Anchor3DLane: Learning to Regress 3D Anchors for Monocular 3D Lane Detection

Shaofei Huang[1,2] Zhenwei Shen[3*] Zehao Huang[3] Zi-han Ding[4,5]
Jiao Dai[1,2] Jizhong Han[1,2] Naiyan Wang[3] Si Liu[4,5]
[1] Institute of Information Engineering, Chinese Academy of Sciences
[2] School of Cyber Security, University of Chinese Academy of Sciences
[3] TuSimple [4] Institute of Artificial Intelligence, Beihang University
[5] Hangzhou Innovation Institute, Beihang University
{nowherespyfly, zehaohuang18, zihanding819, winsty}@gmail.com
shenzhenwei@outlook.com {hanjizhong, daijiao}@iie.ac.cn liusi@buaa.edu.cn

Abstract

Monocular 3D lane detection is a challenging task due to its lack of depth information. A popular solution is to first transform the front-viewed (FV) images or features into the bird-eye-view (BEV) space with inverse perspective mapping (IPM) and detect lanes from BEV features. However, the reliance of IPM on flat ground assumption and loss of context information make it inaccurate to restore 3D information from BEV representations. An attempt has been made to get rid of BEV and predict 3D lanes from FV representations directly, while it still underperforms other BEV-based methods given its lack of structured representation for 3D lanes. In this paper, we define 3D lane anchors in the 3D space and propose a BEV-free method named Anchor3DLane to predict 3D lanes directly from FV representations. 3D lane anchors are projected to the FV features to extract their features which contain both good structural and context information to make accurate predictions. In addition, we also develop a global optimization method that makes use of the equal-width property between lanes to reduce the lateral error of predictions. Extensive experiments on three popular 3D lane detection benchmarks show that our Anchor3DLane outperforms previous BEV-based methods and achieves state-of-the-art performances. The code is available at: https://github.com/tusen-ai/Anchor3DLane.

1. Introduction

Monocular 3D lane detection, which aims at estimating the 3D coordinates of lane lines from a frontal-viewed image, is one of the essential modules in autonomous driving systems. Accurate and robust perception of 3D lanes is not only critical for stable lane keeping, but also serves as an important component for downstream tasks like high-definition map construction [21, 25], and trajectory planning [1, 40]. However, due to the lack of depth information, estimating lanes in 3D space directly from 2D image domain still remains very challenging.

A straightforward way to tackle the above challenges is to detect lanes from the bird-eye-viewed (BEV) space.

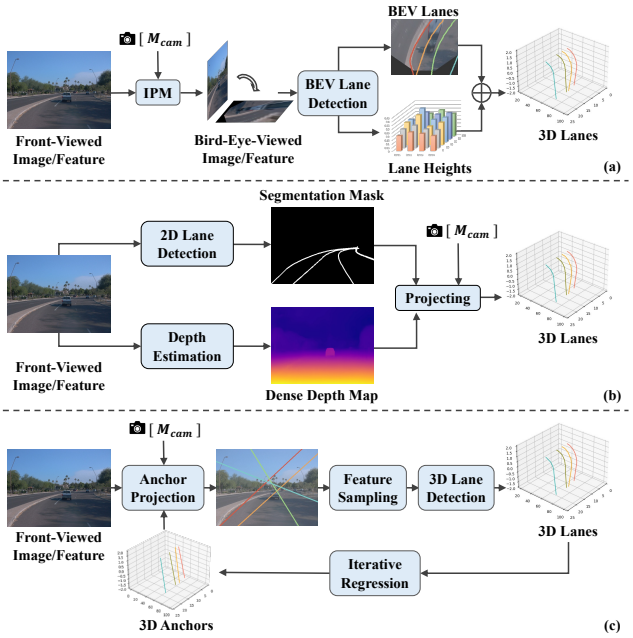

Figure 1. (a) BEV-based methods, which perform lane detection in the warped BEV images or features. (b) Non-BEV method, which projects 2D lane predictions back to 3D space with estimated depth. (c) Our Anchor3DLane projects 3D anchors into FV features to sample features for 3D prediction directly.

*Work done while at TuSimple

As illustrated in Figure 1(a), a common practice of BEV-based methods [5, 7, 8, 20] is to warp images or features from frontal-viewed (FV) space to BEV with inverse perspective mapping (IPM), thereby transforming the 3D lane detection task into 2D lane detection task in BEV. To project the detected BEV lanes back into 3D space, coordinates of the lane points are then combined with their corresponding height values which are estimated by a height estimation head. Though proven effective, their limitations are still obvious: (1) IPM relies on a strict assumption of flat ground, which does not hold true for uphill or downhill cases. (2) Since IPM warps the images on the basis of ground, some useful height information as well as the context information above the road surface are lost inevitably. For example, objects like vehicles on the road are severely distorted after warping. Therefore, information lost brought by IPM hinders the accurate restoration of 3D information from BEV representations.

Given the above limitations of BEV, some works tried to predict 3D lanes from FV directly. As illustrated in Figure 1(b), SALAD [36] decomposes 3D lane detection task into 2D lane segmentation and dense depth estimation. The segmented 2D lanes are projected into 3D space with camera intrinsic parameters and the estimated depth information. Even though getting rid of the flat ground assumption, SALAD lacks structured representations of 3D lanes. As a result, it is unnatural to extend it to more complex 3D lane settings like multi-view or multi-frame. Moreover, their performance is still far behind the state-of-the-art methods due to the unstructured representation.

In this paper, we propose a novel BEV-free method named Anchor3DLane to predict 3D lanes directly from FV concisely and effectively. As shown in Figure 1(c), our Anchor3DLane defines lane anchors as rays in the 3D space with given pitches and yaws. Afterward, we first project them to corresponding 2D points in FV space using camera parameters, and then obtain their features by bilinear sampling. A simple classification head and a regression head are adopted to generate classification probabilities and 3D offsets from anchors respectively to make final predictions. Unlike the information loss in IPM, sampling from original FV features retains richer context information around lanes, which helps estimate 3D information more accurately. Moreover, our 3D lane anchors can be iteratively refined to sample more accurate features to better capture complex variations of 3D lanes. Furthermore, Anchor3DLane can be easily extended to the multi-frame setting by projecting 3D anchors to adjacent frames with the assistance of camera poses between frames, which further improves performances over single-frame prediction.

In addition, we also utilize global constraints to refine the challenging distant parts due to low resolution. The motivation is based on an intuitive insight that lanes in the same image appear to be parallel in most cases except for the fork lanes, i.e., distances between different point pairs on each lane pair are nearly consistent. By applying a global equal-width optimization to non-fork lane pairs, we adjust 3D lane predictions to make the width of lane pairs consistent from close to far. The lateral error of distant parts of lane lines can be further reduced through the above adjustment.

Our contributions are summarized as follows:

- We propose a novel Anchor3DLane framework that directly defines anchors in 3D space and regresses 3D lanes directly from FV without introducing BEV. An extension to the multi-frame setting of Anchor3DLane is also proposed to leverage the well-aligned temporal information for further performance improvement.
- We develop a global optimization method to utilize the equal-width properties of lanes for refinement.
- Without bells and whistles, our Anchor3DLane outperforms previous BEV-based methods and achieves state-of-the-art performances on three popular 3D lane detection benchmarks.

2. Related Works

2.1. 2D Lane Detection

2D lane detection [12, 22, 24, 30, 37] aims at obtaining the accurate shape and locations of 2D lanes in the images. Earlier works [2, 10, 13, 34, 39] mainly focus on extracting low-level handcrafted features, such as edge and color information. However, these approaches often have complex feature extraction and post-processing designs and are less robust under changing scenarios. With the development of deep learning, CNN-based methods have been explored recently and achieve notable performance. Segmentation-based methods [11, 23, 24, 26] formulate 2D lane detection task as a per-pixel classification problem and typically focus on how to explore more effective and semantically informative features. To make predictions more sparse and flexible, keypoint-based methods [15, 27, 33, 35] model lane lines as sets of ordered keypoints and associate keypoints belonged to the same lane together by postprocessing. Apart from the above methods, anchor-based methods [17, 19, 29, 38] are also popular in 2D lane detection task due to their conciseness and effectiveness. LineCNN [17] first defines straight rays emitted from the image boundary to fit the shape of 2D lane lines and applies Non-Maximum Suppression (NMS) to keep only lanes with higher confidence. LaneATT [29] develops anchor-based feature pooling to extract features for the 2D anchors. CLRNet [38] learns to refine the initial anchors iteratively through the feature pyramid.

2.2. 3D Lane Detection

Since projecting 2D lanes back into 3D space suffers from inaccuracy as well as less robustness, 3D lane detection task is proposed to predict lanes in 3D space end to end. Some works utilize multiple sensors, such as stereo cameras [4] and Lidar-camera [3] to restore 3D information. However, the collection and annotation cost of multi-sensor data is expensive, restricting the practical application of these methods. Therefore, monocular camera image based 3D lane detection [6–8,20,36] attracts more attention.

Due to the good geometric properties of lanes in the perspective of BEV, 3DLaneNet [7] utilizes IPM to transform features from FV into BEV and then regresses the anchor offsets of lanes in the BEV space. CLGo [20] transforms raw images into BEV images with the estimated camera pitches and heights and fits the lane lines by predicting polynomial parameters. Since IPM relies heavily on the flat ground assumption, lanes represented in BEV space may be misaligned with 3D space in rough ground cases. To this end, Gen-LaneNet [8] makes a distinction between the virtual top view generated by IPM and the true top view in 3D space for better space alignment. Persformer [5] utilizes deformable attention to generate BEV features more adaptively and robustly. SALAD [36] tries to get rid of BEV by decomposing 3D lane detection into 2D lane segmentation and dense depth estimation tasks. Different from the above methods, our Anchor3DLane defines anchors in the 3D space to explicitly model 3D lanes and bridge the gap between FV space and 3D space. The projection and sampling operations ensure the accuracy of anchor feature extraction, enabling effectively predicting 3D lanes directly from FV representations without introducing BEV.

3. Method

The overall architecture of our Anchor3DLane is illustrated in Figure 3. Given a front-viewed image $\mathbf{I} \in \mathbb{R}^{H\times W\times 3}$ as input, where H and W denote the height and width of the input image, a CNN backbone (e.g., ResNet-18 [9]) is adopted to extract 2D visual features represented in FV space. To enlarge the receptive field of the network, we further insert a single Transformer layer [32] after the backbone to obtain the enhanced 2D feature map $\mathbf{F} \in \mathbb{R}^{H_f\times W_f\times C}$, where H_f, W_f, and C represent the height, width and channel number of feature map respectively. 3D anchors are then projected to this feature map $\mathbf{F}$ with the assistance of camera parameters, and the corresponding anchor features are sampled using bilinear interpolation. Afterward, we apply a classification head and a regression head to the sampled anchor features to make predictions, with each head composed of several lightweight fully connected layers. Furthermore, the predictions can be regarded as new 3D anchors for iterative regression.

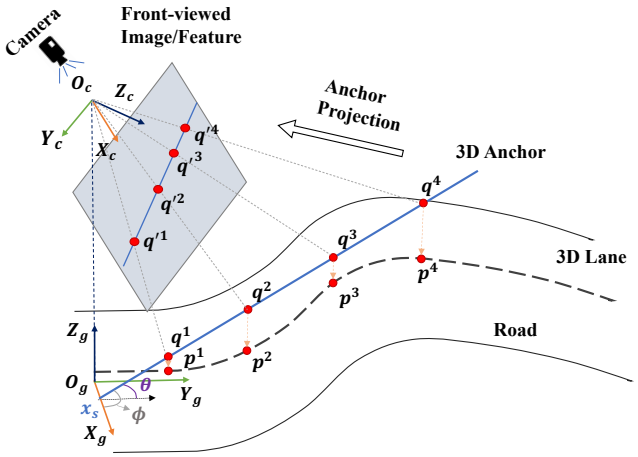

Figure 2. Illustration of 3D anchor and 3D lane in the ground coordinate system.

3.1. 3D Lane Representation

We first revisit the representation of 3D lanes in this section. As shown in Figure 2, two different coordinate systems are involved in our paper, including the camera coordinate system and the ground coordinate system. The camera coordinate directly corresponds with the FV image and is a right-handed coordinate system defined by origin O_c and X_c, Y_c, Z_c axes, with O_c located at the center of the camera and Z_c pointing forward vertical to the camera plane. 3D lanes are commonly annotated in the ground coordinate system, of which the origin O_g is set right below O_c, x-axis X_g points positive to the right, y-axis Y_g points positive forwards and z-axis Z_g points positive upwards. A 3D lane is described by 3D points with N uniformly sampled y-coordinates $\mathbf{y} = \{y^k\}_{k=1}^N$. Thus, we denote the i-th 3D lane as $\mathbf{G}_i = \{\mathbf{p}_i^k\}_{k=1}^N$ and its k-th point is represented as $\mathbf{p}_i^k = (x_i^k, y^k, z_i^k, vis_i^k)$, where the first 3 elements denote the location of $\mathbf{p}_i^k$ in the ground coordinate system and the last one denotes the visibility of $\mathbf{p}_i^k$. It is worth noting that we elaborate our method based on the ground coordinate system following the common practices adopted in previous works [7, 8]. However, our Anchor3DLane is able to work in an arbitrary 3D coordinate system as long as camera calibration is available.

3.2. Anchor3DLane

3.2.1 Representation of 3D Lane Anchors

Our 3D lane anchors are defined in the same coordinate system as 3D lanes, i.e., ground coordinate, for ease of position regression. As illustrated in Figure 2, a 3D anchor is a ray starting from $(x_s, 0, 0)$ with pitch θ and yaw ϕ. Similar to 3D lanes, we also sample N points for each anchor by the same y-coordinates and represent the j-th 3D anchor by $\mathbf{A}_j = \{\mathbf{q}_j^k\}_{k=1}^N$, and its k-th point is denoted by

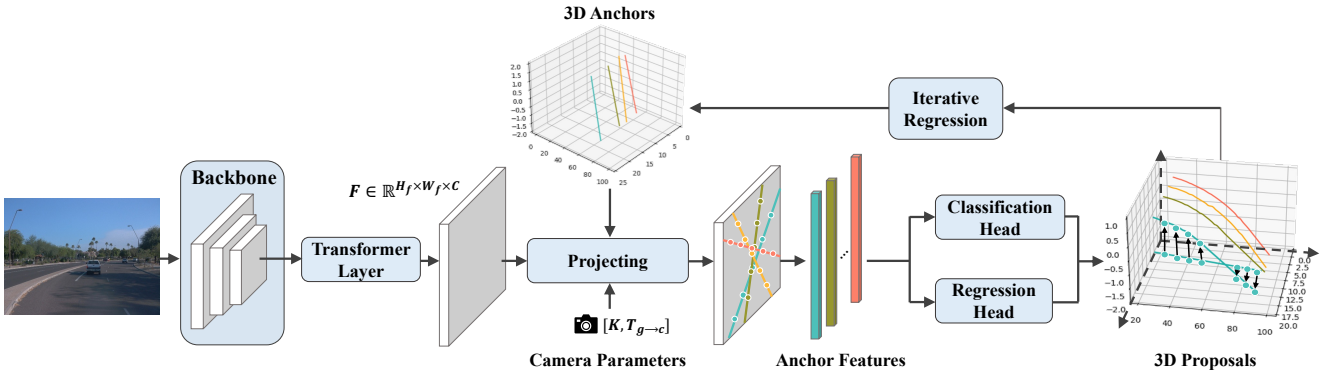

Figure 3. The overall architecture of Anchor3DLane. Given a front-viewed input image, a CNN backbone and a Transformer layer are adopted to first extract visual feature $\mathbf{F}$. 3D anchors are then projected to sample their features from $\mathbf{F}$ given camera parameters. Afterward, a classification head and a regression head are applied to make the final predictions. The lane predictions can also serve as new 3D anchors for iterative regression.

$\mathbf{q}_j^k = (x_j^k, y^k, z_j^k)$. Different from previous works [5, 7] that define anchors in the BEV plane, our 3D anchors have pitches to the ground and could fit the lane shape better.

3.2.2 Anchor Projection and Feature Sampling

To obtain features of 3D anchors, we first project them into the plane of FV feature $\mathbf{F}$ using camera parameters as shown in Figure 2. Given an anchor $\mathbf{A}_j$, we take its k-th point $\mathbf{q}_j^k$ as an example to explain the projection operation and omit the subscript j for simplicity as follows:

$$\begin{bmatrix} \tilde{u}^k \\ \tilde{v}^k \\ d^k \end{bmatrix} = \mathbf{K}\mathbf{T}_{g\to c} \begin{bmatrix} x^k \\ y^k \\ z^k \\ 1 \end{bmatrix}, \tag{1}$$

$$u^k = W_f/W \cdot \frac{\tilde{u}^k}{d^k}, \tag{2}$$

$$v^k = H_f/H \cdot \frac{\tilde{v}^k}{d^k}, \tag{3}$$

where $\mathbf{K} \in \mathbb{R}^{3\times3}$ denotes camera intrinsic parameters, $\mathbf{T}_{g\to c} \in \mathbb{R}^{3\times4}$ denotes the transform matrix from ground coordinate to camera coordinate, and d^k denotes the depth of $\mathbf{q}^k$ to the camera plane. Through the above formulations, $\mathbf{q}^k$ is projected to position (u^k, v^k) in FV feature $\mathbf{F}$. Finally, the feature of anchor $\mathbf{A}_j$ is obtained through bilinear interpolation within the neighborhood of the projected points and is represented as $\{\mathbf{F}_{(u^k,v^k)}\}_{k=1}^N$.

3.2.3 3D Lane Prediction

We concatenate features of points belonging to the same anchor as its feature representation. Then we apply a classification head and a regression head to the anchor features for predicting classification probabilities $\mathbf{c}_j \in \mathbb{R}^L$, anchor points offsets $(\Delta\mathbf{x}_j \in \mathbb{R}^N, \Delta\mathbf{z}_j \in \mathbb{R}^N) = \{(\Delta x_j^k, \Delta z_j^k)\}_{k=1}^N$ and visibility of each point $\mathbf{vis}_j \in \mathbb{R}^N$ respectively, with $j \in [1, M]$. L and M denote the numbers of lane types and 3D anchors respectively. In this way, 3D lane proposals are generated as $\{\mathbf{P}_j = (\mathbf{c}_j, \mathbf{x}_j + \Delta\mathbf{x}_j, \mathbf{y}, \mathbf{z}_j + \Delta\mathbf{z}_j, \mathbf{vis}_j)\}_{j=1}^M$. Furthermore, these 3D lane proposals can also serve as new anchors in the following iterative regression steps as illustrated in Figure 3. Through iterative regression, proposals can be refined progressively to better fit the lane shape.

During training, we associate n nearest anchors to each ground-truth lane and the rest are defined as negative samples. Distance metric between ground-truth $\mathbf{G}_i$ and anchor $\mathbf{A}_j$ is calculated as follows:

$$D(\mathbf{G}_i, \mathbf{A}_j) = \frac{\sum_{k=1}^N vis_i^k \cdot \sqrt{(x_i^k - x_j^k)^2 + (z_i^k - z_j^k)^2}}{\sum_{k=1}^N vis_i^k}. \tag{4}$$

This metric is also used in Non-Maximum Suppression (NMS) during inference to keep a reasonable number of proposals except that distances are calculated between visible parts of two proposals.

We adopt focal loss [18] for training classification to balance the positive and negative proposals as follows:

$$\mathcal{L}_{cls} = -\sum_{j=1}^{M}\sum_{l=1}^{L} \alpha^l (1 - c_j^l)^\gamma \log c_j^l, \tag{5}$$

where α^l and γ are the hyperparamters for focal loss. The regression loss is only calculated between the positive proposals and their assigned ground-truth lanes following [8]:

$$\mathcal{L}_{reg} = \sum_{i=1}^{M_p}\sum_{k=1}^{N}(\|\hat{vis}_i^k \cdot (x_i^k + \Delta x_i^k - \hat{x}_i^k)\|_1 + \sum_{i=1}^{M_p}\sum_{k=1}^{N}\|\hat{vis}_i^k \cdot (z_i^k + \Delta z_i^k - \hat{z}_i^k)\|_1) + \sum_{i=1}^{M_p}\sum_{k=1}^{N}\|\hat{vis}_i^k - vis_i^k\|_1. \quad (6)$$

M_p represents the total number of positive proposals. Here we use $\hat{x}_i^k$, $\hat{z}_i^k$ and $\hat{vis}_i^k$ to denote the x, z coordinates and visibility of the ground-truth lane points.

The total loss function of our Anchor3DLane is a combination of the above two losses with corresponding coefficients:

$$\mathcal{L} = \lambda_{cls}\mathcal{L}_{cls} + \lambda_{reg}\mathcal{L}_{reg}. \quad (7)$$

3.3. Temporal Context Modeling

Thanks to the design of 3D anchors, our Anchor3DLane can be easily extended to multi-frame 3D lane detection. Given a 3D point (x_t, y_t, z_t) in the t-th frame's ground coordinate system, we transform it to the t'-th frame's ground coordinate system with the following formulation:

$$\begin{bmatrix} x_{t'} \\ y_{t'} \\ z_{t'} \end{bmatrix} = \mathbf{T}_{g(t)\to g(t')} \begin{bmatrix} x_t \\ y_t \\ z_t \\ 1 \end{bmatrix}, \quad (8)$$

where $\mathbf{T}_{g(t)\to g(t')} \in \mathbb{R}^{3\times 4}$ denotes the transformation matrix from t-th frame to t'-th frame. Together with Equation 1, anchors defined in the current frame can be projected to previous frames for sampling their features. For each anchor, we take its points from the current frame as query and points from previous frames as key and value to conduct cross-frame attention for feature aggregation. By integrating the well-aligned anchor features from multiple frames, temporal context is incorporated into our Anchor3DLane to enlarge its perception range.

3.4. Optimization with Equal-Width Constraint

In most cases, lanes in 3D space are nearly parallel with each other, which is helpful in generating robust 3D estimations from monocular 2D images. In this work, we leverage this geometry property of 3D lanes and formulate it as an equal-width constraint to adjust the x-coordinates of lane predictions. Given two lane predictions $\mathbf{P}_j = \{\mathbf{p}_j^k\}_{k=1}^N$ and $\mathbf{P}_{j'} = \{\mathbf{p}_{j'}^k\}_{k=1}^N$, width between $\mathbf{P}_j$ and $\mathbf{P}_{j'}$ at point pair $\mathbf{p}_j^k$ and $\mathbf{p}_{j'}^k$ is calculated as:

$$w_{j,j'}^k = |\cos\theta_j^k(x_j^k + \tilde{\Delta}x_j^k - x_{j'}^k - \tilde{\Delta}x_{j'}^k)|, \quad (9)$$

where $\tilde{\Delta}x_j^k$ and $\tilde{\Delta}x_{j'}^k$ denote the adjustment to x_j^k and $x_{j'}^k$ to be optimized respectively and θ_j^k denotes the normal direction of the adjusted lane at $\mathbf{p}_j^k$. The objective function of equal-width constraint is as follows:

$$\min_{\{\tilde{\Delta}\mathbf{x}_j\}_{j\in[1,Q]}} \frac{1}{Q(Q-1)}\sum_{j=1}^{Q}\sum_{j'=1,j'\neq j}^{Q}\mathcal{L}(\mathbf{w}_{j,j'}) + \alpha\frac{1}{Q}\sum_{j=1}^{Q}\|\tilde{\Delta}\mathbf{x}_j\|_2, \quad (10)$$

where

$$\mathcal{L}(\mathbf{w}_{j,j'}) = \sum_{k=1}^{N}|w_{j,j'}^k - \frac{1}{N}\sum_{k'=1}^{N}w_{j,j'}^{k'}|. \quad (11)$$

We use Q to denote the number of lane predictions after NMS. $\mathcal{L}(\mathbf{w}_{j,j'})$ restricts the width between $\mathbf{P}_j$ and $\mathbf{P}_{j'}$ to be consistent and the second item serves as a regularization to avoid the adjusted results being too far from the original predictions. We run this optimization as a post-processing step to refine the prediction results of the network.

4. Experiments

4.1. Experimental Setting

4.1.1 Datasets and Evaluation Metrics

We conduct experiments on three popular 3D lane detection benchmarks, including ApolloSim [8], OpenLane [5], and ONCE-3DLanes [36].

ApolloSim is a photo-realistic synthetic dataset created with Unity 3D engine which contains 10.5K images from various virtual scenes, including highway, urban, residential, downtown, etc. In addition, the data is also diverse in daytime, weather conditions, traffic/obstacles, and road surface qualities.

OpenLane is a large-scale real-world 3D lane detection dataset constructed upon the Waymo Open dataset [28]. It contains 200K frames and over 880K lanes are annotated. Camera intrinsics and extrinsics are provided for each frame. All lanes are annotated including lanes in the opposite direction if no curbside in the middle. Categories and scene tags (e.g., weather and locations) are also provided.

ONCE-3DLanes is a real-world 3D lane detection dataset with 1 million scenes. It consists of 211K images with labeled 3D lane points. It covers different time periods (sunny, cloudy, rainy) and various regions (downtown, suburbs, highway, bridges, and tunnels). Only camera intrinsics are provided in ONCE-3DLanes.

During the evaluation, the predictions and ground truth lanes are matched via minimum-cost flow where the pairwise cost is defined as the square root of the sum of pointwise Euclidean distance. A prediction is considered as true

Scene	Method	AP(%)↑	F1(%)↑	x err/C(m) ↓	x err/F(m) ↓	z err/C(m) ↓	z err/F(m) ↓
Balanced Scene	3DLaneNet [7]	89.3	86.4	0.068	0.477	0.015	**0.202**
	Gen-LaneNet [8]	90.1	88.1	0.061	0.496	0.012	0.214
	CLGo [20]	94.2	91.9	0.061	0.361	0.029	0.250
	PersFormer [5]	-	92.9	0.054	0.356	0.010	0.234
	GP [16]	93.8	91.9	0.049	0.387	**0.008**	0.213
	Anchor3DLane (Ours)	**97.2**	**95.6**	0.052	0.306	0.015	0.223
	Anchor3DLane†(Ours)	97.1	95.4	**0.045**	**0.300**	0.016	0.223
Rare Subset	3DLaneNet [7]	74.6	72.0	0.166	0.855	0.039	**0.521**
	Gen-LaneNet [8]	79.0	78.0	0.139	0.903	0.030	0.539
	CLGo [20]	88.3	86.1	0.147	0.735	0.071	0.609
	PersFormer [5]	-	87.5	0.107	0.782	0.024	0.602
	GP [16]	85.2	83.7	0.126	0.903	**0.023**	0.625
	Anchor3DLane (Ours)	**96.9**	**94.4**	0.094	**0.693**	0.027	0.579
	Anchor3DLane† (Ours)	95.9	**94.4**	**0.082**	0.699	0.030	0.580
Visual Variations	3D-LaneNet [7]	74.9	72.5	0.115	0.601	0.032	0.230
	Gen-LaneNet [8]	87.2	85.3	0.074	0.538	0.015	0.232
	CLGo [20]	89.2	87.3	0.084	0.464	0.045	0.312
	PersFormer [5]	-	89.6	0.074	0.430	0.015	0.266
	GP [16]	92.1	89.9	0.060	0.446	**0.011**	0.235
	Anchor3DLane (Ours)	**93.6**	91.4	0.068	0.367	0.020	0.232
	Anchor3DLane† (Ours)	92.5	**91.8**	**0.047**	**0.327**	0.019	**0.219**

Table 1. Comparison with state-of-the-art methods on ApolloSim dataset with three different split settings. "C" and "F" are short for close and far respectively. † denotes iterative regression.

positive if over 75% of its points' distances to ground-truth points are less than a threshold, i.e., 1.5m. With the definition above, Average Precision (AP) and the maximum F1 score are further calculated, and x/z errors are counted separately at close (0-40m) and far (40-100m) ranges. We report the results of F1 score, AP, and x/z-errors on ApolloSim dataset. On OpenLane dataset, except for F1 score and x/z errors, we further report category accuracy which calculates the proportion of predictions whose categories are correctly predicted to all true positive predictions. ONCE-3DLanes adopts a different way to match predictions and ground truth lanes. The matching degree is firstly decided by IoU on the top-view plane and pairs above the threshold are further calculated with their unilateral Chamfer Distance (CD) as the matching error. A true positive is counted when CD is under the threshold. We report F1 score, precision, recall, and CD error for results on ONCE-3DLanes.

4.1.2 Implementation Details

We choose ResNet-18 [9] as the backbone of our Anchor3DLane. To maintain feature resolution, we set the downsampling stride of its last two stages to 1 and replace the 3×3 convolutions with dilated convolutions. The starting positions x_s of 3D anchors are evenly placed along the x-axis with an interval of 1.3m. For each x_s, different yaws $\phi \in \{0°, \pm1°, \pm3°, \pm5°, \pm7°, \pm10°, \pm15°, \pm20°, \pm30°\}$ and pitches $\theta \in \{0°, \pm1°, \pm2°, \pm5°\}$ are set respectively. The number of points N for each anchor is set to 10 for experiments on ApolloSim and ONCE and 20 for OpenLane. We resize the image to 360×480 before feeding it to the backbone and the shape of $\mathbf{F}$ is $45 \times 60 \times 64$. During training, λ_{cls} and λ_{reg} are both set to 1 and the number of positive proposals is set as 3. The distance threshold for NMS is 2 during inference. For multi-frame Anchor3DLane, each time we randomly select 1 frame from the previous 5 frames to interact with current frame during training, and select the first frame of the previous 5 frames during inference. Since car poses are only available in OpenLane dataset, we only conduct temporal experiments on this dataset. We use Adam optimizer [14] with weight decay set as $1e^{-4}$, and set the initial learning rate to $1e^{-4}$. Step learning rate decay is used during training. α^l is set to 0.5 and γ is set to 2 for focal loss. More details about our Anchor3DLane are included in supplementary materials.

4.2. Quantitative Results

4.2.1 Results on ApolloSim

Table 1 shows the experimental results under three different split settings of the ApolloSim dataset, including balanced scene, rare subset and visual variations. We report the results of both our original Anchor3DLane and Anchor3DLane with iterative regression optimized with equal-width constraint. It is shown that our original Anchor3DLane outperforms previous methods with large margins on AP and F1 score on all the three splits with simple design, i.e., +3.0% AP and +2.7% F1 score on bal-

Method	F1(%)↑	Cate Acc(%)↑	x err/C(m) ↓	x err/F(m) ↓	z err/C(m) ↓	z err/F(m) ↓
3D-LaneNet [7]	44.1	-	0.479	0.572	0.367	0.443
GenLaneNet [8]	32.3	-	0.591	0.684	0.411	0.521
PersFormer [5]	50.5	**92.3**	0.485	0.553	0.364	0.431
Anchor3DLane (Ours)	53.1	90.0	0.300	0.311	**0.103**	0.139
Anchor3DLane† (Ours)	53.7	90.9	0.276	0.311	0.107	0.138
Anchor3DLane-T† (Ours)	**54.3**	90.7	**0.275**	**0.310**	0.105	**0.135**

Table 2. Comparison with state-of-the-art methods on OpenLane validation set. † denotes iterative regression. Anchor3DLane-T denotes incorporating multi-frame information. "Cate Acc" means category accuracy.

Method	All	Up & Down	Curve	Extreme Weather	Night	Intersection	Merge & Split
3D-LaneNet [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 [5]	50.5	42.4	55.6	48.6	46.6	40.0	50.7
Anchor3DLane (Ours)	53.1	45.5	56.2	51.9	47.2	44.2	50.5
Anchor3DLane† (Ours)	53.7	46.7	57.2	52.5	47.8	45.4	51.2
Anchor3DLane-T† (Ours)	**54.3**	**47.2**	**58.0**	**52.7**	**48.7**	**45.8**	**51.7**

Table 3. Comparison with state-of-the-art methods on OpenLane validation set. F1 score is presented for each scenario. † denotes iterative regression. Anchor3DLane-T denotes incorporating multi-frame information.

anced scene, $+8.6\%$ AP and $+6.9\%$ F1 score on rare subset, $+2.4\%$ F1 score and $+1.5\%$ AP on visual variations, showing the superiority of our method. Our Anchor3DLane also achieves comparable or lower x/z errors compared with previous methods, especially for x error far, indicating regressing over 3D anchors have greater advantages for distant predictions. Furthermore, by iteratively regressing over the proposals predicted by Anchor3DLane, x/z errors can be further reduced to better fit the shape of 3D lanes.

4.2.2 Results on OpenLane

We present the experimental results of our method optimized with the equal-width constraint on OpenLane dataset in Table 2. Our original Anchor3DLane outperforms PersFormer by 2.6% F1 score improvement. Moreover, our method achieves much more precise predictions than PersFormer, i.e., -0.185m on x error close, -0.242m on x error far, -0.261m on z error close, and -0.292m on z error far respectively, which are crucial for driving safety. The gap in x/z errors indicates that under real scenarios with diverse conditions, directly sampling features from FV representation could maintain more environment context information, thus producing more precise predictions. By incorporating iterative regression and temporal information in Anchor3DLane, the overall performances can be further boosted. In Table 3, we compare with previous methods under different scenarios and report F1 score for each scenario. Our method produces much better performance in Up&Down scenarios, showing the advantage of 3D anchor regression in uneven ground. It is also worth noting that we adopt a lightweight CNN, i.e., ResNet-18 as the backbone of Anchor3DLane, which still outperforms PersFormer with a larger backbone, i.e., EfficientNet-B7 [31].

4.2.3 Results on ONCE-3DLanes

In Table 4, we present the experimental results on the ONCE-3DLanes dataset. Since camera extrinsics are not available in ONCE-3DLanes, we define the 3D anchors in the camera coordinate system and make predictions in the same space. Our method also achieves state-of-the-art performances on this dataset. Compared with PersFormer, our Anchor3DLane still produces a higher F1 score and reduces CD error by 18.9% relatively, which indicates that 3D anchors are able to adapt different 3D coordinate systems.

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 [36]	64.07	75.90	55.42	0.098
PersFormer [5]	74.33	80.30	69.18	0.074
Anchor3DLane (Ours)	74.44	80.50	69.23	0.064
Anchor3DLane† (Ours)	**74.87**	**80.85**	**69.71**	**0.060**

Table 4. Comparison with state-of-the-art methods on ONCE-3DLanes validation set. Results under $\tau_{CD} = 0.3$ are displayed here. † denotes iterative regression. "P" and "R" are short for precision and recall respectively.

4.2.4 Ablation Study

In this section, we follow previous work [5] to conduct most ablation studies on OpenLane-300, which is a sub-

set of OpenLane. As for feature sampling experiments, we present the results on the original OpenLane to verify the effectiveness of our method. More ablation studies and qualitative results are included in the supplementary materials.

Input	Feat	F1(%)	x err/C(m)	x err/F(m)	z err/C(m)	z err/F(m)
BEV	BEV	47.6	0.466	0.421	0.119	0.170
FV	BEV	47.6	0.443	0.446	0.118	0.160
FV	FV	**53.1**	**0.300**	**0.31**	**0.103**	**0.139**

Table 5. Comparison between sampling anchor features from BEV features and FV features.

Sampling anchor features from FV features. To illustrate the superiority of FV features, we compare the performances of extracting anchor features from FV features and BEV features. The results are shown in Table 5. We explore different ways of obtaining BEV features, including warping FV image to BEV image (line 1) and warping FV feature to BEV feature (line 2), and keep the other settings same as our original Anchor3DLane. Results show that sampling anchor features from FV features produces the best F1 score and x/z errors, especially for x errors, where more than 10cm gap exists between FV anchor features and BEV anchor features. The above performance gap indicates that the context information contained in raw FV features is beneficial for accurate lane predictions.

Iter	F1(%)	x err/C(m)	x err/F(m)	z err/C(m)	z err/F(m)
1	54.8	0.318	0.349	**0.101**	**0.147**
2	56.3	**0.287**	0.335	0.103	0.152
3	**57.0**	**0.287**	**0.327**	0.104	0.148

Table 6. Ablation study on the steps of iterative regression.

Steps of iterative regression. Table 6 presents the results of different steps of iterative regression for Anchor3DLane. Compared with no iterative regression, 2 iterations produces relatively large performance improvements. More steps of iterative regression can further reduce lateral errors as well as elevate F1 score by refining the shape of proposals progressively.

Method	F1(%)	x err/C(m)	x err/F(m)	z err/C(m)	z err/F(m)
w/o Temporal	54.8	0.318	0.349	0.101	0.147
Linear Fusion	54.9	0.322	0.343	0.102	0.148
Weighted Sum	**55.8**	0.320	0.346	0.101	0.150
Attention	55.2	**0.308**	**0.330**	**0.099**	**0.145**

Table 7. Ablation study on temporal integration methods.

Temporal integration methods. In this section, we explore different methods to integrate anchor features of multiple frames. Besides the cross-frame attention that we mentioned in Section 3.3, we also try *linear fusion* which concatenates features of the same anchor along their channels and fuses them with a linear layer, and *weighted sum* which learns to predict a group of weights for each y-coordinate to fuse features of the same anchor elementwisely, As shown in Table 7, comparing with the baseline, incorporating temporal information into Anchor3DLane can improve the overall performance significantly due to the richer context information obtained from previous frames. Weighted sum produces better results than linear fusion, indicating that dynamic weights are necessary for different points at different distances. Although weighted sum achieves a better F1 score compared with single frame setting, x/z errors increase at the same time. Among the 3 integration methods, cross-frame attention, which aggregates anchor features with more anchor points from previous frames, improves both F1 score and x errors and achieves the best performance balance.

Method	F1(%)	x err/C(m)	x err/F(m)
w/o EWC	54.8	**0.318**	0.349
w/ EWC	**55.0**	**0.318**	**0.337**

Table 8. Ablation study on Equal-Width Constraint (EWC).

Effect of equal-width constraint. We also illustrate the comparison between predictions without and with equal-width constraint optimization. As shown in Table 8, by applying the equal-width constraint to the lane predictions, errors of the distant parts of the lane lines can be further reduced by restricting them to have the same width as the close parts. More visualization results of this constraint can be found in the supplementary materials.

5. Conclusion

In this work, we propose a novel Anchor3DLane framework for 3D lane detection which bypasses the transformation to BEV space and predicts 3D lanes from FV directly. By defining anchors in the 3D space and projecting them to the FV features, accurate anchor features are sampled for lane prediction. We further extend our Anchor3DLane to the multi-frame setting to incorporate temporal information, which improves performances due to the enriched context. In addition, a global equal-width optimization method is proposed to utilize the parallel property of lanes for refinement. Experimental results show that our Anchor3DLane outperforms previous methods on three 3D lane detection benchmarks with a simple architecture.

6. Acknowledgments

This research was supported in part by National Key R&D Program of China (2022ZD0115502), National Natural Science Foundation of China (Grant No. 62122010), Zhejiang Provincial Natural Science Foundation of China under Grant No. LDT23F02022F02, Key Research and Development Program of Zhejiang Province under Grant No. 2022C01082.

References

[1] Florent Altché and Arnaud de La Fortelle. An LSTM network for highway trajectory prediction. In *ITSC*, 2017. 1

[2] Mohamed Aly. Real time detection of lane markers in urban streets. In *IV*, 2008. 2

[3] Min Bai, Gellert Mattyus, Namdar Homayounfar, Shenlong Wang, Shrinidhi Kowshika Lakshmikanth, and Raquel Urtasun. Deep multi-sensor lane detection. In *IROS*, 2018. 3

[4] Nabil Benmansour, Raphaël Labayrade, Didier Aubert, and Sébastien Glaser. Stereovision-based 3d lane detection system: a model driven approach. In *ITSC*, 2008. 3

[5] Li Chen, Chonghao Sima, Yang Li, Zehan Zheng, Jiajie Xu, Xiangwei Geng, Hongyang Li, Conghui He, Jianping Shi, Yu Qiao, et al. PersFormer: 3d lane detection via perspective transformer and the OpenLane benchmark. In *ECCV*, 2022. 2, 3, 4, 5, 6, 7

[6] 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. 3

[7] Noa Garnett, Rafi Cohen, Tomer Pe'er, Roee Lahav, and Dan Levi. 3D-LaneNet: end-to-end 3d multiple lane detection. In *CVPR*, 2019. 2, 3, 4, 6, 7

[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 *ECCV*, 2020. 2, 3, 4, 5, 6, 7

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 6

[10] Yinghua He, Hong Wang, and Bo Zhang. Color-based road detection in urban traffic scenes. *T-ITS*, 2004. 2

[11] Yuenan Hou, Zheng Ma, Chunxiao Liu, and Chen Change Loy. Learning lightweight lane detection CNNs by self attention distillation. In *ICCV*, 2019. 2

[12] Dongkwon Jin, Wonhui Park, Seong-Gyun Jeong, Heeyeon Kwon, and Chang-Su Kim. EigenLanes: Data-driven lane descriptors for structurally diverse lanes. In *CVPR*, 2022. 2

[13] ZuWhan Kim. Robust lane detection and tracking in challenging scenarios. *T-ITS*, 2008. 2

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[15] Yeongmin Ko, Younkwan Lee, Shoaib Azam, Farzeen Munir, Moongu Jeon, and Witold Pedrycz. Key points estimation and point instance segmentation approach for lane detection. *T-ITS*, 2021. 2

[16] Chenguang Li, Jia Shi, Ya Wang, and Guangliang Cheng. Reconstruct from top view: A 3d lane detection approach based on geometry structure prior. In *CVPR*, 2022. 6

[17] Xiang Li, Jun Li, Xiaolin Hu, and Jian Yang. Line-CNN: End-to-end traffic line detection with line proposal unit. *T-ITS*, 2019. 2

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 4

[19] Lizhe Liu, Xiaohao Chen, Siyu Zhu, and Ping Tan. CondLaneNet: a top-to-down lane detection framework based on conditional convolution. In *ICCV*, 2021. 2

[20] 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 *AAAI*, 2022. 2, 3, 6

[21] Rong Liu, Jinling Wang, and Bingqi Zhang. High definition map for automated driving: Overview and analysis. *J. Navig.*, 2020. 1

[22] Ruijin Liu, Zejian Yuan, Tie Liu, and Zhiliang Xiong. End-to-end lane shape prediction with transformers. In *CVPR*, 2021. 2

[23] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards end-to-end lane detection: an instance segmentation approach. In *IV*, 2018. 2

[24] Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Spatial as deep: Spatial CNN for traffic scene understanding. In *AAAI*, 2018. 2

[25] Tong Qin, Tongqing Chen, Yilun Chen, and Qing Su. AVP-SLAM: Semantic visual mapping and localization for autonomous vehicles in the parking lot. In *IROS*, 2020. 1

[26] Zequn Qin, Huanyu Wang, and Xi Li. Ultra fast structure-aware deep lane detection. In *ECCV*, 2020. 2

[27] Zhan Qu, Huan Jin, Yang Zhou, Zhen Yang, and Wei Zhang. Focus on local: Detecting lane marker from bottom up via key point. In *CVPR*, 2021. 2

[28] 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 *CVPR*, 2020. 5

[29] 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 *CVPR*, 2021. 2

[30] 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 *ICPR*, 2021. 2

[31] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 7

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3

[33] 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 *CVPR*, 2022. 2

[34] Yue Wang, Eam Khwang Teoh, and Dinggang Shen. Lane detection and tracking using B-Snake. *IVC*, 2004. 2

[35] Shenghua Xu, Xinyue Cai, Bin Zhao, Li Zhang, Hang Xu, Yanwei Fu, and Xiangyang Xue. RCLane: Relay chain prediction for lane detection. In *ECCV*, 2022. 2

[36] 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 *CVPR*, 2022. 2, 3, 5, 7

[37] Jiaxing Yang, Lihe Zhang, and Huchuan Lu. Lane detection with versatile atrousformer and local semantic guidance. *Pattern Recognition*, 2023. 2

[38] Tu Zheng, Yifei Huang, Yang Liu, Wenjian Tang, Zheng Yang, Deng Cai, and Xiaofei He. CLRNet: Cross layer refinement network for lane detection. In *CVPR*, 2022. 2

[39] Shengyan Zhou, Yanhua Jiang, Junqiang Xi, Jianwei Gong, Guangming Xiong, and Huiyan Chen. A novel lane detection based on geometrical model and gabor filter. In *IV*, 2010. 2

[40] Sheng Zhu and Bilin Aksun-Guvenc. Trajectory planning of autonomous vehicles based on parameterized control optimization in dynamic on-road environments. *J INTELL ROBOT SYST*, 2020. 1