically in the process of cut-in or cut-out, which hugely influences ego-vehicle decision-making. (3) We consider objects mainly within the reaction distance or drivable area, or those in left/ego/right lanes more specifically. Thus we annotate Level 3 with objects in the above area and having occlusion rate less than 50%. Note that vehicles in the opposite direction can be in this CIPO level as well. (4) The remainings are labeled as Level 4, which means they are almost unlikely to impact the future path at this moment. They are mainly objects in lanes with far distance, objects out of drivable area, or parked vehicles in our dataset. Examples are provided in Fig. 12.

B.3 3D lane Generation

Fig. 13 shows the intermediate results of the generation process of 3D lane labels. However, the above process could have a few problems in some cases, especially in the last step, *i.e.*, smoothing and fitting. Multiple filtering and fitting algorithms are adopted to realize it,

Figure 11: Visualization example of lane annotation in OpenLane dataset

while all of them require a set of sorted points. Due to the large curvature, the 1-1 mapping probably does not stand either in x or y direction, thus we could not sort the points directly. Towards this problem, for each image with this circumstance, we simply find an angle to rotate the whole points set, do the filtering and fitting process in the temporary coordinate and rotate back in the end. This method is illustrated in Fig. 14.

Appendix C. Experiments

C.1 Evaluation Metrics.

For both 3D lane datasets, we follow the evaluation metric designed by Gen-LaneNet (Guo et al., 2020), with additional category accuracy on OpenLane dataset. The matching between prediction and ground truth is built upon *edit distance*, where one predicted lane is considered to be a true positive only if 75% of its covered y-positions have a point-wise

Figure 12: Visualization example of CIPO and Scene tags annotation in OpenLane dataset

distance less than the max-allowed distance ($1.5m$). Then, with the percentage of matched ground-truth lanes as recall and the percentage of matched prediction lanes as precision, we use F-score to report the regression performance of such a model. Since OpenLane dataset has category information per lane, we present the accuracy upon the matched lanes to show classification performance. We only report the accuracy of PersFormer on OpenLane dataset, as other 3D methods do not support classification task. For the 2D task, the classical metric in CULane (Pan et al., 2018) is adopted.

C.2 Implementation Details

To fairly compare with other methods (Guo et al., 2020; Garnett et al., 2019; Liu et al., 2021b), we retain many model settings of image resolution and BEV scale. We resize the original image to 360×480 as model input, project it to BEV space with a resolution of

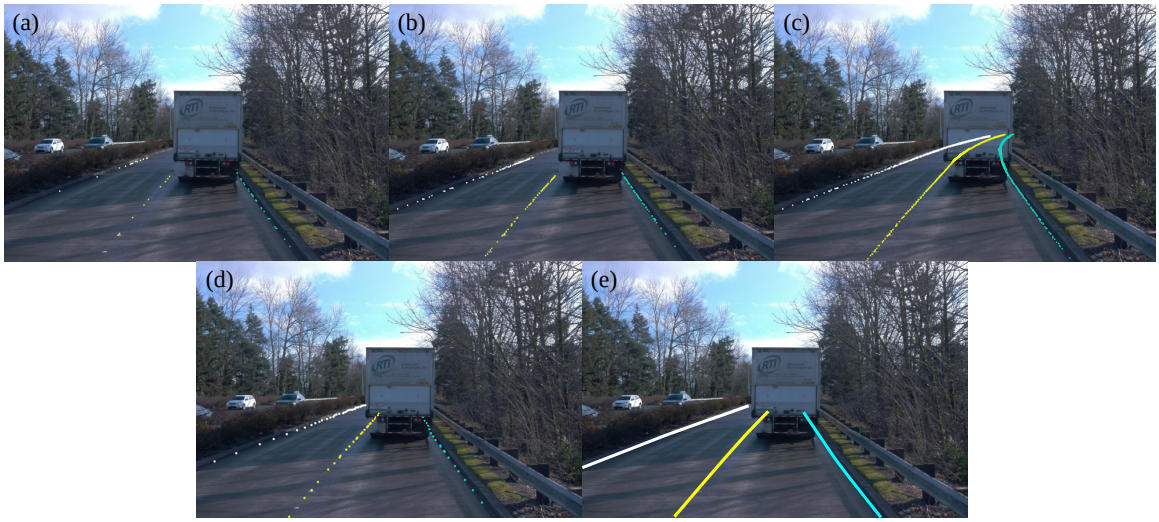

Figure 13: 3D lane generation pipeline. (a) Original point clouds inside a certain threshold of 2D lane annotations are reserved, which is relatively sparse; (b) Positions of points on the 2D annotation are interpolated to get a dense point set; (c) 3D lane points in the same segment are spliced into long, high-density lanes; (d) We remove those too far as they are invisible, while reasonable extensions are desired; (e) A smooth and fitting process is applied to get the final 3D lane annotation

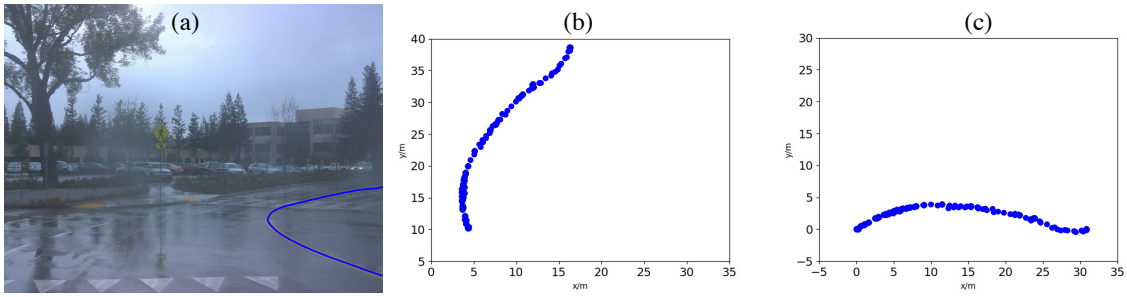

Figure 14: Illustration of 3D lane generation problem with large curvatures. (a) The original image and the 2D lane; (b) Unsorted 3D points set of the lane in (a), which a filtering algorithm is not applicable directly; (c) A simple translation and rotation can result in a 1-1 mapping of x and y

208×108. We use PyTorch (Paszke et al., 2019) to implement the model. The batch size is set to 8; the number of training epochs is set to 100. We re-implement 3D-LaneNet and Gen-LaneNet on OpenLane dataset for a fair comparison. Following previous experience on training vision transformer (Carion et al., 2020; Zhu et al., 2021; Wang et al., 2022), we use Adam optimizer (Kingma and Ba, 2015) with base learning rate of 2×10^{-4}, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and weight decay of 10^{-4}. All of these models are trained on 8 NVIDIA Tesla

Table 8: Comparison with other 2D methods on OpenLane. PersFormer surpasses the baseline model and is on par with SOTA method on 2D lane detection

Method	All	Up & Down	Curve	Extreme Weather	Night	Intersection	Merge & Split
LaneATT-S (Tabelini et al., 2021)	28.3	25.3	25.8	32.0	27.6	14.0	24.3
LaneATT-M (Tabelini et al., 2021)	31.0	28.3	27.4	34.7	30.2	17.0	26.5
CondLaneNet-S (Liu et al., 2021a)	52.3	55.3	57.5	45.8	46.6	48.4	45.5
CondLaneNet-M (Liu et al., 2021a)	55.0	58.5	59.4	49.2	48.6	50.7	47.8
CondLaneNet-L (Liu et al., 2021a)	59.1	62.1	62.9	54.7	51.0	55.7	52.3
PersFormer (ours)	42.0	40.7	46.3	43.7	36.1	28.9	41.2

V100 GPUs. More details about environment setup can be referred to our GitHub repository once accepted.

C.3 More Experimental Results

In this section, we present more experimental results, mainly in more 2D baseline methods comparison, additional ablations and more qualitative examples.

C.3.1 2D Comparisons on OpenLane

In Tab. 8, we compare two current SOTA methods CondLaneNet (Liu et al., 2021a) and LaneATT (Tabelini et al., 2021) on OpenLane dataset. Although PersFormer is focusing on 3D lane detection, its ability on 2D lane is competitive, outperforming baseline method by **11%**.

C.3.2 Ablations

We provide an additional ablative study on the structure of the feature transformation module on a subset of OpenLane (∼300 segments) in Tab. 9. We argue that the IPM-based cross attention is a necessity in PersFormer, as we compare it with two initial designs, naive 1-1 mapping and the learned mapping. The naive 1-1 mapping simply scales every location in the BEV space to the corresponding location in the front view space, not considering camera parameters (Exp.1). A more "aggressive" way to simulate the mapping is directly learning from the front view feature with several fully-connected layers (Exp.2). Neither of them could catch up with the performance of IPM-based mapping, indicating the importance of such a prior in generating BEV feature. We further attempt to adopt Multi-scale Deformable Attention from (Zhu et al., 2021) to implement a several-for-one feature mapping from multi-scale front view feature to multi-scale BEV feature (Exp.3), just like Deformable DETR. The result slightly falls behind our final design (Exp.5), probably due to the influence of tuning of hyper-parameters and the impact of the small-scale feature on the large-scale feature. Finally, we try to remove the classical self attention module in ordinary Transformer design (Exp.4), showing that the self attention module is all there for a reason in Transformer-style structure.

Table 9: Ablative Study on PersFormer Design. IPM prior plays a vital role in guiding the generation of BEV feature compared to naive 1-1 mapping and learned reference-target mapping. Using MSDeformAttn from Deformable DETR (Zhu et al., 2021) to map multi-scale front-view feature to multi-scale BEV feature is competitive, and the self-attention module of BEV query is important in Transformer-style structure

Exp.	Naive 1-1	Learned	Multi-to-Multi	Self Attn.	IPM Prior	3D F-Score
1	✓					36.15
2		✓				13.45
3			✓			51.35
4				✓		47.18
5					✓	52.68

C.3.3 Visualization

We provide qualitative results compared with SOTA 3D lane detection methods in different evaluation scenarios on OpenLane dataset in Fig. 15,16. Results on Apollo 3D synthetic dataset are shown in Fig. 17. We can observe that PersFormer could achieve higher accuracy and capture more lanes to reconstruct the scenes on both datasets.

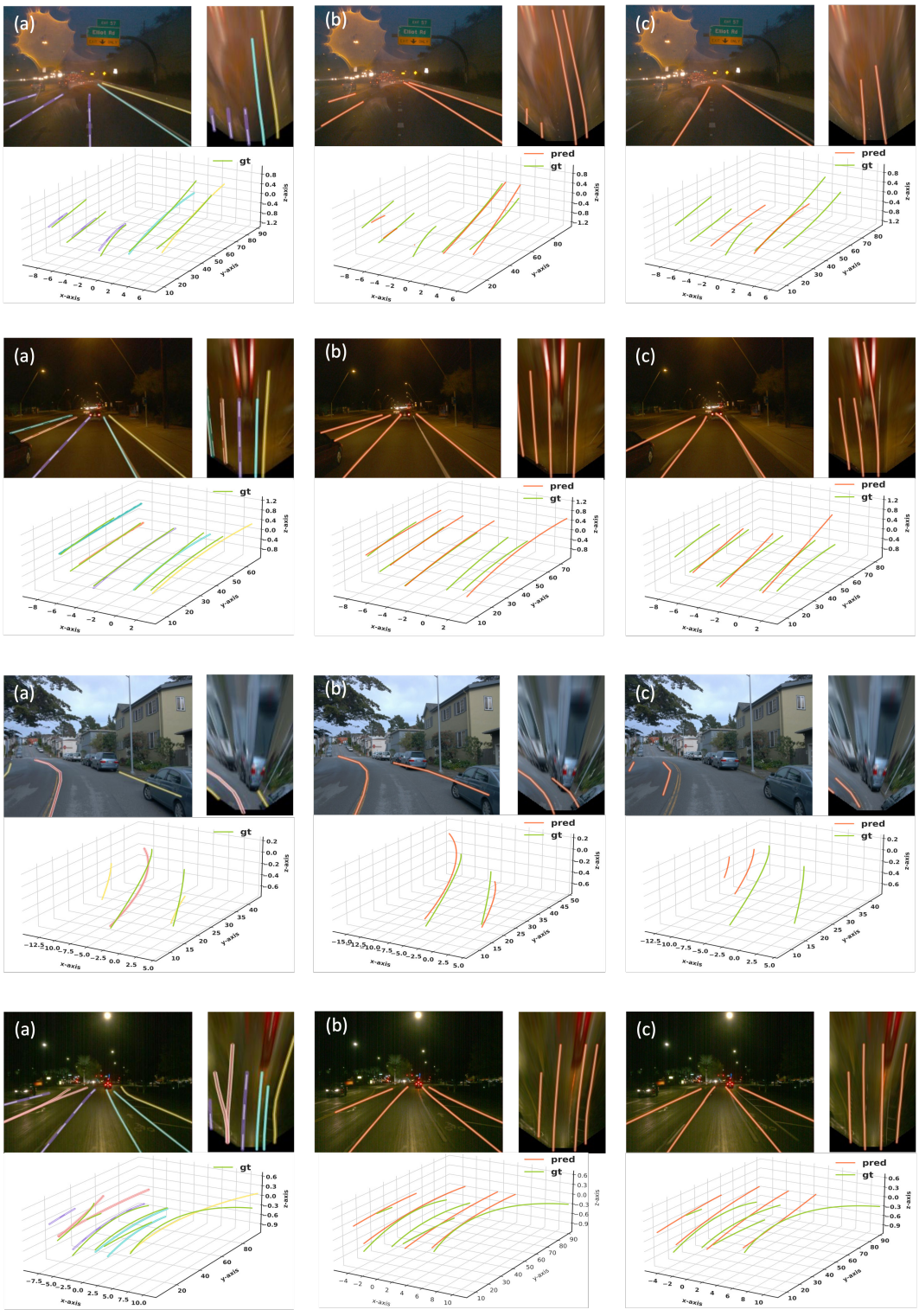

Figure 15: Qualitative results of PersFormer(a), 3D-LaneNet(b) (Garnett et al., 2019), and Gen-LaneNet(c) (Guo et al., 2020) on OpenLane. Night case and Up&Down case

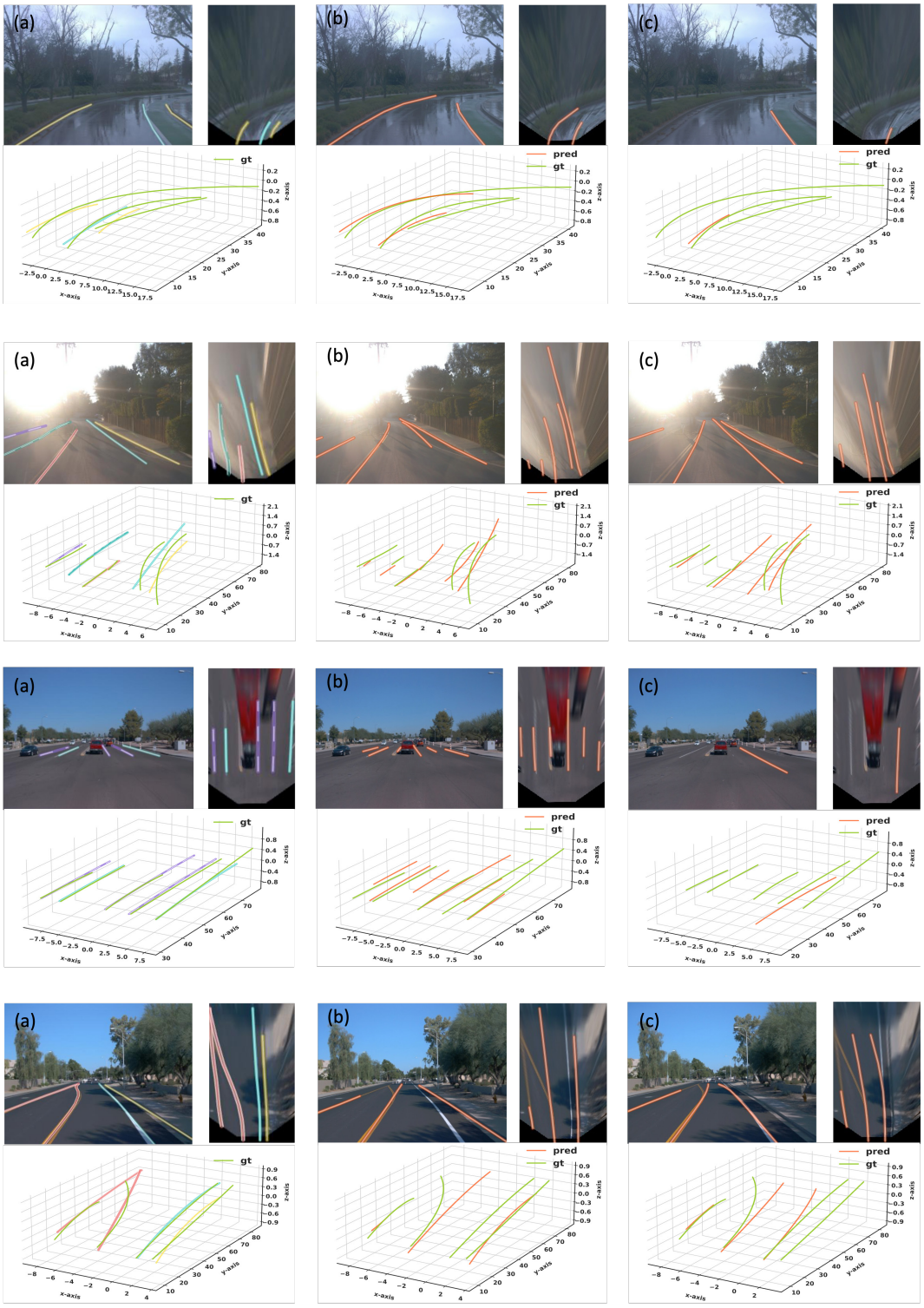

Figure 16: Qualitative results of PersFormer(a), 3D-LaneNet(b) (Garnett et al., 2019), and Gen-LaneNet(c) (Guo et al., 2020) on OpenLane. Extreme weather case, Intersection case and Merge&Split case

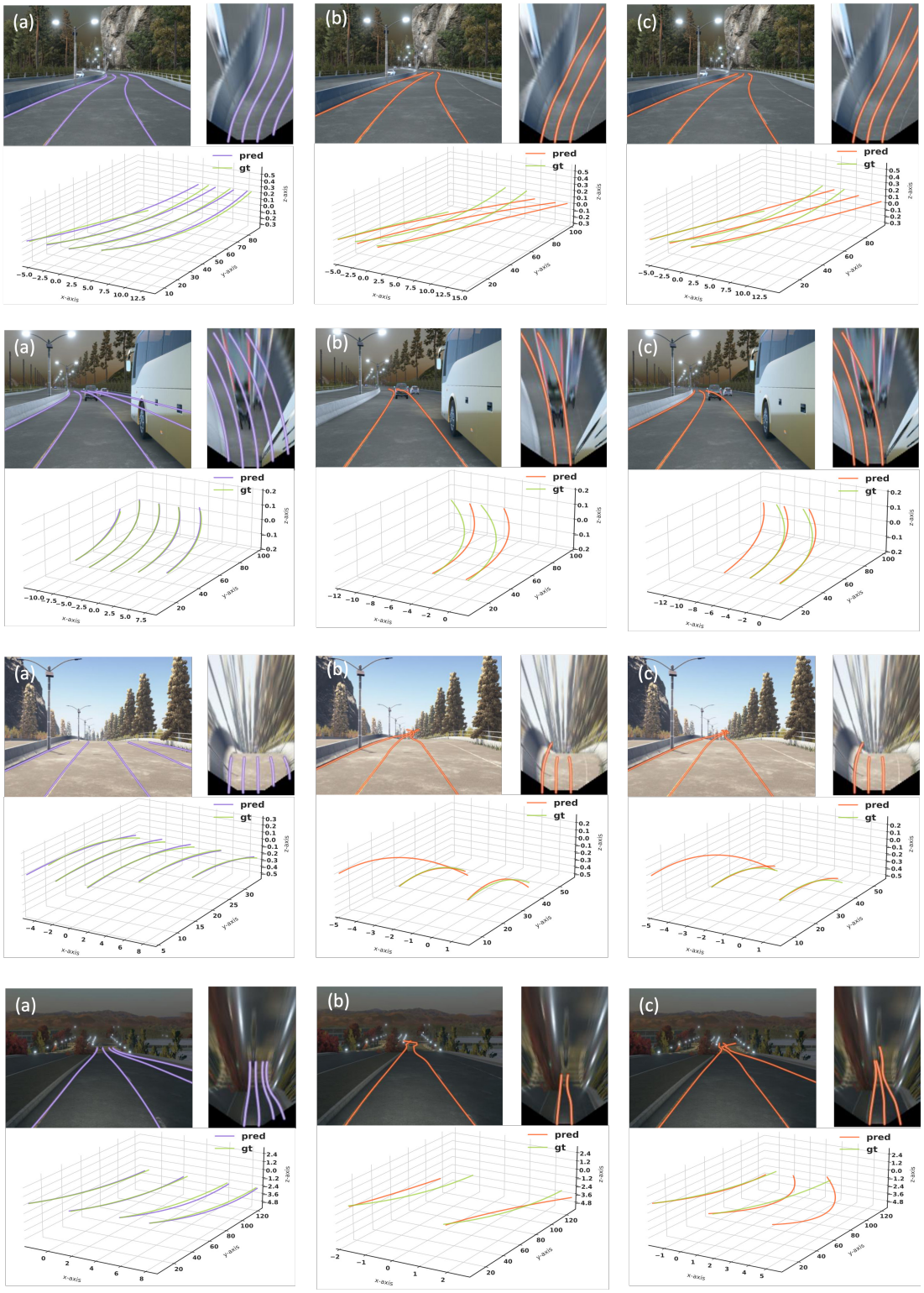

Figure 17: Qualitative results of PersFormer(a), 3D-LaneNet(b) (Garnett et al., 2019), and Gen-LaneNet(c) (Guo et al., 2020) on Apollo. Curve case and Up&Down case