Recursive Video Lane Detection

Dongkwon Jin
Korea University
dongkwonjin@mcl.korea.ac.kr

Dahyun Kim
Korea University
dhkim@mcl.korea.ac.kr

Chang-Su Kim
Korea University
changsukim@korea.ac.kr

Abstract

A novel algorithm to detect road lanes in videos, called recursive video lane detector (RVLD), is proposed in this paper, which propagates the state of a current frame recursively to the next frame. RVLD consists of an intra-frame lane detector (ILD) and a predictive lane detector (PLD). First, we design ILD to localize lanes in a still frame. Second, we develop PLD to exploit the information of the previous frame for lane detection in a current frame. To this end, we estimate a motion field and warp the previous output to the current frame. Using the warped information, we refine the feature map of the current frame to detect lanes more reliably. Experimental results show that RVLD outperforms existing detectors on video lane datasets. Our codes are available at https://github.com/dongkwonjin/RVLD.

Figure 1. (a) For lane detection, it is required to identify visible lane pixels, represented by yellow lines in the insets. (b) Due to occlusions by nearby vehicles or glistening conditions on wet roads, lanes may be unobvious, which are depicted by orange dotted lines. Besides, lanes may not be marked at crossroads. (c) In a current frame I^t, which is also the rightmost one in (b), some lane parts are occluded by a vehicle but visible in past frames. By utilizing visual cues along the temporal axis, depicted by cyan arrows, we can localize the implied lane parts more reliably.

1. Introduction

Lane detection is a task to localize lanes in a road scene, which is crucial for either autonomous or human driving. For lane detection, it is necessary to exploit visual cues of lanes, as illustrated in Figure 1(a). Early methods extract low-level features such as image gradients or colors [1–4]. Recently, deep learning techniques have been developed to cope with challenging scenes. Some of them are based on semantic segmentation [5–9], in which each pixel is classified into either lane category or not. To ensure lane continuity, several techniques have been proposed, including parametric curve modeling [10–14] and keypoint association [15–17]. However, despite yielding promising results, they may fail to detect less visible lanes, as in Figure 1(b). To detect such lanes reliably, anchor-based lane detectors [18–21] have been proposed. They generate a set of lane anchors, detect lanes through the binary classification of each anchor, and then regress the detected ones. However, all these methods detect lanes from still images without considering high inter-frame correlation in a video.

In autonomous driving systems, video frames are captured consecutively. Video lane detection aims to detect lanes in such a video by exploiting inter-frame correlation, rather than processing each frame independently. It can detect implied lanes in a current frame more reliably using past information, as shown in Figure 1(c). But, relatively few techniques have been proposed for video lane detection. Zou *et al.* [22] developed a memory network to aggregate the features of past and current frames. Similarly, in [23–26], they combined the features of a current frame with those of past frames and then detected lanes from the mixed features. However, these techniques require several past frames as input and do not reuse the mixed features in subsequent frames.

Recently, the first video lane dataset called VIL-100 [24] was constructed, containing 100 videos. However, the number of images is only 10K, and most images are collected from highway scenes. Also, OpenLane [27], a huge dataset for 3D lane detection, was proposed. It consists of 200K images from 1,000 videos and annotates lanes with both 2D and 3D coordinates. But, it is unsuitable for video lane detection because it provides annotations for visible lane parts only: First, the same lane is sometimes broken into multiple parts. Second, some annotations are temporally incoherent because of invisible parts in certain frames. To overcome

these issues, we modify OpenLane by filling in missing lane parts semi-automatically based on matrix completion [28]. The modified dataset is called OpenLane-V.

In this paper, we propose a novel video lane detector called RVLD, which records the state of a current frame and passes it recursively to the next frame to improve detection results. Figure 2 is an overview of the proposed RVLD. First, we design the intra-frame lane detector (ILD) that performs encoding, decoding, and then non-maximum suppression (NMS) to localize lanes in a still image. Second, we develop the predictive — or inter-frame — lane detector (PLD) to detect lanes in a current frame using the information in the previous one. Specifically, we estimate a motion field between adjacent frames and warp the previous output to the current frame. Using the warped information, we refine the feature map of the current frame to detect lanes more reliably. Experimental results show that the proposed RVLD outperforms existing techniques on both VIL-100 and OpenLane-V datasets.

This work has the following major contributions:

- The proposed RVLD improves the detection results in a current frame using a single previous frame only and yields outstanding performances on video datasets.
- We develop simple yet effective modules for motion estimation and feature refinement to exploit the previous information reliably.
- We modify the OpenLane dataset to make it more suitable for video lane detection. It is called OpenLane-V.[1]
- We introduce two metrics, flickering and missing rates, for video lane detection.

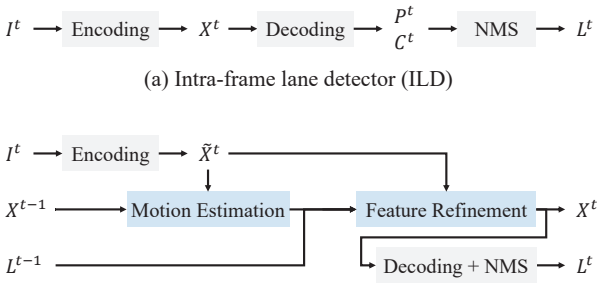
2. Related Work

2.1. Image-Based Lane Detection

Various techniques have been developed to detect lanes in a still image. Some are based on semantic segmentation [5–9], in which pixelwise classification is conducted to decide whether each pixel belongs to a lane or not. Pan *et al*. [5] propagated spatial information through message passing. Zheng *et al*. [6] adopted recurrent feature aggregation, while Qiu *et al*. [7] did multi-scale aggregation. Hou *et al*. [8] performed self-attention distillation. Moreover, Hou *et al*. [9] employed teacher and student networks. In [29], Qin *et al*. determined the location of each lane on selected rows only. Liu *et al*. [30] developed a conditional lane detection strategy based on the row-wise formulation.

In semantic segmentation, the continuity of a detected lane may not be preserved. To maintain the continuity, parametric curve modeling [10–14] and keypoint association [15–17] have been developed. Neven *et al*. [10] did the polynomial fitting of segmented lane pixels. In [11, 12], neural networks were designed to regress polynomial coefficients of lanes. Also, Liu *et al*. [13] developed a transformer network to predict cubic lane curves. Feng *et al*. [14] employed Bezier curves. In [15], Qu *et al*. estimated multiple keypoints and linked them to reconstruct lanes. Wang *et al*. [16] regressed the offsets from a starting point to keypoints and grouped them into a lane instance. Xu *et al*. [17] predicted four offsets from each lane point to the two adjacent ones and the topmost and bottommost ones.

Meanwhile, the anchor-based detection framework also has been developed [18–21, 31, 32]. These techniques generate lane anchors and then classify and regress each anchor by estimating the lane probability and the offset vector. Vertical line anchors were used in [31, 32]. Global features of straight line anchors were extracted to detect lanes in [18, 19]. Zheng *et al*. [21] extracted multi-scale feature maps and refined them by aggregating global features of learnable line anchors. Also, Jin *et al*. [20] introduced data-driven descriptors called eigenlanes. They generated lane anchors via a low-rank approximation of a lane matrix to represent lanes compactly and precisely.



(a) Intra-frame lane detector (ILD)

(b) Predictive lane detector (PLD)

Figure 2. (a) In ILD, we encode a frame I^t at time t into a feature map X^t and decode X^t into a probability map P^t and a coefficient map C^t. Then, via NMS, we generate a lane mask L^t. (b) In PLD, we utilize previous output X^{t-1} and L^{t-1} to detect lanes in I^t. First, we estimate a motion field from I^t to I^{t-1}. Then, we backward warp the previous output and refine the feature map $\tilde{X}^t$ of I^t into X^t using the warped information. Lastly, we obtain a lane mask L^t through the decoding and NMS processes.

2.2. Video-Based Lane Detection

There are relatively few video-based lane detectors. Zou *et al*. [22] and Zhang *et al*. [23] employed recurrent neural networks to exploit temporal correlation by fusing the features of a current frame with those of several past frames. Zhang *et al*. [24] developed a video lane detector using the first video lane dataset VIL-100. They aggregated features of a current frame and multiple past frames based on the attention mechanism [33, 34] to detect lanes in the current frame. Tabelini *et al*. [25] extracted lane features in video

[1] OpenLane-V is available at https://github.com/dongkwonjin/RVLD.

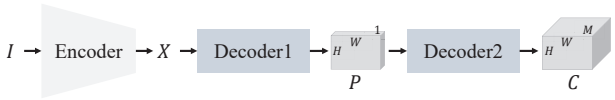


Figure 3. Encoding and decoding in ILD.

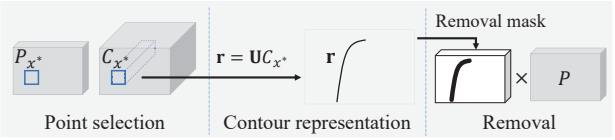


Figure 4. Illustration of NMS in ILD.

frames, based on the anchor-based detector in [19], and combined those features. Wang *et al*. [26] exploited spatial and temporal information in neighboring video frames by extending the feature aggregation module in [6]. They also designed loss functions to maintain the geometric consistency of lanes. These techniques require several past frames for feature aggregation but do not reuse the aggregated features in future frames. On the contrary, the proposed algorithm maintains the memory for a single previous frame only but propagates the temporal information recursively to extract lanes in each frame effectively.

3. Proposed Algorithm

Given a video sequence, we conduct lane detection in the first frame using ILD in Figure 2(a) and then do so in the remaining frames using PLD in Figure 2(b) recursively.

3.1. ILD

Encoding and decoding: ILD includes a single encoder and two decoders in series, as shown in Figure 3. Given an image I, the encoder extracts a feature map $X \in \mathbb{R}^{H\times W\times K}$ by aggregating multi-scale convolutional features of ResNet18 [35]. Then, the two decoders sequentially generate a probability map $P \in \mathbb{R}^{H\times W\times 1}$ and a coefficient map $C \in \mathbb{R}^{H\times W\times M}$ by

$$P = \sigma(f_1(X)) \quad \text{and} \quad C = f_2(P). \tag{1}$$

Here, $\sigma(\cdot)$ is the sigmoid function, and f_1 and f_2 are 2D convolutional layers. Let x denote a pixel coordinate, and P_x be the value of P at x. Note that P_x is the probability that pixel x belongs to a lane.

Also, each element in C is a coefficient vector in the M-dimensional eigenlane space [20], in which lanes are represented compactly with M basis vectors. Specifically, C_x is the coefficient vector for a lane containing x, if the lane exists. To regress such coefficient vectors more accurately, we use a positional bias map [34] as additional input to decoder 2. The architecture of ILD is described in detail in the supplemental document (Section A).

(a) Image I (b) Probability map P (c) Lane mask L

Figure 5. Some lanes are partly occluded in (a). While the probability map P does not represent the invisible parts reliably in (b), the lane mask L reconstructs continuous lanes clearly in (c).

NMS: Using the probability map P and the coefficient map C, we determine reliable lanes through NMS, as illustrated in Figure 4. First, we select the optimal pixel x^* with the highest probability in P. Then, we reconstruct the corresponding lane $\mathbf{r}$ by linearly combining M eigenlanes with the coefficient vector C_{x^*}, which is given by

$$\mathbf{r} = \mathbf{U}C_{x^*} = [\mathbf{u}_1, \cdots, \mathbf{u}_M]C_{x^*} \tag{2}$$

where $\mathbf{u}_1, \cdots, \mathbf{u}_M$ are the M eigenlanes [20]. Note that $\mathbf{r}$ is a column vector composed of the horizontal coordinates of lane points, which are sampled vertically. By dilating the lane curve $\mathbf{r}$, we construct a removal mask, as in Figure 4, and prevent the pixels within the mask from being selected in the remaining iterations. We iterate this NMS process until P_{x^*} is higher than 0.5.

Finally, using the selected lanes, ILD outputs a binary lane mask $L \in \mathbb{R}^{H\times W\times 1}$: $L_x = 1$ if x belongs to a lane, and $L_x = 0$ otherwise. Figure 5 illustrates that the lane mask L reduces the ambiguity in the probability map P and reconstructs continuous lanes effectively.

3.2. PLD

Problem formulation: We aim to detect lanes in a current frame I^t using the information in the past frame I^{t-1}, as in Figure 2(b). Note that X^{t-1} and L^{t-1} are the feature map and the lane mask of I^{t-1}, respectively. Let $\tilde{X}^t$ be the feature map of I^t obtained by the encoder. Both ILD and PLD use the same encoder, but in PLD we refine $\tilde{X}^t$ to X^t by exploiting X^{t-1} and L^{t-1}. Then, from the refined feature X^t, we obtain a more reliable lane mask L^t. The results X^t and L^t are, in turn, used to detect lanes in the future frame I^{t+1}. Notice that PLD is a first-order Markov chain [36], in which the future outcome at time $t+1$ is influenced by the current state at time t only and independent of the past states at times less than or equal to $t-1$. To perform this recursive detection reliably, we develop simple yet effective modules for motion estimation and feature refinement. The detailed structure of PLD is in the supplement (Section A).

Motion estimation: We estimate a motion field from the current frame I^t to the past frame I^{t-1} by designing a lightweight motion estimator in Figure 6. The motion estimator takes feature maps $\tilde{X}^t$ and X^{t-1} as input. Then, it estimates a down-sampled motion field $F_{\text{down}} \in$

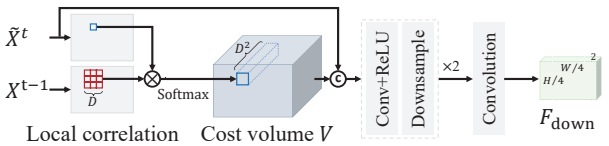


Figure 6. The structure of the proposed motion estimator: Given feature maps $\tilde{X}^t$ and X^{t-1} of adjacent frames I^t and I^{t-1}, we estimate a down-sampled motion field F_{down} from I^t to I^{t-1}.

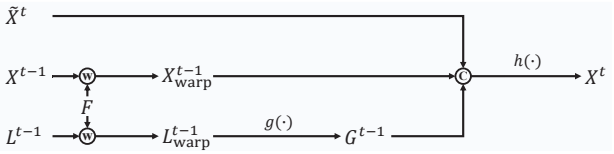


Figure 7. A diagram of the feature refinement module.

$\mathbb{R}^{H/4\times W/4\times 2}$, where each element indicates a 2D motion vector from I^t to I^{t-1}. To this end, as in [37], we construct a cost volume $V \in \mathbb{R}^{H\times W\times D^2}$ by first computing the local correlation

$$\bar{V}_x(d) = (X^{t-1}_{x+d})^\top \tilde{X}^t_x \tag{3}$$

where d is a displacement vector within the search window $\mathcal{D} = [-s, s] \times [-s, s]$. Thus, we compute $|\mathcal{D}| = D^2$ correlations for each x, where $D = 2s + 1$. Then, we obtain the cost volume V through $V_x = \text{softmax}(\bar{V}_x)$. Each element in V_x informs of the similarity between x and a matching candidate. We concatenate the cost volume V with $\tilde{X}^t$ and then apply convolutions and down-sampling operations to obtain the motion field F_{down}.

Feature refinement: Some lanes may be implied or even invisible in I^t due to occlusions, lack of light, or weather conditions. It is difficult to detect such lanes using the contextual information in the current frame only. In other words, $\tilde{X}^t$ of I^t is not sufficiently informative. Thus, we refine $\tilde{X}^t$ into X^t by exploiting the past information in I^{t-1}.

In Figure 7, we first warp the previous output X^{t-1} and L^{t-1} to the current frame by

$$X^{t-1}_{\text{warp}} = \phi_B(X^{t-1}, F), \quad L^{t-1}_{\text{warp}} = \phi_B(L^{t-1}, F), \tag{4}$$

where $F \in \mathbb{R}^{H\times W\times 2}$ is the up-sampled motion field of F_{down} via bilinear interpolation, and ϕ_B is the backward warping operator [38]. Then, from the warped lane mask L^{t-1}_{warp}, we obtain a guidance feature map by

$$G^{t-1} = g(L^{t-1}_{\text{warp}}) \tag{5}$$

where g is composed of 2D convolutional layers to increase the channel dimension to K. Notice that L^{t-1}_{warp} preserves the structural continuity of lanes similarly to Figure 5(c). G^{t-1} also contains such information. Thus, we may deliver the continuous lane information even for partially occluded lanes to I^t. Consequently, we produce the refined feature map X^t by aggregating G^{t-1}, X^{t-1}_{warp}, and $\tilde{X}^t$,

$$X^t = h([G^{t-1}, X^{t-1}_{\text{warp}}, \tilde{X}^t]) \tag{6}$$

where $[\cdot]$ denotes the channel-wise concatenation, and h consists of 2D convolutional layers to reduce the channel dimension back to K.

Then, using the refined feature X^t, we detect a more reliable lane mask L^t by performing the decoding and NMS processes in the same way as ILD, as shown in Figure 2.

3.3. Training

We define the loss for training ILD as

$$\ell_{\text{ILD}} = \ell_{\text{cls}}(P_x, \bar{P}_x) + \ell_{\text{reg}}(C_x, \bar{C}_x) \tag{7}$$

where P_x and C_x are the output of ILD in (1), and $\bar{P}_x$ and $\bar{C}_x$ are their ground-truth (GT), respectively. Also, ℓ_{cls} is the focal loss [39] over binary classes, and ℓ_{reg} is the LIoU loss [21] between a predicted lane contour $\mathbf{r}$ in (2) and its ground-truth $\bar{\mathbf{r}}$.

For PLD, in addition to the loss in (7), we employ a loss function

$$\ell_{\text{flow}} = \|\bar{P}^{t-1}_{\text{warp}} - \bar{P}^t\|^2 = \|\phi_B(\bar{P}^{t-1}, F) - \bar{P}^t\|^2 \tag{8}$$

to train the motion estimation module. In other words, it is trained to yield a motion field F such that the warped result of the ground-truth probability map $\bar{P}^{t-1}$ is close to $\bar{P}^t$.

We first train ILD and then, after fixing it, train the motion estimation and feature refinement modules in PLD from scratch. The results of PLD in a current frame are, in turn, used in the next frame. Hence, for stable training of PLD, we compose three consecutive frames as a training unit. The training process and hyper-parameters are detailed in the supplement (Section A).

Figure 8. In OpenLane [27], only visible lane parts are annotated, so the same lane may be split into multiple parts. In contrast, in OpenLane-V, each lane is seamlessly annotated, and invisible parts are inpainted. It is recommended to watch the accompanying video to see the improved annotations.

4. Datasets

4.1. OpenLane-V

OpenLane [27] is one of the largest lane datasets with about 200K images from 1,000 videos and about 880K lane

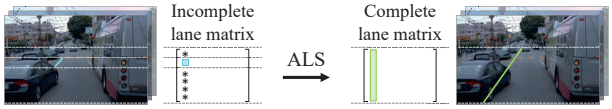


Figure 9. In OpenLane, only visible lane parts, such as the one in cyan, are annotated. Thus, the lane matrix is incomplete. Through ALS [41], we complete the matrix, so the missing parts are reconstructed and the whole lane in green is properly annotated.

annotations. Whereas both visible and invisible lanes are annotated in other datasets [5, 24, 40], OpenLane annotates only visible lane parts as in Figure 8, which makes it unsuitable for video lane detection. Note that some parts are separated, but they actually belong to the same lane. Also, in challenging videos, lane parts are missing unpredictably, so they are not temporally consistent. We improve the annotations in OpenLane and construct a modified dataset, called OpenLane-V, by filling in missing parts semi-automatically based on matrix completion [28].

Figure 9 illustrates the completion process. Let $\mathbf{A} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_L] \in \mathbb{R}^{N \times L}$ be a lane matrix [20] containing L lanes in a dataset, where each lane $\mathbf{x}_i = [x_1, x_2, \ldots, x_N]^\top$ is represented by the x-coordinates of N points sampled uniformly in the vertical direction. In OpenLane, however, some entries in $\mathbf{A}$ are not measured, since only visible lane parts are annotated. In other words, $\mathbf{A}$ is an incomplete matrix.

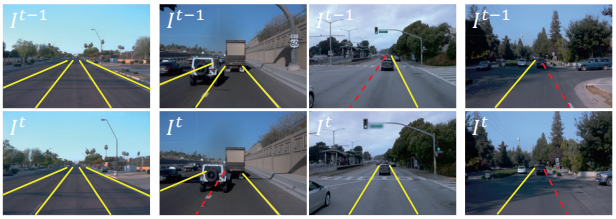
To complete $\mathbf{A}$, we perform factorization $\mathbf{A} \approx \mathbf{U}^\top \mathbf{V}$, where $\mathbf{U} \in \mathbb{R}^{R \times N}$, $\mathbf{V} \in \mathbb{R}^{R \times L}$, and R is the rank of the factored $\mathbf{A}$. In [20], it was shown that a lane matrix has low-rank characteristics, so we set $R = 3$. Then, we find optimal $\mathbf{U}$ and $\mathbf{V}$ to minimize an objective function:

$$\min_{\mathbf{U},\mathbf{V}} \sum_{(i,j)\in\mathcal{O}} (\mathbf{A}_{i,j} - \mathbf{u}_i^\top \mathbf{v}_j)^2 + \lambda\Big(\sum_i \|\mathbf{u}_i\|^2 + \sum_j \|\mathbf{v}_i\|^2\Big), \tag{9}$$

where $\mathcal{O}$ is the coordinate set of observed entries, and λ is a regularization parameter. Since finding the optimal factors is NP-hard [41], we adopt the alternating least square (ALS) algorithm [41, 42], in which we optimize $\mathbf{V}$ after fixing $\mathbf{U}$, and vice versa. This is repeated until convergence. The objective function becomes convex when one of the factors is fixed, so ALS is guaranteed to converge to a local minimum.

Entirely invisible lanes with no annotations, however, cannot be restored automatically. They disappear suddenly in videos, causing flickering artifacts. We manually annotate such lanes using the information in neighboring frames. Moreover, in some videos, too large a portion of lane parts is invisible, making the completed results unreliable. We remove such videos from OpenLane-V. This semi-automatic process is described in detail in the supplement (Section B).

As a result, OpenLane-V consists of about 90K images from 590 videos. It is split into a training set of 70K images from 450 videos and a test set of 20K images from 140 videos. As in the CULane dataset [5], we annotate up to 4 road lanes in each image, which are ego and alternative lanes, to focus on important lanes for driving.



(a) Stable (b) Flickering (c) Missing

Figure 10. Three cases of lane detection results at consecutive frames: Correctly detected lanes are in yellow, while falsely dismissed ones are in red. In (a), each lane is detected correctly and stably in both frames. In (b), a detected lane at one frame is not detected at the other, leading to a flicker. In (c), the right lane is missed at both frames because it is not marked on the crossroad.

4.2. VIL-100

VIL-100 [24] is the first dataset for video lane detection containing 100 videos. It is split into 80 training and 20 test videos. Each video has 100 frames. VIL-100 is less challenging than OpenLane-V and contains only 10K images.

5. Experimental Results

In addition to the results in this section, it is recommended to watch the accompanying video clips to compare video lane detection results more clearly.

5.1. Evaluation Metrics

Conventional metrics: For lane detection, image-based metrics are generally employed. Each lane is regarded as a thin stripe with 30 pixel width [5]. Then, a predicted lane is declared correct if its IoU ratio with GT is greater than τ. The precision and the recall are computed by

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{10}$$

where TP is the number of correctly detected lanes, FP is that of false positives, and FN is that of false negatives. Then, the F-measure at threshold τ is defined as

$$\text{F1}^\tau = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision+Recall}}. \tag{11}$$

Also, mIoU is computed by averaging the IoU scores of correctly detected lanes.

Video metrics: We propose two video metrics to assess the temporal stability of detected lanes. When a lane suddenly disappears or a new lane is suddenly detected at a frame, vehicle maneuvers are hard to control, possibly leading to dangerous situations. It is important to achieve temporally

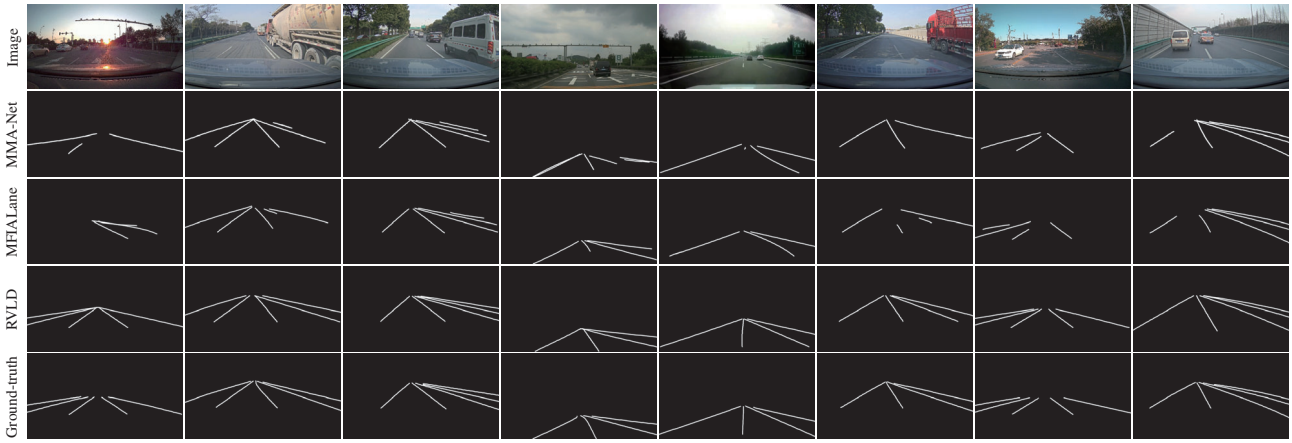


Figure 11. Comparison of lane detection results on the VIL-100 dataset.

Table 1. Comparison of mIoU and F1 scores on VIL-100: image lane detectors and video ones are listed separately.

	mIoU	$F1^{0.5}$	$F1^{0.8}$
LaneNet [10]	0.633	0.721	0.222
ENet-SAD [8]	0.616	0.755	0.205
LSTR [43]	0.573	0.703	0.131
RESA [6]	0.702	0.874	0.345
LaneATT [19]	0.664	0.823	-
MFIALane [7]	-	<u>0.905</u>	<u>0.565</u>
MMA-Net [24]	0.705	0.839	0.458
LaneATT-T [25]	0.692	0.846	-
TGC-Net [26]	<u>0.738</u>	0.892	0.469
RVLD (Proposed)	**0.787**	**0.924**	**0.582**

Table 2. Flickering and missing rates on VIL-100.

	$R_F^{0.5}$	$R_M^{0.5}$	$R_F^{0.8}$	$R_M^{0.8}$
MMA-Net [24]	0.047	0.128	**0.200**	0.428
MFIALane [7]	<u>0.042</u>	<u>0.127</u>	0.206	<u>0.323</u>
RVLD (Proposed)	**0.038**	**0.050**	<u>0.203</u>	**0.306**

stable lane detection. There are three cases for a matching pair of lanes at adjacent frames: *Stable*, *Flickering*, and *Missing*. A stable case is one where a lane is successfully detected at both frames, as shown in Figure 10(a). In a flickering case, a lane is detected at one frame but missed at the other, as in Figure 10(b). A missing case is the worst one in which a lane is missed consecutively at both frames, as in Figure 10(c).

Let N be the number of GT lanes that have matching instances at previous frames, and let N_S, N_F, N_M be the numbers of stable, flickering, and missing cases, respectively. Note that $N = N_S + N_F + N_M$. Then, we define the flickering and missing rates as

$$R_F^\tau = \frac{N_F}{N}, \quad R_M^\tau = \frac{N_M}{N}, \tag{12}$$

where τ is the IoU threshold for correct detection.

5.2. Comparative Assessment

VIL-100: We compare the proposed RVLD with conventional image lane detectors [6–8, 10, 19, 43] and video ones [24–26] on VIL-100. Table 1 lists the mIoU and F1 scores. RVLD outperforms all conventional algorithms in all image metrics. Especially, RVLD is better than the state-of-the-art video lane detector TGC-Net by significant margins of 0.049, 0.032, and 0.113 in mIoU, $F1^{0.5}$, and $F1^{0.8}$, respectively. Note that RVLD uses a single previous frame only, whereas the existing video lane detectors [24–26] use two or more past frames as input. MFIALane, an image lane detector, yields high F1 scores, but it underperforms as compared with RVLD. This is because it processes each image independently and may fail to detect implied lanes. In contrast, RVLD detects lanes in a current frame more reliably by exploiting past information.

Table 2 compares the R_F and R_M rates. LaneATT-T and TGC-Net are not compared because their source codes are unavailable. MMA-Net achieves the lowest flickering rate $R_F^{0.8}$, but it does so because its missing rate $R_M^{0.8}$ is too high. Except for $R_F^{0.8}$, the proposed RVLD yields the lowest flickering and missing rates, indicating that RVLD provides temporally more stable detection results.

Figure 11 presents some detection results. MMA-Net does not detect unobvious lanes precisely, even though it uses several past frames as input. MFIALane also fails to process those lanes reliably, for it is image-based. In contrast, the proposed RVLD provides better results using past information effectively.

OpenLane-V: Table 3 compares RVLD with the state-of-the-art image lane detectors [7, 16, 21, 30] and the video lane detectors [22–24] on OpenLane-V. Notice that MFIALane [7] ranks 2nd on VIL-100 in Table 1. Also, CondLaneNet [30], GANet [16], and CLRNet [21] are recent detectors achieving outstanding performances on image datasets. We

Table 3. Comparison on OpenLane-V.

	mIoU	$F1^{0.5}$	$F1^{0.8}$	$R_F^{0.5}$	$R_M^{0.5}$	$R_F^{0.8}$	$R_M^{0.8}$
ConvLSTM [22]	0.529	0.641	0.353	0.058	0.282	0.091	0.574
ConvGRUs [23]	0.540	0.641	0.355	0.064	0.288	0.094	0.576
MMA-Net [24]	0.574	0.573	0.328	0.044	0.461	0.071	0.671
MFIALane [7]	0.697	0.723	0.475	0.061	0.300	0.080	0.519
CondLaneNet [30]	0.698	0.780	0.450	0.047	0.239	0.084	0.531
GANet [16]	0.716	0.801	0.530	0.048	0.198	0.082	0.443
CLRNet [21]	**0.735**	0.789	0.554	0.054	0.224	0.086	0.430
RVLD (Proposed)	0.727	**0.825**	**0.566**	**0.014**	**0.167**	**0.051**	**0.406**

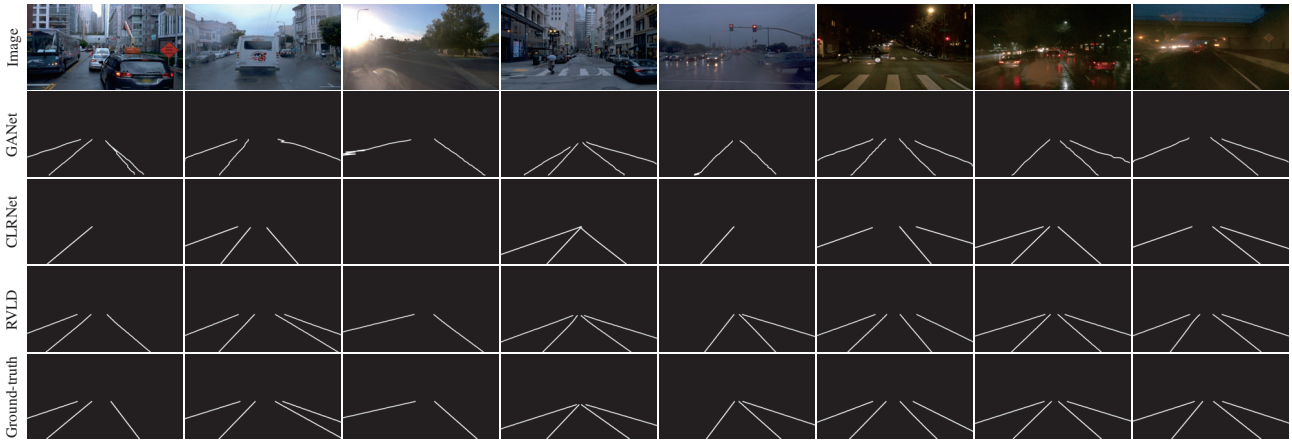


Figure 12. Comparison of lane detection results on the OpenLane-V dataset.

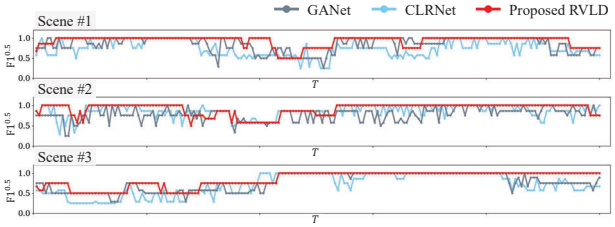


Figure 13. Comparison of $F1^{0.5}$ scores per frame in three challenging videos in OpenLane-V. Compared to RVLD, GANet and CLRNet suffer from more fluctuating results. The frame-by-frame comparisons of lane detection results of CLRNet and RVLD on these three scenes are available in a supplemental video clip.

train all these detectors on OpenLane-V using publicly available source codes.

In Table 3, we see that RVLD outperforms the existing techniques in every metric, except that it yields a slightly lower mIoU score than CLRNet. GANet and CLRNet yield decent results in terms of image-based mIoU and F1 scores, but they perform poorly in terms of video-based flickering and missing rates. Especially, their $R_F^{0.5}$ scores are over 0.044, which are at least three times higher than that of RVLD. In other words, they provide temporally less stable lane detection results than RVLD, as also supported by Figure 13. Moreover, RVLD provides better results than the existing video lane detectors, which are ConvLSTM, ConvGRUs, and MMA-Net. Note that RVLD uses a single previous frame only to improve detection results, whereas the existing methods do several past frames. This confirms that the proposed RVLD exploits temporal correlation more effectively. The efficacy of using a single previous frame is demonstrated in the supplement (Section C).

Figure 12 shows detection results. The state-of-the-art image-based techniques inaccurately detect implied lanes or simply miss them in challenging scenes. In contrast, RVLD detects the lanes reliably. More detection results are presented in the supplemental document (Section D).

5.3. Ablation Studies

We conduct ablation studies to analyze the efficacy of the proposed RVLD and its components. Table 4 compares several ablated methods on OpenLane-V. Method I detects road lanes in each frame using ILD only, without exploiting temporal correlation in a video. In Methods II∼V, both ILD and PLD are employed: Method II does not warp the previous output and uses it directly to refine the feature map of a current frame. Method III does not use the guidance feature map G^{t-1} in (6). Method IV does not reuse the refined feature in the future frame. Specifically, in Figure 2(b), it passes the ILD feature $\tilde{X}^t$, instead of the refined feature X^t, to the future frame.

Efficacy of PLD: Method I underperforms badly. As compared with the proposed RVLD (Method V), its F1 score is reduced from 0.822 to 0.784, and its flickering rate $R_F^{0.5}$ is

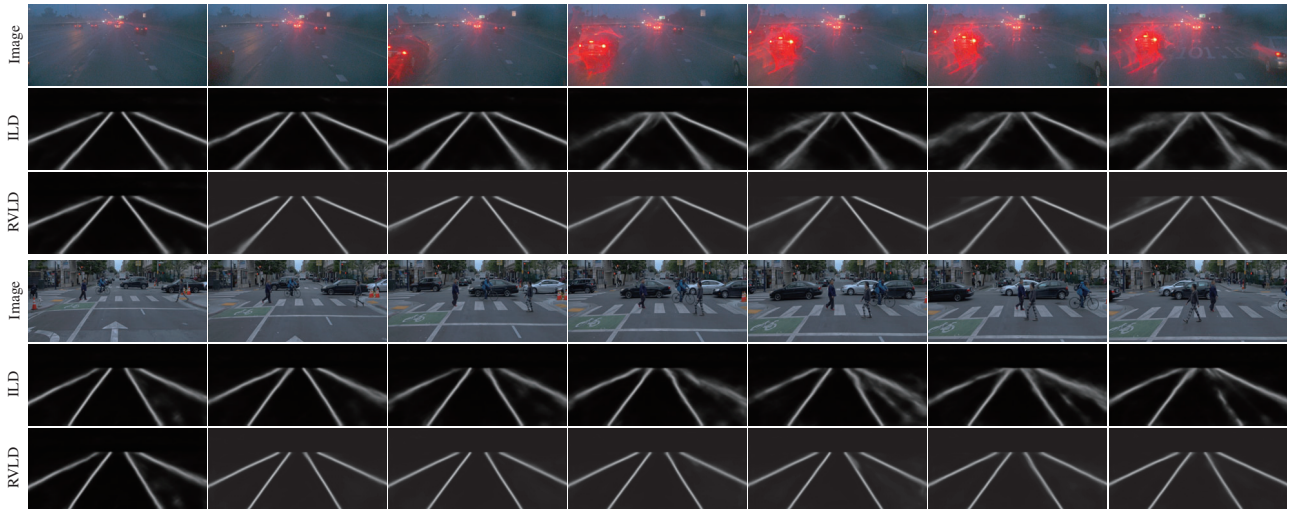


Figure 14. Comparison of the lane probability maps generated by ILD and RVLD for consecutive video frames in the OpenLane-V dataset. It is recommended to see the accompanying video clip for more comparisons of lane probability maps.

Table 4. Ablation studies of the proposed RVLD on OpenLane-V.

		$F1^{0.5}$	$R_F^{0.5}$	$R_M^{0.5}$
I.	ILD	0.787	0.053	0.197
II.	ILD + PLD (w/o warping)	0.816	0.017	0.191
III.	ILD + PLD (w/o guidance)	0.821	0.027	0.187
IV.	ILD + PLD (w/o reuse)	0.822	0.017	0.172
V.	RVLD	0.825	0.014	0.167

about 4 times higher. Using ILD only, it fails to detect unobvious lanes reliably and suffers from temporal instability.

Figure 14 compares the lane probability maps of ILD and RVLD on challenging videos in OpenLane-V. As lanes vanish or gradually disappear in videos, ILD yields highly unstable results. On the contrary, RVLD estimates the probability maps clearly and stably, for it refines the feature map of each frame using the past output and propagates the refined map to the future frame.

Efficacy of motion estimation: Compared to RVLD, Method II yields inferior results, indicating that RVLD can exploit temporal correlation more effectively via the motion-based alignment of past information.

Efficacy of guidance information: Without guidance features in Method III, the flickering rate $R_F^{0.5}$ nearly doubles. Guidance features are essential for stable lane detection, especially when lanes are partially occluded, since they help to maintain lane continuity.

Efficacy of feature reuse: Compared to Method V, Method IV yields poorer results, indicating that it is more effective to pass the PLD feature X^t, rather than the ILD feature $\tilde{X}^t$, to the future frame. In other words, it is better to reuse the refinement feature in the future frame.

Runtime: Table 5 lists the runtime for each stage of RVLD. We use a PC with AMD Ryzen 9 5900X CPU and NVIDIA RTX 3090 GPU. The processing speed of RVLD is about 80 frames per second, which is sufficiently fast for applications. ILD, excluding the motion estimation and feature refinement, is faster, but it is less accurate and suffers from temporal incoherence of detected lanes.

Table 5. Runtime analysis of the proposed RVLD. MEFR means the motion estimation and feature refinement processes in PLD. The processing times are reported in seconds per frame.

Encoding	MEFR	Decoding	NMS	Total
0.0060s	0.0034s	0.0011s	0.0020s	0.0125s

6. Conclusions

We proposed a novel video lane detector, called RVLD, which extracts informative features for a current frame and passes them recursively to the next frame. First, we designed ILD to localize lanes in a still image. Second, we developed PLD to exploit past information to detect lanes in a current frame more reliably. Experimental results demonstrated that RVLD outperforms existing techniques meaningfully. Moreover, we modified OpenLane to construct OpenLane-V, which is about 10 times larger than VIL-100, and proposed two new video-based metrics, the flickering rate R_F and the missing rate R_M.

Acknowledgements

This work was conducted by Center for Applied Research in Artificial Intelligence (CARAI) grant funded by DAPA and ADD (UD230017TD) and also supported by the NRF grants funded by the Korea government (MSIT) (No. NRF-2021R1A4A1031864 and No. NRF-2022R1A2B5B03002310).

References

[1] Y. He, H. Wang, and B. Zhang, "Color-based road detection in urban traffic scenes," *IEEE Trans. Intel. Transp. Syst.*, vol. 5, no. 4, pp. 309–318, 2004. 1

[2] M. Aly, "Real time detection of lane markers in urban streets," in *Intelligent Vehicles Symposium*, 2008. 1

[3] A. B. Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progresss in road and lane detection: A survey," *Mach Vis. Appl.*, vol. 25, no. 3, pp. 727–745, 2014. 1

[4] S. Zhou, Y. Jiang, J. Xi, J. Gong, G. Xiong, and H. Chen, "A novel lane detection based on geometrical model and Gabor filter," in *Intelligent Vehicles Symposium*, 2010. 1

[5] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial CNN for traffic scene understanding," in *Proc. AAAI*, 2018. 1, 2, 5

[6] T. Zheng, H. Fang, Y. Zhang, W. Tang, Z. Yang, H. Liu, and D. Cai, "RESA: Recurrent feature-shift aggregator for lane detection," in *Proc. AAAI*, 2021. 1, 2, 3, 6

[7] Z. Qiu, J. Zhao, and S. Sun, "MFIALane: Multiscale feature information aggregator network for lane detection," *IEEE Trans. Intel. Transp. Syst.*, 2022. 1, 2, 6, 7

[8] Y. Hou, Z. Ma, C. Liu, and C. C. Loy, "Learning lightweight lane detection CNNs by self attention distillation," in *Proc. IEEE ICCV*, 2019. 1, 2, 6

[9] Y. Hou, Z. Ma, C. Liu, T.-W. Hui, and C. C. Loy, "Inter-region affinity distillation for road marking segmentation," in *Proc. IEEE CVPR*, 2020. 1, 2

[10] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, and L. Van Gool, "Towards end-to-end lane detection: An instance segmentation approach," in *Intelligent Vehicles Symposium*, 2018. 1, 2, 6

[11] B. Wang, Z. Wang, and Y. Zhang, "Polynomial regression network for variable-number lane detection," in *Proc. ECCV*, 2020. 1, 2

[12] L. Tabelini, R. Berriel, T. M. Paixao, C. Badue, A. F. De Souza, and T. Oliveira-Santos, "PolyLaneNet: Lane estimation via deep polynomial regression," in *Proc. IEEE ICPR*, 2021. 1, 2

[13] R. Liu, Z. Yuan, T. Liu, and Z. Xiong, "End-to-end lane shape prediction with transformers," in *Proc. IEEE WACV*, 2021. 1, 2

[14] Z. Feng, S. Guo, X. Tan, K. Xu, M. Wang, and L. Ma, "Rethinking efficient lane detection via curve modeling," in *Proc. IEEE CVPR*, 2022. 1, 2

[15] Z. Qu, H. Jin, Y. Zhou, Z. Yang, and W. Zhang, "Focus on local: Detecting lane marker from bottom up via key point," in *Proc. IEEE CVPR*, 2021. 1, 2

[16] J. Wang, Y. Ma, S. Huang, T. Hui, F. Wang, C. Qian, and T. Zhang, "A keypoint-based global association network for lane detection," in *Proc. IEEE CVPR*, 2022. 1, 2, 6, 7

[17] S. Xu, X. Cai, B. Zhao, L. Zhang, H. Xu, Y. Fu, and X. Xue, "RCLane: Relay chain prediction for lane detection," in *Proc. ECCV*, 2022. 1, 2

[18] X. Li, J. Li, X. Hu, and J. Yang, "Line-CNN: End-to-end traffic line detection with line proposal unit," *IEEE Trans. Intel. Transp. Syst.*, vol. 21, no. 1, pp. 248–258, 2019. 1, 2

[19] L. Tabelini, R. Berriel, T. M. Paixao, C. Badue, A. F. De Souza, and T. Oliveira-Santos, "Keep your eyes on the lane: Real-time attention-guided lane detection," in *Proc. IEEE CVPR*, 2021. 1, 2, 3, 6

[20] D. Jin, W. Park, S.-G. Jeong, H. Kwon, and C.-S. Kim, "Eigenlanes: Data-driven lane descriptors for structurally diverse lanes," in *Proc. IEEE CVPR*, 2022. 1, 2, 3, 5

[21] T. Zheng, Y. Huang, Y. Liu, W. Tang, Z. Yang, D. Cai, and X. He, "CLRNet: Cross layer refinement network for lane detection," in *Proc. IEEE CVPR*, 2022. 1, 2, 4, 6, 7

[22] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen, and Q. Wang, "Robust lane detection from continuous driving scenes using deep neural networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 41–54, 2019. 1, 2, 6, 7

[23] J. Zhang, T. Deng, F. Yan, and W. Liu, "Lane detection model based on spatio-temporal network with double convolutional gated recurrent units," *IEEE Trans. Intel. Transp. Syst.*, vol. 23, no. 7, pp. 6666–6678, 2021. 1, 2, 6, 7

[24] Y. Zhang, L. Zhu, W. Feng, H. Fu, M. Wang, Q. Li, C. Li, and S. Wang, "VIL-100: A new dataset and a baseline model for video instance lane detection," in *Proc. IEEE ICCV*, 2021. 1, 2, 5, 6, 7

[25] L. Tabelini, R. Berriel, A. F. De Souza, C. Badue, and T. Oliveira-Santos, "Lane marking detection and classification using spatial-temporal feature pooling," in *International Joint Conference on Neural Networks*, 2022. 1, 2, 6

[26] M. Wang, Y. Zhang, W. Feng, L. Zhu, and S. Wang, "Video instance lane detection via deep temporal and geometry consistency constraints," in *Proc. ACM Multimedia*, 2022. 1, 3, 6

[27] L. Chen, C. Sima, Y. Li, Z. Zheng, J. Xu, X. Geng, H. Li, C. He, J. Shi, Y. Qiao, and J. Yan, "PersFormer: 3D lane detection via perspective transformer and the OpenLane benchmark," in *Proc. ECCV*, 2022. 1, 4

[28] E. J. Candès and T. Tao, "The power of convex relaxation: Near-optimal matrix completion," *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2053–2080, 2010. 2, 5

[29] Z. Qin, H. Wang, and X. Li, "Ultra fast structure-aware deep lane detection," in *Proc. ECCV*, 2020. 2

[30] L. Liu, X. Chen, S. Zhu, and P. Tan, "CondLaneNet: A top-to-down lane detection framework based on conditional convolution," in *Proc. IEEE ICCV*, 2021. 2, 6, 7

[31] Z. Chen, Q. Liu, and C. Lian, "PointLaneNet: Efficient end-to-end CNNs for accurate real-time lane detection," in *Intelligent Vehicles Symposium*, 2019. 2

[32] H. Xu, S. Wang, X. Cai, W. Zhang, X. Liang, and Z. Li, "CurveLane-NAS: Unifying lane-sensitive architecture search and adaptive point blending," in *Proc. ECCV*, 2020. 2

[33] S. W. Oh, J.-Y. Lee, N. Xu, and S. J. Kim, "Video object segmentation using space-time memory networks," in *Proc. IEEE ICCV*, 2019. 2

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NeurIPS*, 2017. 2, 3

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE CVPR*, 2016. 3

[36] W. L. Winston and J. B. Goldberg, *Operations Research: Applications and Algorithms*, vol. 3. Thomson Brooks/Cole Belmont, 2004. 3

[37] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," in *Proc. IEEE CVPR*, 2018. 4

[38] G. Wolberg, *Digital Image Warping*. IEEE Computer Society Press, 1990. 4

[39] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE CVPR*, 2017. 4

[40] "TuSimple benchmark." [Online]. Available: https://github.com/TuSimple/tusimple-benchmark. 5

[41] T. Hastie, R. Mazumder, J. D. Lee, and R. Zadeh, "Matrix completion and low-rank SVD via fast alternating least squares," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 3367–3402, 2015. 5

[42] P. Jain, P. Netrapalli, and S. Sanghavi, "Low-rank matrix completion using alternating minimization," in *Proc. the 45th Annual ACM Symp. Theory of Computing*, 2013. 5

[43] R. Liu, Z. Yuan, T. Liu, and Z. Xiong, "End-to-end lane shape prediction with transformers," in *Proc. IEEE WACV*, 2021. 6