Sparse Point Guided 3D Lane Detection

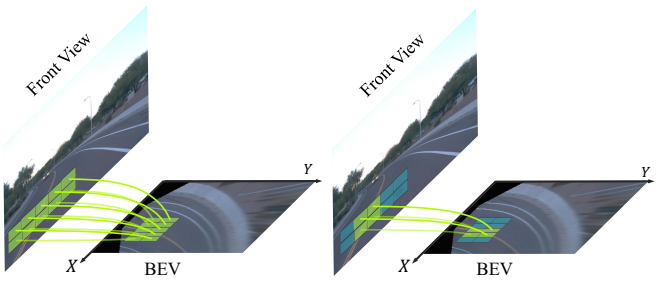
Chengtang Yao[1,3], Lidong Yu[3], Yuwei Wu[1]*, Yunde Jia[2]
[1]Beijing Key Laboratory of Intelligent Information Technology,
School of Computer Science & Technology, Beijing Institute of Technology, China
[2]Guangdong Laboratory of Machine Perception and Intelligent Computing,
Shenzhen MSU-BIT University, China
[3]Autonomous Driving Algorithm, DeepRoute
yao.c.t@bit.edu.cn, yvlidong@gmail.com, jiayunde@bit.edu.cn, wuyuwei@bit.edu.cn

Abstract

3D lane detection usually builds a dense correspondence between the front-view space and the BEV space to estimate lane points in the 3D space. 3D lanes only occupy a small ratio of the dense correspondence, while most correspondence belongs to the redundant background. This sparsity phenomenon bottlenecks valuable computation and raises the computation cost of building a high-resolution correspondence for accurate results. In this paper, we propose a sparse point-guided 3D lane detection, focusing on points related to 3D lanes. Our method runs in a coarse-to-fine manner, including coarse-level lane detection and iterative fine-level sparse point refinements. In coarse-level lane detection, we build a dense but efficient correspondence between the front view and BEV space at a very low resolution to compute coarse lanes. Then in fine-level sparse point refinement, we sample sparse points around coarse lanes to extract local features from the high-resolution front-view feature map. The high-resolution local information brought by sparse points refines 3D lanes in the BEV space hierarchically from low resolution to high resolution. The sparse point guides a more effective information flow and greatly promotes the SOTA result by 3 points on the overall F1-score and 6 points on several hard situations while reducing almost half memory cost and speeding up 2 times.

1. Introduction

3D lane detection is an indispensable part of the advanced driver assistance system, supporting functionalities such as automated lane centering and lane departure warning. It relies on building a dense correspondence between the front view space and BEV space to localize lane points in 3D space. Previous methods build a dense correspondence by directly projecting the 2D image or 2D feature map into 3D space using Inverse Perspective Mapping (IPM) [18, 38], but this kind of correspondence is hard to handle the complex road conditions (e.g., uphill and downhill). Recently, some methods [3, 21] build a dense correspondence using a learnable transformer to resolve this problem. The transformer-based correspondence builds a more effective information flow from the front view space to the BEV space, which results in better performance in extreme scenes.



(a) Dense Correspondence (b) Sparse Correspondence

Figure 1: The visualization of dense correspondence (a) and sparse correspondence (b) between font view and BEV. The green grids represent areas near lanes, while blue grids are redundant areas far from lanes.

Building such dense correspondence is actually redundant for 3D lane detection. As shown in Figure 1, even among a local window near the lane, only $2/5$ correspondences are beneficial for 3D lane detection, while the others are redundant. Among the dense correspondence, 3D lanes only occupy a small ratio, while most correspondences are built for the background. This redundant problem becomes more severe at high resolution, where both the high-resolution front-view and BEV feature maps are necessary for high-quality lane detection in most methods [6, 7, 18, 10, 38, 15]. To this end, we propose to construct a sparse correspondence only focusing on a limited num-

*Corresponding author.

ber of points around 3D lanes. The sparse correspondence builds an effective yet efficient information flow from the front-view space to the BEV space. It directly brings the high-resolution lane information from the front-view space, ensuring fine-grained details in high-resolution BEV space. Without redundant correspondence, the information flow learns to focus more on features beneficial for 3D lane results. Meanwhile, it has a naturally efficient performance by only allocating computation to points related to lanes.

In this paper, we present a sparse-point-guided 3D lane detection method that decomposes the 3D lane detection into coarse-level lane detection and fine-level sparse point refinement. In coarse-level lane detection, we extract multi-scale feature maps from the front-view image. An efficient but dense BEV feature map is built from the front-view feature map at the lowest resolution to detect coarse 3D lanes. Then the fine-level sparse point refinement refines coarse results hierarchically from low resolution to high resolution. In each refinement, we first sample sparse points around coarse 3D lanes within a specific window. And then, we project sampled points onto the front-view plane to extract local features from the high-resolution front-view feature map. The local feature provides fine-grained information to refine the local structure of 3D lanes. At the same time, we compress the front-view feature map into a single vector as the global features to ensure the global smoothness of 3D lanes. We fuse the global feature, the local features, and the sparse point coordinates to predict the location and category of each lane. The fusion of the local and global features refers to the point coordinates, which ensures both the global smoothness and local discrimination of lanes.

We validate our method on two anchor-based approaches and a widely used segmentation-based approach. The demonstrations are conducted on two real-world datasets (OpenLane [3] and ONCE [18]), including different weathers, lane structures, and road conditions. Our method outperforms the two anchor-based approaches, showing our feasibility to be seamlessly integrated with different prototypical methods to offer consistent improvement. Besides, our method simply outperforms the SOTA anchor-based approach by 2 points on the overall F1-score and more than 6 points on the F1-score in several extreme conditions while reducing half memory cost and speeding up 2 times. As for the segmentation-based approach, our method achieves comparable performance and reduces the memory cost of the 3D lane head by 80% and speeds up 2 times.

2. Related Work

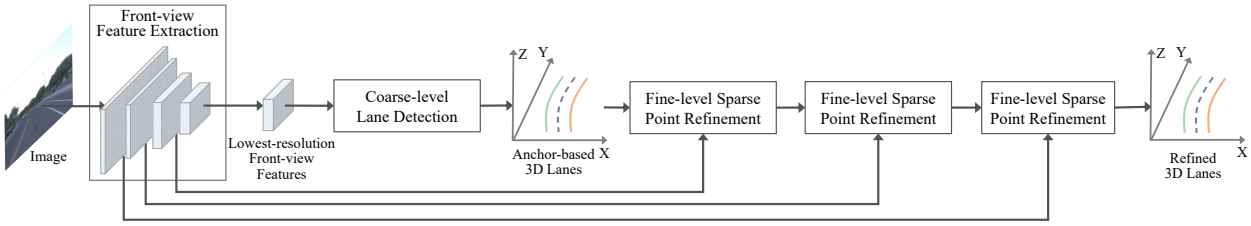
2.1. Lane Detection

According to the lane representation, there are mainly three kinds of approaches for lanes detection, including segmentation-based [14, 26, 37, 42], anchor-based [33, 43, 17, 16, 29], and parameter-based methods[5, 35, 32, 19]. In order to estimate the lane representation in the 3D space, some methods first conduct the estimation in the front-view space and then use inverse perspective mapping (IPM) to bridge the front-view space to BEV space for 3D results. The IPM correspondence is based on the planar road assumption, which is hard to handle road undulates, like a hilly road. To tackle this issue, some methods estimate the 3D lanes in the BEV space to resolve complex road conditions [6, 7, 18, 10, 38, 15, 3, 1, 9, 21]. They usually build a dense correspondence by transforming the front-view feature to the BEV feature map. The dense correspondence builds an effective information flow for the BEV feature to improve the quality of the 3D lane.

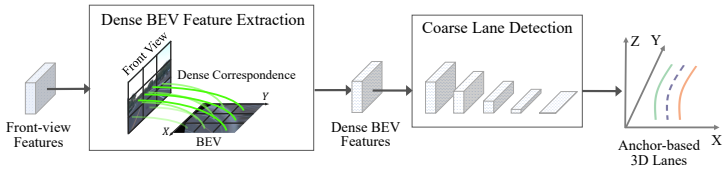
To enhance the BEV feature, some methods introduce auxiliary tasks for the joint learning[6, 7, 18, 38, 15]. The others focus on designing a more effective transformation [10, 3, 15]. RobustLane [10] uses an attention mechanism to aggregate better global information and thus provides a better global smoothness. RTVLane [15] introduces a geometry consistency between 2D and 3D space to guide the learning of BEV features. Recently, PersFormer [3] uses a transformer to build the dense correspondence between the multi-scale front-view features and BEV feature through cross-attention, which achieves SOTA performance. Building a proper correspondence between the front-view space and the BEV space is the key to improving the 3D lane performance, but a dense correspondence is actually redundant. We observe that lanes only occupy a very small ratio of the dense correspondence. Thus, we build a sparse correspondence that only focuses on the points mostly belonging to a lane. The sparse correspondence efficiently brings the high-resolution lane information from the front-view space, ensuring fine-grained details in high-resolution BEV space. Besides, it also guides the information flow to learn better features beneficial for 3D lane results.

2.2. Coarse-to-Fine Methods

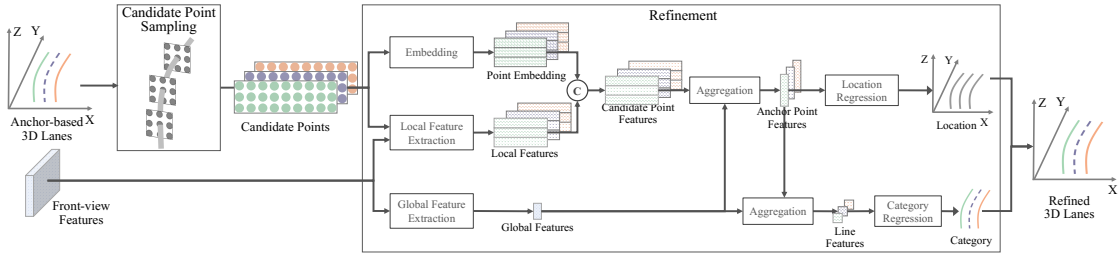
The coarse-to-fine design is widely adopted in many methods to hierarchically improve results [43, 42, 13, 39, 30, 24, 28, 40, 11, 22, 2, 36, 25]. The general design consists of a coarse result estimation using low-frequency information and a hierarchical refinement using high-frequency information at high resolution. Due to the sparsity of high-frequency information, many methods [13, 39, 30, 24] propose to only focus on the valuable regions and operate on sparse point sets rather than regular grids to keep the balance between efficiency and accuracy. They mainly focus on common objects and use local features to recover fine-grained structures, like corners and boundaries. Different from common objects, 3D lanes have a uniquely long and thin structure, requiring a globally smooth and locally accurate estimation. In this paper, we fuse the local and global



(a) The overall pipeline of our model.



(b) The coarse-level lane detection.



(c) The fine-level sparse point refinement. ⓒ represents concatenation.

Figure 2: The overall structure of our model is shown in (a). We decompose the 3D lane detection into the coarse-level lane detection shown in (b) and the fine-level sparse point refinement shown in (c).

information referring to the coordinates of sparse points and jointly refine the global and local structures of lanes.

3. Method

As 3D lanes only occupy a small ratio among the dense correspondence between the front-view space and BEV space, we propose only focusing on sparse points related to lanes in the high-resolution space. As shown in Figure 2a, we decompose the 3D lane detection into a coarse-level lane detection in the lowest resolution and a series of fine-level sparse point refinement. We extract multi-scale front-view features from a front-view image. The lowest-resolution features are fed into coarse-level lane detection to acquire a rough structure of 3D lanes. The other high-resolution features are used to refine the structure of 3D lanes hierarchically from low resolution to high resolution in fine-level sparse point refinement. As shown in Figure 2b, in coarse-level lane detection, we first build a dense but efficient correspondence between the low-resolution front-view feature and the BEV feature. The low-resolution BEV feature is extracted to estimate the anchor representation of 3D lanes [*]. As for the fine-level sparse point refinement shown in Figure 2c, candidate points are first sampled around the anchor points of a lane. The candidate point coordinates are then used as indexes to fetch local features from front-view features. We also compress the front-view features into a single vector as global features, which are fused with local features to refine the location and category of lanes.

3.1. Lane Representation

Given an image, 3D lane detection aims to detect the location l and category p of lanes $L = \{l_i, p_i\}_{i=0}^{i=N-1}$ where N is the number of lanes. The location of lane l_i is represented by an ordered point sequence:

$$l_i = \{(x_j^i, y_j^i, z_j^i)\}_{j=0}^{j=M_i-1}. \quad (1)$$

M_i is the number of points.

In anchor-based representation [3, 6, 7], 3D lanes are initialized by a set of anchor lanes with predefined x-coordinates $\{\bar{x}^i\}_{i=0}^{i=N-1}$, where N is the number of lanes. The anchor lanes are further defined by a series of anchor points with predefined y-coordinates $\{\bar{y}_j\}_{j=0}^{j=M}$, where M is the anchor point count for each anchor lane. In the anchor-based representation, the location of the 3D lane is formu-

[*]We select the anchor-based approach, i.e., Persformer[3], as the baseline in the main paper. The realization of sparse-point guided lane detection for the segmentation-based approach is shown in the Supplementary Material with modification on Persformer backbone.

lated as

$$l_i^{'} = \{(\Delta_{x_j^i}, z_j^i, v_j^i)\}_{j=0}^{j=M}, \tag{2}$$

where $\Delta_{x_j^i}$ is the offset along the x-axis, z_j^i is the absolute height along the z-axis and v_j^i is the visibility for each anchor point. The anchor-based representation can be easily converted into the ordered point sequence representation by

$$l_i = \{(\bar{x}^i + \Delta_{x_j^i}, \bar{y}_j, z_j^i\}_{j=0}^{j=M-1}. \tag{3}$$

3.2. Coarse-level Lane Detection

In coarse-level lane detection, we aim to efficiently acquire the coarse location and category of lanes. We first extract the multi-scale features from the front-view image. Then we transform the feature map from the front view to the BEV at the lowest resolution and compute the coarse location and category of 3D lanes, as shown in Figure 2b. Specifically, we build a dense correspondence between the BEV and front view space at the lowest resolution. We then use the front-view points as the index to fetch front-view features for BEV feature extraction. The details of transformation follow the realization of the Persformer [3] and are discussed in Section 4.2. With the dense BEV feature map, we use a standard anchor-based approach [3, 6, 7] to estimate the anchor-based lane representation.

3.3. Fine-level Sparse Point Refinement

As aforementioned, 3D lanes occupy very little 3D space, which means a large number of 3D points are redundant. Based on the observation, we design a fine-level sparse point refinement, which only deals with sparse points related to lanes instead of the entire BEV space to improve the coarse results. The refinement is carried out hierarchically from the low resolution to the high resolution. We take coarse 3D lanes estimated from the previous lower resolution as reference lanes and sample candidate sparse points around them, as shown in Figure 2c. The sampled sparse points are then used as indexes to extract the high-resolution front-view features. The fine-grained details from the high-resolution space gradually refine the local structure of coarse results.

3.3.1 Candidate Point Sampling

Intuitively, points can be sampled along all three dimensions, i.e., x, y, z, but the number of candidate points will become too large to take. In the anchor-based representation, anchor points are predefined and have fixed values on the y-axis. Thus, we sample points only around the anchor points and only on the x-z plane, as shown in Figure 2c. Specifically, we deem coarse points computed from the previous resolution as reference points. The candidate points are uniformly sampled from the neighbors of reference points within a window $\mathcal{W}$, which has a predefined size and sampling step. The above sampling process can be indicated as

$$S(x_j^i, y_j^i, z_j^i) = \{(x_t, y_j^i, z_t) \mid (x_t, z_t) \in \mathcal{W}_{(x_j^i, z_j^i)}\}. \tag{4}$$

It is worth noting that the sampling process has a different requirement for training and inference. During the training stage, we require as much information as possible to supervise the learning of anchors. So, we keep all of the candidate points. In contrast, in the inference stage, we need to reduce as many redundant anchor lanes as possible to save time and memory costs. Thus, we filter redundant anchor lanes in the inference stage before sampling. The filtering is designed by three rules. First, the detected categories of lanes are not backgrounds. Second, there are at least two visible points on a lane. Third, the distance of different lanes is far enough, where we define the distance between two lines as the sum of the distance between anchor points with the same $\bar{y}_j$.

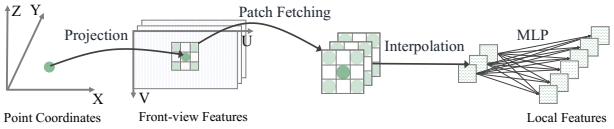


Figure 3: The pipeline of local feature extraction. We project the 3D point (x, y, z) onto image plane (u, v). A local patch around the point (u, v) is fetched to compute local features using interpolation and multi-layer perception (MLP).

3.3.2 Refinement

Anchor-based lane representation consists of two kinds of properties, including the location of each anchor point and the category of each anchor lane.

Point Location Refinement Lane has a globally smooth and locally complex structure due to its long and thin shape. To this end, we propose to use both local features of candidate points and global features of the whole image to refine the coarse results. As shown in Figure 3, we project candidate points (x, y, z) onto image plane (u, v) for local feature extraction. This process can be written as

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \frac{1}{z} \cdot K \cdot E \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}, \tag{5}$$

where K is intrinsic matrix, E is extrinsic matrix. As the projected point coordinate is a real value and the feature map only consists of regular grids, we use a local area around the projected point (u, v) to approximate the coordinates. We then use the local area to compute local features F_l through bilinear interpolation and multi-layer perception

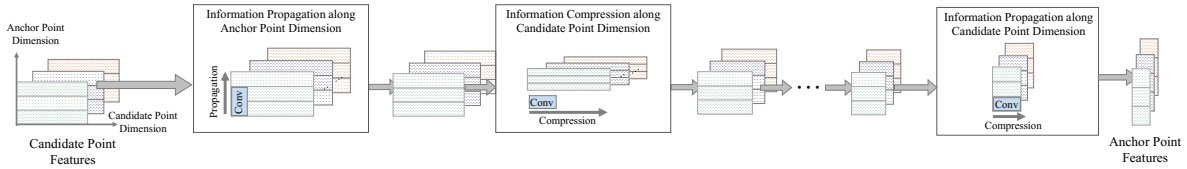


Figure 4: The pipeline of aggregation. Without loss of generality, we take the aggregation on candidate point features as an example to visualize the pipeline of aggregation.

(MLP). The extracted local feature for sparse points is indicated as $F_l \in \mathbb{R}^{C\times N\times M\times ||S||}$, where C is the feature size, N is the number of anchor lanes, M is the number of anchor points and $||S||$ is the sampling size. To guide the fusion of local and global information, we also encode coordinates of sampled sparse points into a high-dimension feature $F_p \in \mathbb{R}^{C\times N\times M\times ||S||}$ through a single MLP.

The point embedding F_p and local features F_l are then concatenated as candidate point features $F_c \in \mathbb{R}^{C\times N\times M\times ||S||}$. In order to compute anchor point features $F_a \in \mathbb{R}^{C\times N\times M}$ from F_c, we propose to aggregate candidate point features belonging to the same anchor point. As shown in Figure 4, we gradually compress the candidate point dimension from $||S||$ to 1. During the compression, we use convolution with the kernel size of 3×1 and padding of $(1, 0)$ to propagate context information along the anchor point dimension. This operation constrains the context information to only flow among the anchor points in the same lane. Meanwhile, we use convolution with the kernel size of 1×3 and padding of $(0, 0)$ to compress local information along the candidate point dimension. This operation fuses the local information from the sampled sparse points to the corresponding anchor. These two operations are alternately repeated to propagate and compress information progressively.

As for the global feature, we compress the entire front-view features into a single feature vector $F_g \in \mathbb{R}^C$ through pooling operation, following previous methods [24, 27, 4, 20, 8, 41]. We then concatenate the global feature vector F_g to each anchor feature vector F_a and fuse them together to balance the global and local information. The fused features are then used to update the anchor point location $\{(\Delta_{x_j^i}, z_j^i, v_j^i)\}_{j=0}^{j=M}$ through 3 convolution layers.

Lane Category Refinement In order to detect the lane categories, we extract lane features by aggregating the feature of anchor points belonging to the same lane. Similar to the above aggregation operation for the anchor point features, we first concatenate the global feature F_g with anchor features F_a and then gradually propagate the information along the anchor point dimension. The aggregated lane features $F_l \in \mathbb{R}^{C\times N}$ are then used to update the lane category p_i through several convolution layers.

3.4. Loss

The predicted anchor-based lane representation consists of category $\tilde{p_i}$ and location $\tilde{l_i} = \{(\tilde{\Delta}_{x_j^i}, \tilde{z}_j^i, \tilde{v}_j^i)\}_{j=0}^{j=M}$. We use the cross entropy to supervise the predicted category $\tilde{p}$:

$$\mathcal{L}_p = -\frac{1}{N}\sum_{i=0}^{i=N-1} p_i log(\tilde{p}_i). \tag{6}$$

As for the location, we supervise the learning of visibility $\tilde{v}$ by l_1 loss and use the visibility as weights to compute the l_1 loss for $\tilde{\Delta}_x$ and $\tilde{z}$:

$$\mathcal{L}_\Delta = \frac{1}{M\cdot N}\sum_{i=0}^{i=N-1}\sum_{j=0}^{j=M-1} v_j^i||\Delta_{x_j^i} - \tilde{\Delta}_{x_j^i}||_1, \tag{7}$$

$$\mathcal{L}_z = \frac{1}{M\cdot N}\sum_{i=0}^{i=N-1}\sum_{j=0}^{j=M-1} v_j^i||z_j^i - \tilde{z}_j^i||_1, \tag{8}$$

$$\mathcal{L}_v = \frac{1}{M\cdot N}\sum_{i=0}^{i=N-1}\sum_{j=0}^{j=M-1} ||v_j^i - \tilde{v}_j^i||_1. \tag{9}$$

The joint loss can be indicated as

$$\mathcal{L} = \mathcal{L}_p + \mathcal{L}_\Delta + \mathcal{L}_z + \mathcal{L}_v. \tag{10}$$

During the hierarchical refinement from the low resolution to the high resolution, we separately compute the above loss for each resolution and sum up the losses as the final loss to train the network.

4. Experiments

In the main paper, we present the sparse-point guided 3D lane detection method on the anchor-based approach. We mainly use Persformer [3] as the baseline, while Gen-LaneNet [7] is taken as an optional baseline to verify the feasibility of our method. We validate our method on the real-world datasets OpenLane [3] and ONCE-3DLanes [18] to demonstrate our priority to handle complex environments and variable scenes. As for our method on the segmentation-based approach, we present the results obtained on the synthetic dataset Apollo 3D Lane [7] in the Supplementary Material. The code is available at `https://github.com/YaoChengTang/Sparse-Point-Guided-3D-Lane-Detection`

Method	All ↑	Up & Down ↑	Curve ↑	Extreme Weather ↑	Night ↑	Intersection ↑	Merge & Split ↑
3D-LaneNe [6]	44.1	40.8	46.5	47.5	41.5	32.1	41.7
Gen-LaneNet [7]	32.3	25.4	33.5	28.1	18.7	21.4	31
PersFormer [3]	50.5	42.4	55.6	48.6	46.6	40	50.7
Ours + PersFormer	**53.7 (3.2↑)**	**46.2 (3.8↑)**	**59.2 (3.6↑)**	**54.8 (6.2↑)**	**49.8 (3.2↑)**	**41.9 (1.9↑)**	**52.1 (1.4↑)**

Table 1: Performance comparison with state-of-the-art methods on OpenLane benchmark in different scenarios, where the evaluation metric is F1-score. ↑ means that the larger the value, the better the result.

Method	F-score (%) ↑	X error near (m) ↓	X error far (m) ↓	Z error near (m) ↓	Z error far (m) ↓
3D-LaneNe [6]	44.1	0.479	0.572	0.367	0.443
Gen-LaneNet [7]	32.3	0.591	0.684	0.411	0.521
Cond-IPM [3]	36.6	0.563	1.080	0.421	0.892
PersFormer [3]	50.5	0.485	0.553	**0.364**	0.431
Gen-LaneNet* [7]	42.8	0.488	0.632	0.374	0.481
Ours + Gen-LaneNet*	46.6	0.475	0.577	0.371	0.445
Ours + PersFormer	**52.3**	**0.468**	**0.514**	0.371	**0.418**

Table 2: Performance comparison with state-of-the-art methods using different evaluation metrics on the validation set of OpenLane benchmark. * means that we re-implement the method on the OpenLane benchmark. ↑ means that the larger the value, the better the result. ↓ means that the smaller the value, the better the result.

4.1. Datasets

OpneLane Dataset OpneLane [3] is a challenging real-world 3D lane dataset constructed on Waymo Open dataset [31]. It consists of 200K frames with 14 kinds of categories, complex lane structures, and five kinds of weather. 50% frames have an altitude change of more than $1m$, and 25% frames have more than six lanes. Currently, it is the most challenging lane detection dataset, so we choose it as the main dataset to conduct ablation studies and demonstrate the performance promotion of our method.

ONCE-3DLanes Dataset ONCE-3DLanes [18] is a real-world 3D lane dataset constructed on ONCE [23]. It consists of various scenes, such as highways, bridges, tunnels, suburbs, and downtown, with different weather conditions (sunny/rainy) and lighting conditions (day/night).

4.2. Implementation Details

We select the PersFormer [3] as the baseline of the anchor-based approach to leverage the sparse point. EfficientNet [34] is used as the backbone for the front-view feature extraction. Perspective Transformer [3] and 3D-GeoNet [7] are used in coarse-level lane detection to extract the dense BEV features and detect coarse 3D lanes at the lowest resolution. Following PersFormer, we set the highest resolution of BEV space as 208×128 with the range of $[-10, 10]$ meters along the x-axis and $[3, 101]$ meters along the y-axis, respectively. We set the count of anchor lanes as $N = 182$ and anchor points for each lane as $M = 10$, where the fixed $\{y^i\}$ is set as $\{5, 10, 15, 20, 30, 40, 50, 60, 80, 100\}$ for each lane. In fine-level sparse point refinement, we set the size of sampling window $\mathcal{W}$ as 3×3 with $(1, 0.5)$ sampling step along the x-axis and the z-axis separately.

Method	Training Memory (G)	Inference Speed (FPS)
3D-LaneNe [6]	6.6	103
Gen-LaneNet [7]	4.7	67
PersFormer [3]	19.6	27
Ours + PersFormer	12.5	45

Table 3: Training memory cost and inference speed comparison with state-of-the-art approaches. We acquire the memory cost on a single A30 GPU with a batch size of 8. We obtain the inference speed on a single A30 GPU with a batch size of 1.

In the training stage, we pre-train the coarse-level lane detection for 100 epochs and then finetune our fine-level sparse point refinement for another 100 epochs. For both pertaining and finetuning, we use Adam optimizer [12] with a base learning rate of 2×10^{-4} and a weight decay of 10^{-3}. The cosine learning rate policy is applied to optimization with a maximum iteration of 8 and a minimum learning rate of 10^{-5}. We also use gradient norm clipping during the optimization with a maximum norm weight of 35. In the Inference stage, we use the number of valid anchor points in each lane as the threshold to filter anchor lanes that are too close. For more details about training and inference, please refer to the code and our supplemental materials.

4.3. Evaluation Metrics

Following OpenLane, we use bipartite matching to evaluate the predicted and ground-truth lanes. The matching is true positive when the distance error of 75% lane points is less than $1.5m$. The percentage of matched ground-truth lanes is deemed as the recall, and the percentage of matched predicted lanes is deemed as the precision. The average of recall and precision is F-score. The lane points are also divided into near points $(3-40m)$ and far points $(40-101m)$.

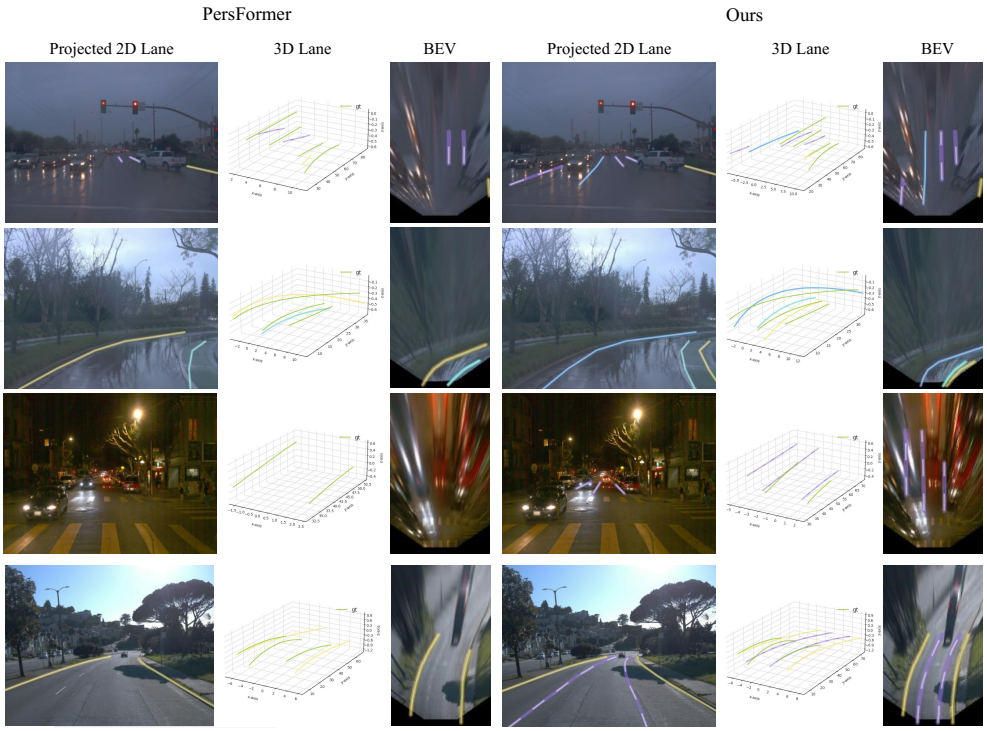


Figure 5: The visualization results of PersFormer and our method on the OpneLane dataset.

The point-wise distance error along the x-axis and z-axis is used to evaluate the location accuracy for near and far points. In ONCE, we also use the unilateral Chamfer Distance (CD) metric that computes the chamfer distance between the predicted and ground truth lane.

4.4. Benchmark Performance

OpenLane Following the setting in OpenLane, we evaluate the performance in different scenarios. As shown in Table 1, we achieve the best results in all scenarios. It is worth noting that our method achieves more than 6 points promotion in extreme weather scenarios. This promotion shows that our method greatly resists the feature noise and extracts valuable information by only focusing on sparse points around 3D lanes. In Table 2, we further compare our method with state-of-the-art approaches using different evaluation metrics. Our method outperforms all state-of-the-art approaches in almost every evaluation metric. Specially, we improve the location on the x-axis in both near and far distances and thus improve the overall F1-score. What's more, our method effectively helps Gen-LaneNet outperform both the original result of Gen-LaneNet published by OpenLane and our reproduced Gen-LaneNet* on all metrics. These results prove that our method is flexible and can seamlessly integrate with different prototypical methods to offer consistent improvement.

We also compare our method with state-of-the-art approaches from the training memory cost and the inference speed. As shown in Table 3, our method achieves the lowest memory cost and the fasted inference speed. Compared to the baseline, we greatly reduce the memory cost, which enables the front-view images to have higher resolution. Furthermore, the sparse point guides the network to achieve a better result in most cases with a much faster inference speed. It should be noted that the speed and memory cost reported in this paper is computed based on a pure Python realization of sparse-point sampling and aggregation without any unfair engineering optimization like the custom CUDA kernel that significantly reduces the running time.

Besides the quantitive analysis, we also compare visualization results of the PersFormer and our method in different scenarios for qualitative analysis. As shown in Figure 5, our method obviously outperforms the PersFormer in cloudy, rainy, night, and different road conditions. This further demonstrates that the sparse point guides the BEV feature to extract more meaningful information from the front-view image and thus improve the recall and accuracy of the detected lanes.

ONCE-3DLanes As illustrated in Table 4, we follow the PersFormer using the validation set of ONCE-3DLanes to evaluate the results of our methods. Considering scenes are easier in the ONCE benchmark, the quantity results are already very high. The sparse point slightly promotes the baseline in the F1-score, Precision, and CD error with a much more efficient performance, which only costs half the memory and inference time.

Method	F1-score (%) ↑	Precision (%) ↑	Recall (%) ↑	CD error (m) ↓
3D-LaneNe [6]	44.73	61.46	35.16	0.127
Gen-LaneNet [7]	45.59	63.95	35.42	0.121
SALAD [3]	64.07	75.90	55.42	0.098
PersFormer [3]	74.33	80.30	**69.18**	0.074
Ours + PersFormer	**74.79**	**81.85**	68.86	**0.070**

Table 4: Performance comparison with state-of-the-art methods on ONCE benchmark. ↑ means that the larger the value, the better the result. ↓ means that the smaller the value, the better the result.

Front-view Feature Extraction	Coarse level	Fine level 1	Fine level 2	Fine level 3
4.8ms	6.1ms	3.8ms	3.1ms	3.4ms

Table 5: Inference time cost of each step in our method.

Coarse level	Fine level 1	Fine level 2	Fine level 3	F1-score (%)	X error near (m)
✓				51.3	0.484
✓	✓			52.1	0.475
✓	✓	✓		53.6	0.470
✓	✓	✓	✓	53.7	0.468

Table 6: The performance of coarse-level lane detection and fine-level sparse point refinement.

4.5. Ablation Study and Discussion

Complexity Analysis We evaluate the time cost of each step in our method to show the efficiency of our sparse refinement. As shown in Table 5, it is more efficient to use sparse points for feature fetching and 3d lane refinement, even compared to the coarse-level lane detection conducted in the lowest resolution.

Effectiveness of Refinement In order to validate the effectiveness of our fine-level sparse point refinement, we compare results from the coarse level and from the different fine levels. As shown in Table 6, the fine-level sparse point refinement greatly improves the results of coarse-level lane detection in both F1-score and X error near.

Effectiveness of Features To ensure globally smooth and locally discriminative performance, we leverage both the global and local features to refine the lanes. We demonstrate the advantage of balancing the global and local information by comparing three kinds of feature conditions, including only using global features, only using local features, and using both local and global features. As shown in Table 7, the first condition results in a very low F1 score of 46.8%, and the second one is already comparable to the baseline, while the last one fusing with sparse points gives the best result.

Effectiveness of Point Sampling As aforementioned, we sample candidate points around the anchor point on the x-z plane. Here, we analyze the effectiveness of point sampling using different sampling steps.

As illustrated in Table 8, the result gets better when the value of the sampling step is smaller along the z-axis and larger along the x-axis. This phenomenon shows that the local features fetched along the x-axis are more valuable than the features fetched along the z-axis, where the sampling along the x-axis actually gives us more valid sample points on the front-view image.

Global Features	Local Features	F1-score (%)
✓		46.8
	✓	51.8
✓	✓	53.7

Table 7: The performance of our method with/without global features and local features.

x-axis	z-axis	F1-score
1	1	53.61
1	0.5	53.65
0.5	1	53.34
0.5	0.5	53.46
1	0.25	53.67

Table 8: The performance of our method with different sampling steps along x-axis and z-axis.

4.6. Limitations and Discussion

As shown in the above experiments, we have achieved great progress in both accuracy and efficiency. However, our method still has some limitations, e.g., assuming all 3D lane instances could be detected on BEV features in the lowest resolution. This assumption becomes less powerful when meeting extremely complex and crowded road conditions. A slow-fast update strategy for refinement at different scales might give better results, but it requires much more engineering tricks on feature manipulation. In this paper, we mainly focus on proving the priority and possibility of simply using sparse correspondence instead of dense correspondence for 3D lane detection.

5. Conclusion

In this paper, we have proved that using sparse points is more efficient and adequate for high-quality 3D lane detection than building a dense correspondence between the HR front-view space and BEV space. We presented a sparse point-guided 3D lane detection method, including coarse-level lane detection and fine-level sparse point refinement. Experiments proved that our method could achieve much better results with less memory and time cost.

Acknowledgement This work was supported by the Natural Science Foundation of China (NSFC) under Grants No.62176021.

References

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

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5470–5479, 2022.

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

[4] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5932–5941, 2019.

[5] Zhengyang Feng, Shaohua Guo, Xin Tan, Ke Xu, Min Wang, and Lizhuang Ma. Rethinking efficient lane detection via curve modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17041–17049, 2022.

[6] 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 International Conference on Computer Vision (ICCV)*, pages 2921–2930, 2019.

[7] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[8] Jiahui Huang, Shi-Sheng Huang, Haoxuan Song, and Shimin Hu. Di-fusion: Online implicit 3d reconstruction with deep priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8928–8937, 2021.

[9] Shaofei Huang, Zhenwei Shen, Zehao Huang, Zihan Ding, Jiao Dai, Jizhong Han, Naiyan Wang, and Si Liu. Anchor3dlane: Learning to regress 3d anchors for monocular 3d lane detection. *arXiv preprint arXiv:2301.02371*, 2023.

[10] Yujie Jin, Xiangxuan Ren, Fengxiang Chen, and Weidong Zhang. Robust monocular 3d lane detection with dual attention. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, pages 3348–3352, 2021.

[11] Longlong Jing, Yucheng Chen, and Yingli Tian. Coarse-to-fine semantic segmentation from image-level labels. *IEEE Transactions on Image Processing (TIP)*, 29:225–236, 2019.

[12] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[13] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 9799–9808, 2020.

[14] Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Taeyeop 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 1965–1973, 2017.

[15] Chenguang Li, Jia Shi, Ya Wang, and Guangliang Cheng. Reconstruct from top view: A 3d lane detection approach based on geometry structure prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4369–4378, 2022.

[16] 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.

[17] 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 International Conference on Computer Vision (ICCV)*, pages 3773–3782, 2021.

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

[19] Ruijin Liu, Zejian Yuan, Tie Liu, and Zhiliang Xiong. End-to-end lane shape prediction with transformers. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 3694–3702, 2021.

[20] Shilin Liu, Haoxiang Guo, Hao Pan, Peng-Shuai Wang, Xin Tong, and Yang Liu. Deep implicit moving least-squares functions for 3d reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1788–1797, 2021.

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

[22] Zhe Liu, Xin Zhao, Tengteng Huang, Ruolan Hu, Yu Zhou, and Xiang Bai. Tanet: Robust 3d object detection from point clouds with triple attention. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 11677–11684, 2020.

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

[24] Lars M. Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2019.

[25] Roozbeh Mottaghi, Yu Xiang, and Silvio Savarese. A coarse-to-fine model for 3d pose estimation and sub-category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 418–426, 2015.

[26] 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)*, pages 7276–7283, 2018.

[27] Jeong Joon Park, Peter R. Florence, Julian Straub, Richard A. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 165–174, 2019.

[28] Marco Pedersoli, Andrea Vedaldi, Jordi Gonzalez, and Xavier Roca. A coarse-to-fine approach for fast deformable object detection. *Pattern Recognition (PR)*, 48(5):1844–1853, 2015.

[29] Zequn Qin, Huanyu Wang, and Xi Li. Ultra fast structure-aware deep lane detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 276–291. Springer, 2020.

[30] Shunsuke Saito, Tomas Simon, Jason M. Saragih, and Hanbyul Joo. Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 81–90, 2020.

[31] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2446–2454, 2020.

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

[33] Lucas Tabelini, Rodrigo Berriel, Thiago Meireles Paixão, Claudine Santos Badue, Alberto Ferreira de Souza, and Thiago Oliveira-Santos. Keep your eyes on the lane: Real-time attention-guided lane detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 294–302, 2021.

[34] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 6105–6114. PMLR, 2019.

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

[36] Jun Wang, Shiyi Lan, Mingfei Gao, and Larry S Davis. Infofocus: 3d object detection for autonomous driving with dynamic information modeling. In *European Conference on Computer Vision*, pages 405–420. Springer, 2020.

[37] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 689–704. Springer, 2020.

[38] Fan Yan, Ming-Jun Nie, Xinyue Cai, Jianhua Han, Han Xu, Zhengxin Yang, Chao Ye, Yanwei Fu, Michael B. Mi, and Li Zhang. Once-3dlanes: Building monocular 3d lane detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17122–17131, 2022.

[39] Chengtang Yao, Yunde Jia, Huijun Di, Pengxiang Li, and Yuwei Wu. A decomposition model for stereo matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6091–6100, 2021.

[40] Jian Yao, Marko Boben, Sanja Fidler, and Raquel Urtasun. Real-time coarse-to-fine topologically preserving segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2947–2955, 2015.

[41] Jianglong Ye, Yuntao Chen, Naiyan Wang, and X. Wang. Gifs: Neural implicit function for general shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12819–12829, 2022.

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

[43] Tu Zheng, Yifei Huang, Yang Liu, Wenjian Tang, Zheng Yang, Deng Cai, and Xiaofei He. Clrnet: Cross layer refinement network for lane detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 898–907, 2022.