Detecting Lane and Road Markings at A Distance with Perspective Transformer Layers

Zhuoping Yu[1,2], Xiaozhou Ren[1,2], Yuyao Huang[1,2], Wei Tian[1,2], Junqiao Zhao[2,3,*]

Abstract— Accurate detection of lane and road markings is a task of great importance for intelligent vehicles. In existing approaches, the detection accuracy often degrades with the increasing distance. This is due to the fact that distant lane and road markings occupy a small number of pixels in the image, and scales of lane and road markings are *inconsistent* at various distances and perspectives. The Inverse Perspective Mapping (IPM) can be used to eliminate the perspective distortion, but the inherent interpolation can lead to artifacts especially around distant lane and road markings and thus has a negative impact on the accuracy of lane marking detection and segmentation. To solve this problem, we adopt the Encoder-Decoder architecture in Fully Convolutional Networks and leverage the idea of Spatial Transformer Networks to introduce a novel semantic segmentation neural network. This approach decomposes the IPM process into multiple consecutive differentiable homographic transform layers, which are called "Perspective Transformer Layers". Furthermore, the interpolated feature map is refined by subsequent convolutional layers thus reducing the artifacts and improving the accuracy. The effectiveness of the proposed method in lane marking detection is validated on two public datasets: TuSimple and ApolloScape.
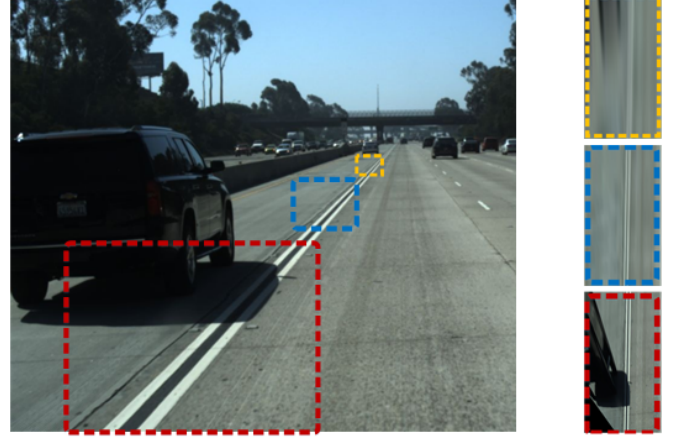
Fig. 1. View of the lane markings at different distances. Similar lane markings in bird's eye view show very different shape and scale features in original view.

I. INTRODUCTION

Lane and road markings are critical elements in traffic scenes. The lane lines or road signs such as arrows can provide valuable information for planning the vehicle trajectory or controlling its driving behavior. Thus, accurate lane marking detection is of great importance for self-driving cars.

Current lane marking detection methods mostly utilize the segmentation technique [1] [2] [3], which is based on fully convolutional deep neural networks (FCNs). The segmentation networks rely on local features which are extracted from the raw RGB pattern and mapped into semantic spaces for pixel-level classification. However, such an architecture often suffers from an accuracy degradation for lane and road markings far away from the ego-vehicle, because distant lane and road markings occupy a small number of pixels in the image, and their features become inconsistent for varying distances and perspectives, as shown in Fig. 1. This also imposes a negative effect on the performance of autonomous driving, since both distant and close lane marking information is important for the controlling and planning tasks.

An intuitive solution to the above problem is to transform the original image to a bird's-eye view (BEV) using the Inverse Perspective Mapping (IPM). In principle, this can get rid of the perspective distortion in the image and solve the problem of inconsistent scales of lane and road markings at different distances. However, the IPM is typically implemented by interpolation which seriously reduces the resolution of the distant road surface in the image and create unnatural blurring and stretching (Fig. 2), leading to a negative impact on the final detection accuracy. To tackle this issue, we adopt the Encoder-Decoder architecture in Fully Convolutional Networks [4] and leverage the idea of Spatial Transformer Networks [5] to build a semantic segmentation neural network. As shown in Fig. 3, Fully Convolutional layers are interleaved with a series of differentiable homographic transform layers, which are called "Perspective Transformer Layers" (PTLs) and can transform the multi-channel feature maps from the original view to the bird's-eye view during the encoding process. Afterwards, it transforms feature maps back to the original perspective in the decoding process, where subsequent convolutional layers are employed to refine each interpolated feature map. Therefore, this network can still use the labels in the original view for an end-to-end training.

In this work, our contributions can be summarized as follows:

- We proposed a lane marking detection network based on FCN, which integrates with novel PTLs to reduce the perspective distortion at a distance.
- We build a mathematical model to derive the parameters of consecutive PTLs, enabling the mutual conversion

[1] School of Automotive Studies, Tongji University, Shanghai, China
[2] Institute of Intelligent Vehicles, Tongji University, Shanghai, China
[3] Department of Computer Science and Technology, Tongji University, Shanghai, China
[*] Corresponding author. E-mail: zhaojunqiao@tongji.edu.cn

Fig. 2. Disadvantages of typical IPM. *Left:* the front-facing image. *Right:* the Bird's-eye view created by applying typical IPM, leading to resolution reduction of the distant road surface.

between the original view and the bird's eye view step-wisely.

- The effectiveness of the proposed method in both instance segmentation and semantic segmentation for lane and road markings is approved on two public datasets: TuSimple [6] and ApolloScape [7].

II. RELATED WORKS

Lane marking detection has been intensively explored and recent progresses have mainly focused on the semantic segmentation-based and instance segmentation-based methods.

A. Lane Marking Detection by Semantic Segmentation

The emerging lane and road marking datasets enable the introduction of deep semantic segmentation-based methods into the lane marking detection task [9], [10]. The work [9] proposed both a road marking dataset and a segmentation network using the ResNet with Pyramid Pooling. Lee et al. [1] proposed a unified end-to-end trainable multi-task network that jointly handles lane marking detection and road marking recognition under adverse weather conditions with the guidance by a vanishing point. Zhang et al. [11] proposed a segmentation-by-detection method for road marking extraction, which delivers outstanding performances on cross datasets. In this method, a lightweight network is dedicatedly designed for road marking detection. However, the segmentation is mainly based on conventional image morphological algorithms.

B. Lane Marking Detection by Instance Segmentation

Semantic segmentation is essentially just a pixel-level classification problem, it neither can distinguish different instances within the same category, nor can interpret separated parts of the same marking (dashed lines, zebra lines, *etc.*) as a unity. Therefore, researchers' attention gradually shifted to study the problem of instance segmentation.

Pan et al. [3] proposed the Spatial CNN (SCNN), which generalizes traditional spatial convolutions to slice-wise convolutions within feature maps, thus enabling message passing between pixels across rows and columns in a layer. This is particularly suitable for long continuous shape structure or large objects with strong spatial relationship but less appearance clues, such as traffic lanes. Hsu et al. [12] proposed a novel learning objective function to train the deep neural network to perform an end-to-end image pixel clustering. They applied this approach on instance segmentation, and used the pairwise relationship between pixels for supervision. Neven et al. [2] went beyond the modelling limitation by pre-defined number of lanes, and proposed to cast the lane detection problem as an instance segmentation problem, in which each lane forms its own instance that can be trained end-to-end. To parameterize the segmented lane instances before the lane fitting, they further proposed to apply a learned perspective transform, which is conditioned on the image and called H-Net.

C. Perspective Transform in CNNs

The accuracy of existing lane marking detection methods often degrades at distant markings due to the inconsistent scales caused by perspective projection. To compensate the perspective distortion, a spatial transform, such as IPM, should be involved. A typical work to implement such a transform using neural network is the Spatial Transformer Network [5]. It introduces a learnable module, *i.e.*, the Spatial Transformer, which explicitly allows the spatial manipulation of data within the network. This differentiable module can be applied to existing convolutional architectures, enabling actively spatial transform of feature maps.

The most similar work to ours is [13]. In this work, an adversarial learning approach is proposed for generating an improved IPM using the STN [5] based on a single image. The generated BEV images contain sharper features (*e.g.*, lane and road markings) than that produced by traditional IPM. The main difference between this work and ours is that they took a ground-truth BEV image (obtained by visual odometry) for supervision and trained their network with a GAN loss. Their target is to generate a high-resolution IPM, while ours is to improve the segmentation accuracy. Besides, they apply STN layers at the bottleneck of the encoder-decoder network, while our PTLs are interleaved with the convolutional and downsampling layers, which can utilize subsequent convolutional layers to refine each interpolated feature map.

III. PROPOSED METHOD

In this work, we boost the performance of lane marking detection by inserting differentiable PTLs into the standard encoder-decoder architecture. One challenge in designing Transformer layers lies in dividing and distributing the integral transform into several even steps. Another is about how to determine the proper cropping range for these intermediate views. In this section, we firstly describe the improved backbone in section III-A. Then, we address how to apply Transformer layers as well as how to solve above difficulties in section III-B. Finally, we illustrate the deployment of the backbone in both semantic and instance segmentation
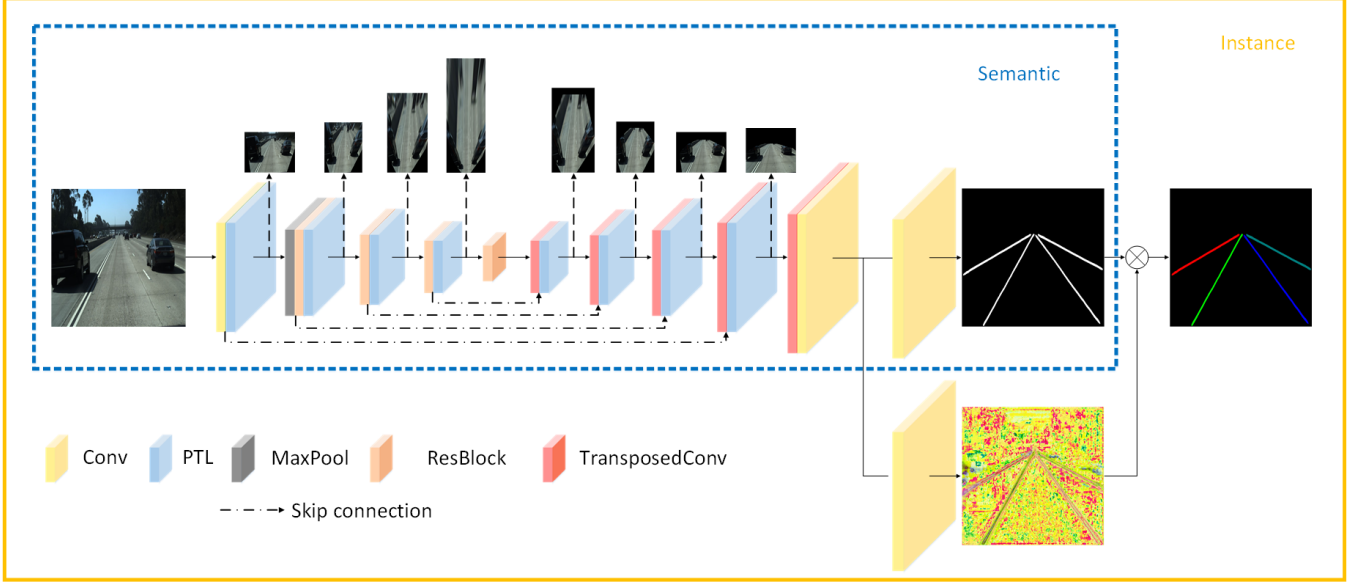
Fig. 3. TPSeg architecture. Semantic segmentation network is the main body of TPSeg. It is based on a standard encoder-decoder network introduced in [4], of which the encoder is implemented with a ResNet-34 network [8]. Each Perspective Transformer Layer (PTL) follows a ResBlock or a Transposed Convolution layer, perspectively and gradually warping feature maps into a bird's-eye view. The process of perspective transform is visualized qualitatively using color images above the main network. As for instance segmentation, following [2], an instance embedding branch is added. It shares previous layers with the semantic segmentation networks and outputs N-dimensional embedding per lane pixel, which is also visualized as a color map. When conducting forward inference, outputs of both branches are merged to get the final instance segmentation results.

context with details about detection heads and loss functions dedicated to these tasks in III-C.

A. Network Structure

As shown in Fig. 3, the overall semantic segmentation network is based on a standard encoder-decoder network [4], in which the encoder is implemented with a ResNet-34 network [8]. In this structure, PTLs interleave with the convolutional and down-sampling layers. We refer our network as TPSeg. In this network, images go through the encoder and are down-sampled to a feature map with 5 times of the stride-2 down-sampling operation. And the feature map is gradually warped into a pseudo BEV. Afterwards, the decoder reverts the previous transforms by up-sampling and back-projecting the feature map into its original size and perspective, while keeping the accumulated high-level semantic information of lane and road markings.

Mathematically, the PTL is nothing different than a linear transformation of the coordinates and a bi-linear sampling of the feature map. And the sampling procedure does not affect the overall differentiability, so that the network with PTL can be trained in an end-to-end manner. For a better understanding, we use a RGB image instead of feature maps and give a qualitative visual description of the yielded results by PTLs in Fig. 4.

Since feature maps in the middle of the convolutional neural network have perspective transform relationships with the input image, this transform is equivalent to warping the input image to a BEV for training and detection, thus solving the problem of inconsistent scales of lane and road markings due
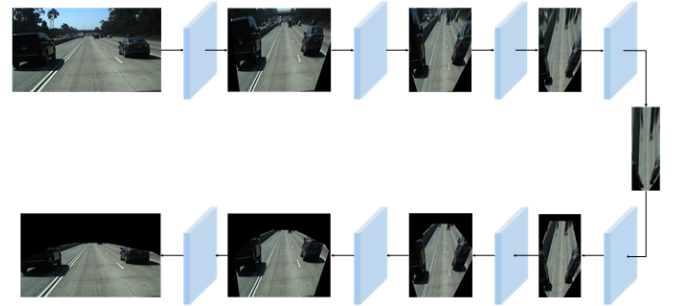

Fig. 4. Demonstrations of PTLs in encording and decoding process. The real PTLs work on high-dimensional feature maps of CNNs.

to different distances. Meanwhile, the subsequent refinement reduces blur and artifacts caused by interpolation. Similar to the FCN, we also add skip-connections to merge feature maps from the up- and down-sampling layers of the same size. This can compensate for the information loss during the down-sampling, resulting in clear boundaries for detected lane and road markings.

B. Consecutive Perspective Mapping

In order to map the front-view image captured by the vehicle-mounted camera into a bird's-eye view smoothly, we adopt an approach differing from the standard IPM method. Here we decompose the integral transform $\mathbf{H}$ into a series of shortest-path consecutive transforms $\{\mathbf{H}_{i,i+1}\}$ ($\mathbf{H}_i$ for short) that project the view i into view $i+1$. This procedure is

interpreted as

$$\mathbf{H} = \prod_{i=0}^{N} \mathbf{H}_i \sim \prod_{i=0}^{N} \left[\mathbf{K}_{i+1} \left(\mathbf{R}_{i,i+1} - \frac{\mathbf{t}_i \mathbf{n}_i^T}{d_i} \right) \mathbf{K}_i^{-1} \right], \quad (1)$$

where $\mathbf{R}_{i,i+1}$ (can be denoted as $\mathbf{R}_i$ for short) is the rotation matrix by which virtual camera $i+1$ is rotated in relation to virtual camera i; $\mathbf{t}_i$ is the translation vector from i to $i+1$; $\mathbf{n}_i^\top$ and d_i are the normal vector of the ground plane and the distance to the plane respectively. $\mathbf{K}_i$ and $\mathbf{K}_{i+1}$ are the cameras' intrinsic parameter matrices.

However, to control the transform process, the value of internal parameters, i.e., $\mathbf{K}_i, \mathbf{R}_i, \mathbf{t}_i$, and d_i should be selected for each $\mathbf{H}_{i,i+1}$ by trial and error, which is a tedious job. To simplify this process, we use a pure rotation virtual camera model to eliminate $\mathbf{t}_i, d_i$, and use a Key-Point Bounding-Box Trick to estimate $\mathbf{K}_i$ for optimal viewports of intermediate feature maps. Whereas the traditional IPM uses at least 4 pairs of pre-calibrated correspondences on each view to estimate the integral $\mathbf{H}$ directly, we estimate the integral rotation by the horizon line specified on the image, which can be obtained by horizon line detection models, e.g., the HLW [14]. By representing the rotation in the axis-angle form, it is much easier to divide the rotation $\mathbf{R} \in \mathbb{SO}_3$ into sections by dividing the angle $\|\boldsymbol{\omega}\| \in \mathbb{R}$ and keep the axis direction unchanged. In this way, all internal parameters of each $\mathbf{H}_{i,i+1}$ are determined. Details about above procedure are given as follows.

1) Pure Rotation Virtual Cameras: It can be proven that a translated camera with unchanged intrinsic matrix can produce the same image as a fixed camera with accordingly modified intrinsic matrix. Thus, the consecutive perspective transform is modeled as synthesizing the ground plane image captured by a pure rotating camera, and (1) is simplified as,

$$\mathbf{H}_i = \mathbf{K}_{i+1} \mathbf{R}_{i,i+1} \mathbf{K}_i^{-1}, \quad (2)$$

and only the rotation matrix should be decomposed as

$$\mathbf{R} = \prod_{i=0}^{N} \mathbf{R}_{i,i+1}. \quad (3)$$

2) Estimating Integral Extrinsic Rotation by the Horizon Line: As extrinsic matrices with respect to the ground plane are not provided in TuSimple and ApolloScape datasets, we roughly estimate the integral rotation by the horizon line. Given two horizon points in the camera coordinates, $\mathbf{p}_{\text{left}}$ and $\mathbf{p}_{\text{right}}$, the normal vector of ground plane (facing to the ground) is calculated by a cross-production, i.e.,

$$\mathbf{n} = \frac{\mathbf{p}_{\text{left}} \times \mathbf{p}_{\text{right}}}{\|\mathbf{p}_{\text{left}} \times \mathbf{p}_{\text{right}}\|}. \quad (4)$$

In order to rotate the camera to face to the ground, its z-axis should be rotated to align with the normal vector $\mathbf{n}$. Hence, the rotation in axis-angle form is calculated as

$$\boldsymbol{\omega} = \frac{\mathbf{e}_3 \times \mathbf{n}}{\|\mathbf{e}_3 \times \mathbf{n}\|} \cdot \text{atan2}\left(\|\mathbf{e}_3 \times \mathbf{n}\|, \mathbf{e}_3 \cdot \mathbf{n}\right), \quad (5)$$

where $\mathbf{e}_3 = (0,0,1)^\top$ is a unit vector on z-axis.

Algorithm 1 Determine the Optimal Viewports through Key-Point Bounding Boxes

Require: $\mathbf{K}_i$, $\mathbf{R}_i$, $\mathbf{p}_{i,j}^I$, W_{i+1}

1: Convert points from Image I to Camera C: $\mathbf{p}_{i,j}^C = \mathbf{K}_i^{-1} \mathbf{p}_{i,j}^I$

2: Rotate points to view $i+1$: $\mathbf{p}_{i+1,j}^C = \mathbf{R}_i \mathbf{p}_{i,j}^C$

3: Normalize by the Z-dimension: $\mathbf{p}_{i+1,j}^C / = \text{abs}\left(Z_{\mathbf{p}_{i+1,j}^C}\right)$

4: Get the bounding box: $[bb_{top}, bb_{left}, bb_{width}, bb_{height}] = bbox(\mathbf{p}_{i+1,:}^C)$

5: Estimate focal length f_{i+1} as a scale ratio: $f_{i+1} = W_{i+1}/bb_{width}$

6: Estimate target view height with the same scale ratio: $H_{i+1} = f_{i+1} \times bb_{height}$

7: Estimate translation t_{i+1} by aligning left-top corner of target image view and bounding box in target camera coordinate: $t_{i+1} = (0,0)^\top - (f_{i+1} \times bb_{left}, f_{i+1} \times bb_{top})^\top$

8: Compose target intrinsic matrix: $\mathbf{K}_{i+1} = (f_{i+1}, t_{i+1})$

9: **return** $\mathbf{K}_{i+1}, H_{i+1}$

---

3) Decomposing the Extrinsic Rotation: Here we use the axis-angle representation for decomposing the rotation. We simply divide the integral angle into several even parts, and then convert each ω_i to the corresponding rotation matrix $\mathbf{R}_i$.

4) Optimal Viewports by Key-Point Bounding Boxes: While conducting IPM, image pixels at the edge often need to be cropped to prevent the target view from being too large. In order to preserve the informative pixels as many as possible, we roughly annotate the ground region by a set of border points in the front-view. The points are projected to the new view during each perspective transform. And we use a bounding box in the new view to determine the minimal available viewport which does not crop any projected key point. Thus, given a desirable target view width W_{i+1}, the corresponding intrinsic K_{i+1} and target view height H_{i+1} is determined, as shown in Algorithm 1.

C. Segmentation Heads

1) Semantic Segmentation: The lane and road marking detection problems are often cast as a semantic segmentation task [1] [9] [11]. In this work, we adopt a FCN-like network. By representing label classes as one-hot vectors, we predict the logits of each class at each pixel location. Then, we use the classic cross-entropy loss function to train this semantic segmentation branch.

2) Instance Segmentation: Semantic segmentation neither can distinguish different instances within the same category, nor can interpret separated parts of the same marking (dashed lines, zebra lines, etc.) as a unity. In order to solve that issue, we follow the work of LaneNet [2] to interpret the lane detection problem as an instance segmentation task. The network contains two branches. The semantic branch outputs a binary mask, while the instance embedding branch outputs an N-dimensional embedding vector for each pixel. The instance embedding branch is to disentangle the lane pixels

identified by the semantic branch. Here we also use a one-shot method based on distance metric learning [15]. During training, pixels from the same instance are "pulled" close to each other, and those from different instances are "pushed" away from each other by the loss function. During inference, the unmasked pixels are clustered in the embedding space to output the final result. For details of the loss function please refer to [2].

IV. EXPERIMENTS

We evaluate our network on the TuSimple [6] and ApolloScape [7] dataset respectively for the instance segmentation and semantic segmentation tasks. Our network is implemented by the PyTorch [16] framework.

A. TuSimple Benchmark

1) Dataset: The TuSimple Benchmark is a dedicated dataset for lane detection and consists of 3626 training and 2782 testing images, under good and medium weather conditions. The image resolution of this dataset is 1280x720 pixels. The annotation includes the *x*-position of the lane points at a number of discretized *y*-positions.

2) Metrics: The detection accuracy is calculated as the average correct number of points per image:

$$\text{acc} = \sum_{im} \frac{C_{im}}{S_{im}}, \tag{6}$$

where C_{im} denotes the number of correct points and S_{im} is the number of groundtruth points. A point is regarded as correctly detected when the difference between the groundtruth and the predicted point is smaller than a predefined threshold. Together with the accuracy, the false positive and false negative scores can also be calculated by

$$\text{FP} = \frac{F_{\text{pred}}}{N_{\text{pred}}}, \quad \text{FN} = \frac{M_{\text{pred}}}{N_{gt}}, \tag{7}$$

where F_{pred} denotes the number of mispredicted lanes, N_{pred} indicates the number of predicted lanes, M_{pred} is the number of missed groundtruth lanes and N_{gt} represents the number of all groundtruth lanes.

3) Training Details: Here we regard the lane detection as an instance segmentation task and train the instance segmentation network as shown in Fig. 3. Considering that the two classes (lane/background) are highly unbalanced, we apply the bounded inverse class weighting [17]. During the training process we use the Adam [18] optimizer, with a weight decay of 0.0005, a momentum of 0.95, a learning rate of 0.00004, and a batch size of 2. When the accuracy is without promotion up to 60 epochs, the learning rate drops to 10%. And the model converges after 220 epochs.

4) Evaluation Results: In comparison with other state-of-the-art methods [2] [3] [12], we show the test results in Table I, from which we can find out that our detection accuracy is already in the first echelon. It is worth to mention that all evaluation results above are in strict accordance with the metric defined by TuSimple. However, in our method the feature maps are warped into a bird's-eye view of the ground,

	acc	FP	FN	Ext.data	acc_under_horizon
Xingang Pan [3]	96.53	0.0617	0.018	yes	N/A
Yen-Chang Hsu [12]	96.50	0.0851	0.0269	no	N/A
Davy Neven [2]	96.40	0.078	0.0244	no	N/A
xxxxcvcxxxx	96.14	0.2033	0.0387	N/A	N/A
ResNet34-FCN (ours)	96.24	0.0746	0.0347	no	95.67
ResNet34-PTL-FCN (ours)	96.15	0.0818	0.0314	no	95.72
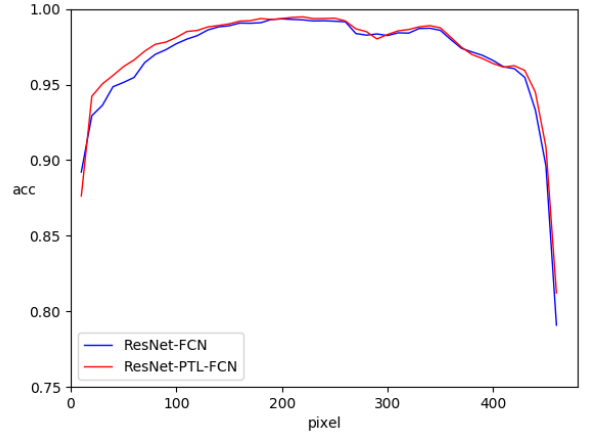
(a) Lane points accuracy vs. distance in pixels.
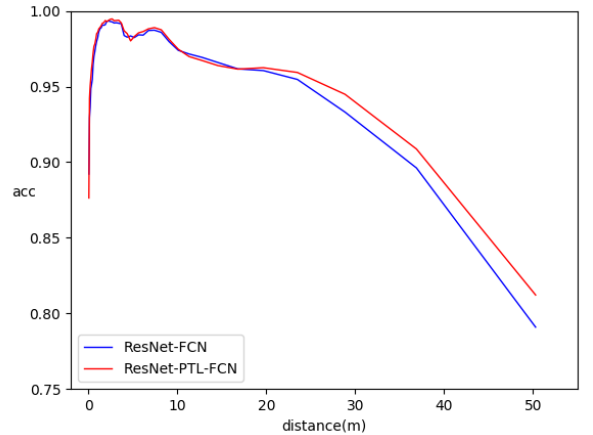
(b) Lane points accuracy vs. distance in meters.

Fig. 5. TuSimple Lane Detection Benchmark Results.

which force a part of the image above the horizon to be ignored by our method, and would lead to a slight decrease of the results. In order to make a fair comparison, we re-evaluated those samples only below the horizon, and ignored null samples which is labeled as -2 in the annotation. The new evaluation result is named as *acc_under_horizon* shown in the rightmost column of the Table I. We also plot the accuracy versus different distances from ego-vehicle in the line charts.

Fig. 5 (a) shows the accuracy in dependence of pixel distance to the image bottom. Fig. 5 (b) shows the accuracy in dependence of the real distance to the ego-vehicle. Both charts imply that our method can improve the detection accuracy of lane and road markings at longer distances. The qualitative comparison is shown in Figure 6.
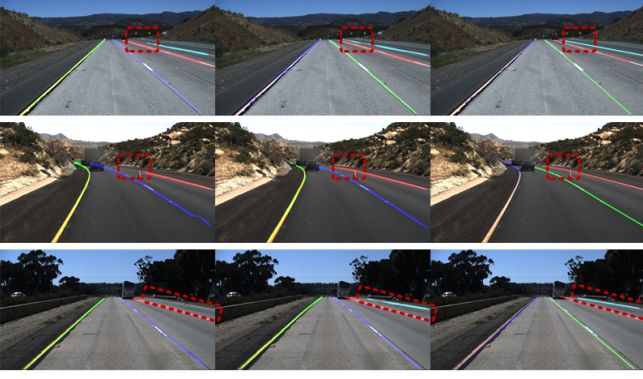
Fig. 6. Visualization of the comparison among the base-line, our method and the groundtruth on Tusimple dataset. Each row contains three submaps. From left to right: results w/o PTLs, results w/ PTLs, GT. And the area inside the red wireframe should be paid more attention.

B. ApolloScape Benchmark

1) Dataset: ApolloScape is a large scale dataset and contains seven branches for different tasks in the field of autonomous driving. Among them, the Lane Segmentation branch contains a diverse set of stereo video sequences recorded in street scenes from different cities, with high quality pixel-level annotations of more than 110,000 frames. Images in ApolloScape dataset are at a resolution of 3384x2710 pixels. The annotation information includes 35 kinds of lane and road markings from daily traffic scenarios, including but not limited to lanes, turning arrows, stop lines, zebra crossings. To the best of the authors' knowledge, no related works have been trained on the ApolloScape Lane Segmentation dataset. Therefore, we only show the ablation experimental results of our own method.

2) Metrics: The evaluation follows the recommendation of ApolloScape which uses the mean-IOU (mIOU) as the evaluation metric just like in [19]. For each class, given the predicted masks $M_{c,i}$ and the groundtruth $M_{c,i}^*$ for image i and class c, the evaluation metric is defined as:

$$
\begin{aligned}
IOU_c &= TP/(TP+FP+FN), \\
TP &= \sum_i \left\| M_{c,i} \cdot M_{c,i}^* \right\|_0, \\
FP &= \sum_i \left\| M_{c,i} \cdot \left(1 - M_{c,i}^*\right) \right\|_0, \\
FN &= \sum_i \left\| \left(1 - M_{c,i}\right) \cdot M_{c,i}^* \right\|_0, \\
mIOU &= \frac{1}{C} \sum_c^C IOU_c.
\end{aligned}
\tag{8}
$$

3) Training Details: Since the ApolloScape only provides pixel-level semantic annotations instead of instance information, we train the semantic segmentation network as shown in Fig. 3. Here we split 6288 frames out from the training set for validation. During the training process we use the Adam [18] optimizer, with a weight decay of 0.0005, a momentum of 0.95, a learning rate of 0.00004, and a batch size of 2. And the model converges after 25 epochs.
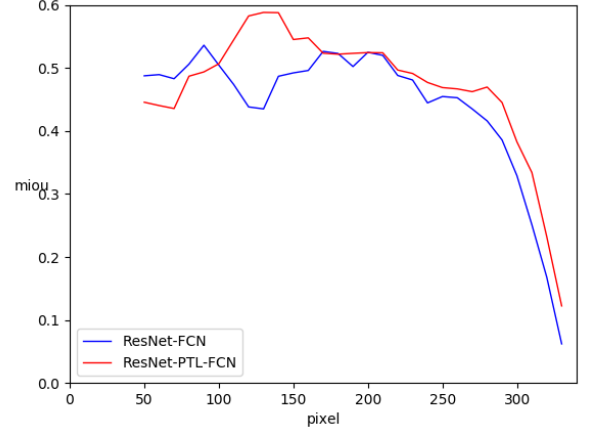
(a) mean-IOU vs. distance in pixels
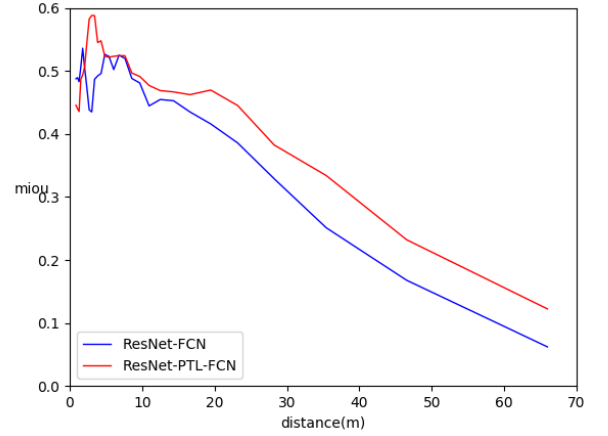
(b) mean-IOU vs. distance in meters

Fig. 7. Apollo Road Marking Semantic Segmentation Results.

4) Evaluation Results: Also in a fair way, we evaluate the image part below the horizon. Fig. 7 shows the results of mean-IOU accuracy at different distances. Table II shows mIOU value and IOU values of some common types of lane and road markings. We ignored the rest classes whose frequency is less than 0.001.

According to the experimental results, our method can effectively improve the detection accuracy of road markings at further distances, especially for the lane and road markings with richer structural features such as turning arrows. The qualitative comparison is shown in Fig. 8.

V. CONCLUSION

In this paper, we introduced a segmentation network architecture improved by consecutive homographic transforms for ground plane road marking detection. The parameters of consecutive transforms are clearly yielded by a pure rotating camera model and a key-point bounding-box trick. The proposed method is proven to be beneficial for distant lane and road marking detection. For the future research, we are going to incorporate an online scheme of extrinsic estimation into this structure. Also handling the non-flat ground surface

| category	class	ResNet18-FCN	ResNet18-PTL-FCN
arrow	thru	0.611	**0.692**
	thru & left turn	0.768	**0.800**
	thru & right turn	**0.824**	0.808
	left turn	0.767	**0.768**
stopping	stop line	0.665	**0.747**
zebra	crosswalk	**0.859**	0.858
lane	white solid	**0.832**	0.800
	yellow solid	**0.813**	0.803
	yellow double solid	0.886	**0.893**
	white broken	**0.791**	0.790
diamond	zebra attention	0.749	**0.775**
rectangle	no parking	0.652	**0.724**
mIOU		0.768	**0.788**
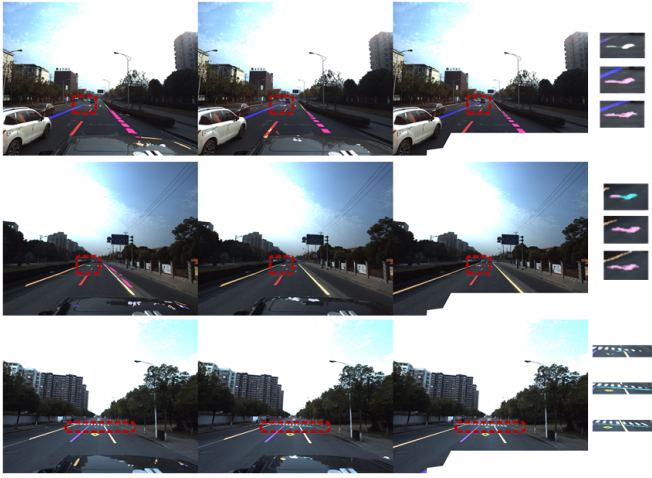
Fig. 8. Visualization of the comparison among the base-line, our method and the groundtruth on ApolloScape. Each row contains three submaps. From left to right: results w/o PTLs, results w/ PTLs, GT, the enlarged view of distant area, in which from top to bottom is corresponding to results w/o PTLs, results w/ PTLs, and GT.

which has unclear definition of ground normal vectors and horizon lines is one of the interesting topics.

ACKNOWLEDGMENT

REFERENCES

[1] Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Tae-Hee Lee, Hyun Seok Hong, Seung-Hoon Han, and In So Kweon. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. *arXiv:1710.06288 [cs]*, Oct 2017. arXiv: 1710.06288.

[2] Davy Neven, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Towards end-to-end lane detection: an instance segmentation approach. *arXiv:1802.05591 [cs]*, Feb 2018. arXiv: 1802.05591.

[3] Xingang Pan, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Spatial as deep: Spatial cnn for traffic scene understanding. 2018.

[4] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[5] Max Jaderberg, Karen Simonyan, and Andrew Zisserman. Spatial transformer networks. page 9.

[6] Tusimple lane detection benchmark. http://benchmark.tusimple.ai/.

[7] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The apolloscape dataset for autonomous driving. *arXiv: 1803.06184*, 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016.

[9] Xiaolong Liu, Zhidong Deng, Hongchao Lu, and Lele Cao. Benchmark for road marking detection: Dataset specification and performance baseline. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2017.

[10] Y. Wu, T. Yang, J. Zhao, L. Guan, and W. Jiang. Vh-hfcn based parking slot and lane markings segmentation on panoramic surround view. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1767–1772, June 2018.

[11] Weiwei Zhang, Zeyang Mi, Yaocheng Zheng, Qiaoming Gao, and Wenjing Li. Road marking segmentation based on siamese attention module and maximum stable external region. *IEEE Access*, 7:143710–143720, 2019.

[12] Yen-Chang Hsu, Zheng Xu, Zsolt Kira, and Jiawei Huang. Learning to cluster for proposal-free instance segmentation. *arXiv:1803.06459 [cs]*, Mar 2018. arXiv: 1803.06459.

[13] Tom Bruls, Horia Porav, Lars Kunze, and Paul Newman. The right (angled) perspective: Improving the understanding of road scenes using boosted inverse perspective mapping. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, page 302–309, Jun 2019.

[14] Scott Workman, Menghua Zhai, and Nathan Jacobs. Horizon lines in the wild. In *Procedings of the British Machine Vision Conference 2016*, pages 20.1–20.12. British Machine Vision Association, 2016.

[15] Bert De Brabandere, Davy Neven, and Luc Van Gool. Semantic instance segmentation with a discriminative loss function, 2017.

[16] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[17] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation, 2016.

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

[19] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.