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

Shaofei Huang¹,²  Zhenwei Shen³* Zehao Huang³  Zi-han Ding⁴,⁵
Jiao Dai¹,²  Jizhong Han¹,²  Naiyan Wang³  Si Liu⁴,⁵
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 driv-

*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 offsets of lanes in the BEV space. The projection and sampling operations ensure the accuracy of anchor feature extraction 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.

2.3. 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 \( y = \{y_k\}_{k=1}^{N} \). Thus, we denote the \( i \)-th 3D lane as \( \mathbf{G}_i = \{\mathbf{p}_k^i\}_{k=1}^{N} \) and its \( k \)-th point is represented as \( \mathbf{p}_k^i = (x_k^i, y_k^i, z_k^i, \text{vis}_k^i) \), where the first 3 elements denote the location of \( \mathbf{p}_k^i \) in the ground coordinate system and the last one denotes the visibility of \( \mathbf{p}_k^i \). 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 \( \theta \) and yaw \( \phi \). 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 \( \mathbf{A}_j = \{\mathbf{q}_k^j\}_{k=1}^{N} \), and its \( k \)-th point is denoted by
\[ q_j^k = (x_j^k, y_j^k, z_j^k). \] 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 \(F\) using camera parameters as shown in Figure 2. Given an anchor \(A_j\), we take its \(k\)-th point \(q_j^k\) as an example to explain the projection operation and omit the subscript \(j\) for simplicity as follows:

\[
\begin{bmatrix}
\hat{u}^k \\
\hat{v}^k \\
\hat{d}^k
\end{bmatrix} = K T_{g \rightarrow c} \begin{bmatrix}
x_j^k \\
y_j^k \\
z_j^k
\end{bmatrix},
\]

\[ u^k = \frac{W_f}{W} \cdot \frac{\hat{u}^k}{\hat{d}^k}, \]

\[ v^k = \frac{H_f}{H} \cdot \frac{\hat{v}^k}{\hat{d}^k}, \]

where \(K \in \mathbb{R}^{3 \times 3}\) denotes camera intrinsic parameters, \(T_{g \rightarrow c} \in \mathbb{R}^{3 \times 4}\) denotes the transform matrix from ground coordinate to camera coordinate, and \(d^k\) denotes the depth of \(q^k\) to the camera plane. Through the above formulations, \(q^k\) is projected to position \((u^k, v^k)\) in FV feature \(F\). Finally, the feature of anchor \(A_j\) is obtained through bilinear interpolation within the neighborhood of the projected points and is represented as \(\{F(u^k, v^k)\}_{k=1}^N\).

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 \(c_j \in \mathbb{R}^L\), anchor points offsets \((\Delta x_j, \Delta z_j) \in \mathbb{R}^N\) and visibility of each point \(\text{vis}_j \in \mathbb{R}^N\), 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 \(\{P_j = (c_j, x_j + \Delta x_j, y_j, z_j + \Delta z_j, \text{vis}_j)\}_{j=1}^M\). 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 \(G_i\) and anchor \(A_j\) is calculated as follows:

\[
D(G_i, A_j) = \frac{\sum_{k=1}^N \text{vis}_i^k \cdot \sqrt{(u_i^k - u_j^k)^2 + (v_i^k - v_j^k)^2}}{\sum_{k=1}^N \text{vis}_i^k}.
\]

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:

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

where \(\alpha^l\) and \(\gamma\) are the hyperparameters for focal loss. The regression loss is only calculated between the positive proposals and their assigned ground-truth lanes following \([5]\):
predictions. Given two lane predictions and an equal-width constraint to adjust the x-coordinates of lane geometry this property of 3D lanes and formulate it as tions from monocular 2D images. In this work, we lever-
each other, which is helpful in generating robust 3D estima-
tions. During the evaluation, the predictions and ground truth
lanes are matched via minimum-cost flow where the pair-
wise Euclidean distance. A prediction is considered as true
where $\tilde{\Delta}x^k_j$ and $\tilde{\Delta}x^k_{j'}$ denote the adjustment to $x^k_j$ and $x^k_{j'}$ to be optimized respectively and $\theta^k_j$ denotes the normal direction of the adjusted lane at $p^k_j$. The objective function of equal-width constraint is as follows:

$$\min_{(\Delta x_j)_{j \in [1, Q]}} \frac{1}{Q(Q-1)} \sum_{j=1}^{Q} \sum_{j'=1, j' \neq j}^{Q} L(w_{j,j'})$$

$$+ \alpha \frac{1}{Q} \sum_{j=1}^{Q} ||\tilde{\Delta}x_j||_2,$$

where

$$L(w_{j,j'}) = \sum_{k=1}^{N} |w^k_{j,j'} - \frac{1}{N} \sum_{k'=1}^{N} w^k_{j,j'}|.$$  

We use $Q$ to denote the number of lane predictions after NMS. $L(w_{j,j'})$ restricts the width between $P_j$ and $P_{j'}$ 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 point-wise Euclidean distance. A prediction is considered as true
<table>
<thead>
<tr>
<th>Scene</th>
<th>Method</th>
<th>AP(%)†</th>
<th>F1(%)†</th>
<th>x err/C(m)↓</th>
<th>x err/F(m)↓</th>
<th>z err/C(m)↓</th>
<th>z err/F(m)↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balanced Scene</td>
<td>3DLaneNet [7]</td>
<td>89.3</td>
<td>86.4</td>
<td>0.068</td>
<td>0.477</td>
<td>0.015</td>
<td>0.202</td>
</tr>
<tr>
<td></td>
<td>Gen-LaneNet [8]</td>
<td>90.1</td>
<td>88.1</td>
<td>0.061</td>
<td>0.496</td>
<td>0.012</td>
<td>0.214</td>
</tr>
<tr>
<td></td>
<td>CLGo [20]</td>
<td>94.2</td>
<td>91.9</td>
<td>0.061</td>
<td>0.361</td>
<td>0.029</td>
<td>0.250</td>
</tr>
<tr>
<td></td>
<td>PersFormer [5]</td>
<td>-</td>
<td>92.9</td>
<td>0.054</td>
<td>0.356</td>
<td>0.010</td>
<td>0.234</td>
</tr>
<tr>
<td></td>
<td>GP [16]</td>
<td>93.8</td>
<td>91.9</td>
<td>0.049</td>
<td>0.387</td>
<td>0.008</td>
<td>0.213</td>
</tr>
<tr>
<td></td>
<td>Anchor3DLane (Ours)</td>
<td>97.2</td>
<td>95.6</td>
<td>0.052</td>
<td>0.306</td>
<td>0.015</td>
<td>0.223</td>
</tr>
<tr>
<td></td>
<td>Anchor3DLane†(Ours)</td>
<td>97.1</td>
<td>95.4</td>
<td>0.045</td>
<td>0.300</td>
<td>0.016</td>
<td>0.223</td>
</tr>
<tr>
<td>Rare Subset</td>
<td>3DLaneNet [7]</td>
<td>74.6</td>
<td>72.0</td>
<td>0.166</td>
<td>0.855</td>
<td>0.039</td>
<td>0.521</td>
</tr>
<tr>
<td></td>
<td>Gen-LaneNet [8]</td>
<td>79.0</td>
<td>78.0</td>
<td>0.139</td>
<td>0.903</td>
<td>0.030</td>
<td>0.539</td>
</tr>
<tr>
<td></td>
<td>CLGo [20]</td>
<td>88.3</td>
<td>86.1</td>
<td>0.147</td>
<td>0.735</td>
<td>0.071</td>
<td>0.609</td>
</tr>
<tr>
<td></td>
<td>PersFormer [5]</td>
<td>-</td>
<td>87.5</td>
<td>0.107</td>
<td>0.782</td>
<td>0.024</td>
<td>0.602</td>
</tr>
<tr>
<td></td>
<td>GP [16]</td>
<td>85.2</td>
<td>83.7</td>
<td>0.126</td>
<td>0.903</td>
<td>0.023</td>
<td>0.625</td>
</tr>
<tr>
<td></td>
<td>Anchor3DLane (Ours)</td>
<td>96.9</td>
<td>94.4</td>
<td>0.094</td>
<td>0.693</td>
<td>0.027</td>
<td>0.579</td>
</tr>
<tr>
<td></td>
<td>Anchor3DLane†(Ours)</td>
<td>95.9</td>
<td>94.4</td>
<td>0.082</td>
<td>0.699</td>
<td>0.030</td>
<td>0.580</td>
</tr>
<tr>
<td>Visual Variations</td>
<td>3DLaneNet [7]</td>
<td>74.9</td>
<td>72.5</td>
<td>0.115</td>
<td>0.601</td>
<td>0.032</td>
<td>0.230</td>
</tr>
<tr>
<td></td>
<td>Gen-LaneNet [8]</td>
<td>87.2</td>
<td>85.3</td>
<td>0.074</td>
<td>0.538</td>
<td>0.015</td>
<td>0.232</td>
</tr>
<tr>
<td></td>
<td>CLGo [20]</td>
<td>89.2</td>
<td>87.3</td>
<td>0.084</td>
<td>0.464</td>
<td>0.045</td>
<td>0.312</td>
</tr>
<tr>
<td></td>
<td>PersFormer [5]</td>
<td>-</td>
<td>89.6</td>
<td>0.074</td>
<td>0.430</td>
<td>0.015</td>
<td>0.266</td>
</tr>
<tr>
<td></td>
<td>GP [16]</td>
<td>92.1</td>
<td>89.9</td>
<td>0.060</td>
<td>0.446</td>
<td>0.011</td>
<td>0.235</td>
</tr>
<tr>
<td></td>
<td>Anchor3DLane (Ours)</td>
<td>93.6</td>
<td>91.4</td>
<td>0.068</td>
<td>0.367</td>
<td>0.020</td>
<td>0.232</td>
</tr>
<tr>
<td></td>
<td>Anchor3DLane†(Ours)</td>
<td>92.5</td>
<td>91.8</td>
<td>0.047</td>
<td>0.327</td>
<td>0.019</td>
<td>0.219</td>
</tr>
</tbody>
</table>

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 error 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 $\phi \in \{0^\circ, \pm 1^\circ, \pm 3^\circ, \pm 5^\circ, \pm 7^\circ, \pm 10^\circ, \pm 15^\circ, \pm 20^\circ, \pm 30^\circ\}$ and pitches $\theta \in \{0^\circ, \pm 1^\circ, \pm 2^\circ, \pm 5^\circ\}$ 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 $F$ is $45 \times 60 \times 64$. During training, $\lambda_{cls}$ and $\lambda_{reg}$ 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 $1/4$ 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-
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.

<table>
<thead>
<tr>
<th>Method</th>
<th>F1(%) †</th>
<th>Cate Acc(%) †</th>
<th>x err/C(m) ↓</th>
<th>x err/F(m) ↓</th>
<th>z err/C(m) ↓</th>
<th>z err/F(m) ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D-LaneNet [7]</td>
<td>44.1</td>
<td>-</td>
<td>0.479</td>
<td>0.572</td>
<td>0.567</td>
<td>0.443</td>
</tr>
<tr>
<td>GenLaneNet [8]</td>
<td>32.3</td>
<td>-</td>
<td>0.591</td>
<td>0.684</td>
<td>0.411</td>
<td>0.521</td>
</tr>
<tr>
<td>PersFormer [5]</td>
<td>50.5</td>
<td>92.3</td>
<td>0.485</td>
<td>0.553</td>
<td>0.364</td>
<td>0.431</td>
</tr>
<tr>
<td>Anchor3DLane (Ours)</td>
<td>53.1</td>
<td>90.0</td>
<td>0.300</td>
<td>0.311</td>
<td>0.103</td>
<td>0.139</td>
</tr>
<tr>
<td>Anchor3DLane † (Ours)</td>
<td>53.7</td>
<td>90.9</td>
<td>0.276</td>
<td>0.311</td>
<td>0.107</td>
<td>0.138</td>
</tr>
<tr>
<td>Anchor3DLane-T † (Ours)</td>
<td>54.3</td>
<td>90.7</td>
<td>0.275</td>
<td>0.310</td>
<td>0.105</td>
<td>0.135</td>
</tr>
</tbody>
</table>

Table 3. Comparison with state-of-the-art methods on ONCE-3DLanes dataset. Results under τCD = 0.3 are displayed here. † denotes iterative regression. “P” and “R” are short for precision and recall respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>F1(%) †</th>
<th>P(%) †</th>
<th>R(%) †</th>
<th>CD Error(m) ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>3D-LaneNet [7]</td>
<td>44.73</td>
<td>61.46</td>
<td>35.16</td>
<td>0.127</td>
</tr>
<tr>
<td>Gen-LaneNet [8]</td>
<td>45.59</td>
<td>63.95</td>
<td>35.42</td>
<td>0.121</td>
</tr>
<tr>
<td>SALAD [36]</td>
<td>64.07</td>
<td>75.90</td>
<td>55.42</td>
<td>0.098</td>
</tr>
<tr>
<td>PersFormer [5]</td>
<td>74.33</td>
<td>80.30</td>
<td>69.18</td>
<td>0.074</td>
</tr>
<tr>
<td>Anchor3DLane (Ours)</td>
<td>74.44</td>
<td>80.50</td>
<td>69.23</td>
<td>0.064</td>
</tr>
<tr>
<td>Anchor3DLane † (Ours)</td>
<td>74.87</td>
<td>80.85</td>
<td>69.71</td>
<td>0.060</td>
</tr>
</tbody>
</table>

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.

advanced 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 PV 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.

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.

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

<table>
<thead>
<tr>
<th>Input</th>
<th>FEat</th>
<th>F1(%)</th>
<th>x err/C(m)</th>
<th>x err/F(m)</th>
<th>z err/C(m)</th>
<th>z err/F(m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BEV</td>
<td>BEV</td>
<td>47.6</td>
<td>0.466</td>
<td>0.342</td>
<td>0.159</td>
<td>0.152</td>
</tr>
<tr>
<td>FV</td>
<td>BEV</td>
<td>47.6</td>
<td>0.443</td>
<td>0.446</td>
<td>0.118</td>
<td>0.160</td>
</tr>
<tr>
<td>FV</td>
<td>FV</td>
<td>53.1</td>
<td>0.300</td>
<td>0.31</td>
<td>0.103</td>
<td>0.139</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Iter</th>
<th>F1(%)</th>
<th>x err/C(m)</th>
<th>x err/F(m)</th>
<th>z err/C(m)</th>
<th>z err/F(m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>54.8</td>
<td>0.318</td>
<td>0.349</td>
<td>0.101</td>
<td>0.147</td>
</tr>
<tr>
<td>2</td>
<td>56.3</td>
<td>0.287</td>
<td>0.335</td>
<td>0.103</td>
<td>0.152</td>
</tr>
<tr>
<td>3</td>
<td>57.0</td>
<td>0.287</td>
<td>0.327</td>
<td>0.104</td>
<td>0.148</td>
</tr>
</tbody>
</table>

Table 7. Ablation study on temporal integration methods.

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

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

[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.
[34] Yue Wang, Eam Khwang Teoh, and Dinggang Shen. Lane detection and tracking using B-Snake. IVC. 2004.
[36] Fan Yan, Ming Nie, Xinyue Cai, Jianhua Han, Hang Xu, Zhen Yang, Chaoqiang Ye, Yanwei Fu, Michael Bi Mi, and

[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
