OMR: Occlusion-Aware Memory-Based Refinement for Video Lane Detection

Dongkwon Jin[1,2] and Chang-Su Kim[1]

[1] School of Electrical Engineering, Korea University, Seoul, Korea
[2] Samsung Advanced Institute of Technology
dongkwonjin@mcl.korea.ac.kr, changsukim@korea.ac.kr

Abstract. A novel algorithm for video lane detection is proposed in this paper. First, we extract a feature map for a current frame and detect a latent mask for obstacles occluding lanes. Then, we enhance the feature map by developing an occlusion-aware memory-based refinement (OMR) module. It takes the obstacle mask and feature map from the current frame, previous output, and memory information as input, and processes them recursively in a video. Moreover, we apply a novel data augmentation scheme for training the OMR module effectively. Experimental results show that the proposed algorithm outperforms existing techniques on video lane datasets. Our codes are available at https://github.com/dongkwonjin/OMR.

Keywords: Video lane detection · Occlusion · Feature refinement

1 Introduction

Lane detection aims to localize lanes in a road scene, which is essential for either enabling autonomous driving or assisting human driving. It is, however, difficult to detect lanes, which may be unobvious due to occlusions by nearby vehicles or severe weather conditions. For lane detection, early methods tried to find visible lane cues by extracting low-level features [1, 7, 8, 42]. Recently, many techniques have been developed to deal with implied lanes using deep features. Some adopt the semantic segmentation framework [9, 10, 21, 24, 40] and classify each pixel into either the lane category or not. Several attempts have been made to extract continuous lane information, including curve modeling [5, 16, 19, 29, 31] and keypoint association [25, 32, 37]. Meanwhile, anchor-based lane detectors [12, 13, 28, 34, 41] have been proposed. They predefine a set of lane anchors and then detect lanes through the classification and regression of each anchor, ensuring lane continuity. However, all these methods are image-based detectors that process each frame independently, so they often fail to yield temporally stable detection results, especially when some lanes are occluded by objects, as illustrated in Fig. 1.

Video lane detectors also have been developed. These techniques exploit past information to detect lanes in a current frame, which may help to identify implied

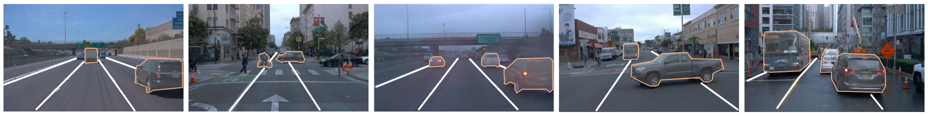


Fig. 1: Examples of road scenes, in which some lane parts are occluded by several objects. Visible lanes and obstructing objects are depicted by white lines and orange polygons, respectively.

lanes more reliably. Most of them [27, 33, 38, 39, 44] adopt the framework in Fig. 2(a). These video detectors extract the features of several past and current frames, aggregate those features, and detect lanes in the current frame using the mixed features. However, they do not reuse the mixed features in future frames. Recently, a recursive video lane detector (RVLD) [11] was proposed. As in Fig. 2(b), RVLD enhances the features of a current frame using the single previous frame only through motion estimation and feature refinement. Also, it passes the state of the current frame recursively to the next frame. RVLD outperforms existing image and video lane detectors, but it may detect lanes inaccurately because it heavily relies on the information in a current frame. In particular, when lanes in a current frame are severely occluded by nearby vehicles, RVLD tends to produce unreliable detection results.

In this paper, we propose a novel video lane detector incorporating an occlusion-aware memory-based refinement (OMR) module. As in Fig. 2(c), it utilizes a latent obstacle mask and memory information to enhance the feature map of a current frame. First, we extract a feature map and detect latent obstacles, hindering lane visibility, from the current frame. Then, we refine the feature map through the OMR module, which takes the obstacle mask and feature map from the current frame, the previous output, and the memory information as input. Moreover, we develop an effective data augmentation scheme for training the OMR module robustly. Experimental results demonstrate that the proposed algorithm outperforms existing techniques on both VIL-100 [39] and OpenLane-V [11] datasets.

This work has the following major contributions:

- The proposed OMR module improves lane detection results in a current frame by exploiting an obstacle mask and memory information.
- We introduce a novel training strategy for video lane detection to identify lanes more robustly.
- The proposed algorithm yields outstanding lane detection results on video datasets.

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 [9, 10, 21, 24, 40], in which each pixel is

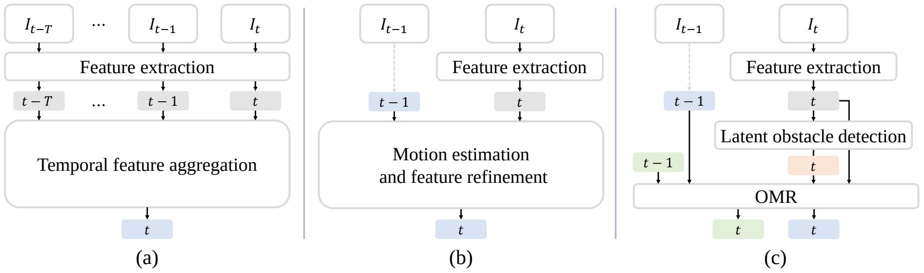


Fig. 2: There are three approaches to video lane detection. In (a), the feature maps of a current frame I_t and the past T frames are extracted and mixed to refine the feature map of I_t. In (b), only a single previous frame is used to enhance the feature map of I_t, and the enhanced one is passed recursively to the subsequent frame. The proposed algorithm in (c) utilizes obstacle and memory information to improve the feature map of I_t via the OMR module. Note that gray, blue, green, and orange boxes represent intra-frame features, refined features, recorded memory, and a latent obstacle mask, respectively.

dichotomized into the lane category or not. To boost the pixelwise classification, Pan *et al.* [21] propagated the information of pixels spatially. In [24,40], recurrent or multi-scale feature aggregation was performed. Hou *et al.* [10] performed self-attention distillation, while Hou *et al.* [9] employed teacher and student networks. For efficient lane detection, Qin *et al.* [23] determined the location of each lane on selected rows only. Liu *et al.* [15] developed a conditional lane detection scheme based on the row-wise approach.

Several methods attempt to maintain lane continuity by regressing curve parameters [5,16,19,29,31] or associating multiple keypoints [25,32,37]. Neven *et al.* [19] did the polynomial fitting of segmented lane pixels. In [29,31], polynomial coefficients of lanes were regressed using neural networks. Also, Liu *et al.* [16] predicted cubic lane curves based on a transformer network. Feng *et al.* [5] employed Bezier curves. In [25], Qu *et al.* extracted multiple keypoints and linked them to reconstruct lanes. Wang *et al.* [32] estimated the offsets from a starting point to keypoints and grouped them into a lane instance. Xu *et al.* [37] predicted four offsets bilaterally from each lane point to the two nearest ones and the two farthest ones.

Meanwhile, the anchor-based detection framework has been adopted in [3, 12, 13, 28, 34, 36, 41]. These techniques form lane anchors and then classify and regress each anchor by estimating the lane probability and the positional offset. In [3, 36], vertical line anchors were employed. In [13, 28], straight line anchors were used to extract global features of lanes. Zheng *et al.* [41] extracted multi-scale feature maps and refined them by aggregating global features of learnable line anchors. Jin *et al.* [12] introduced data-driven descriptors called eigenlanes. They generated curved anchors as well as straight ones by clustering all training lanes in the eigenlane space. Xiao *et al.* [34] produced a heat map to estimate the starting points and directions of anchors.

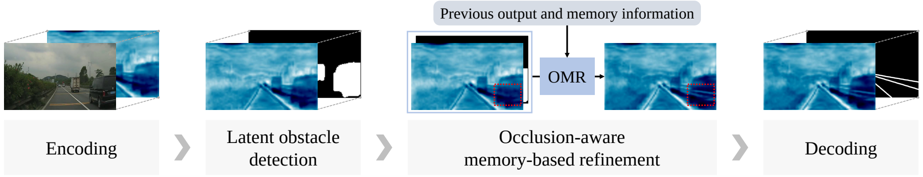

Previous output and memory information
OMR
Encoding
Latent obstacle detection
Occlusion-aware memory-based refinement
Decoding

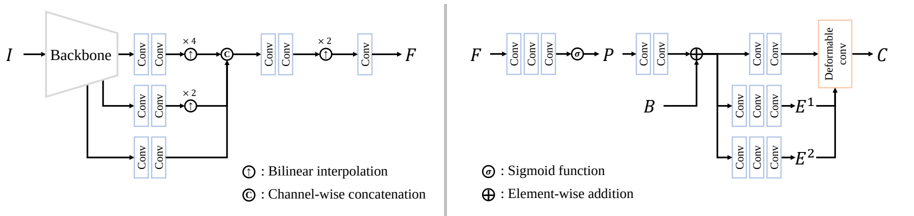
Fig. 3: Overview of the proposed algorithm, which performs four steps: encoding, latent obstacle detection, OMR, and decoding. In this example, the rightmost lane is partially occluded by nearby vehicles, so the encoded features are defective, making lane detection difficult. The proposed algorithm, however, can detect the implicit lane precisely by refining the features within the occluded regions effectively. As depicted by dotted red boxes, we see that the proposed OMR module enhances the features of the occluded lane into more discriminative ones.

2.2 Video-Based Lane Detection

There are several video-based lane detectors. Most of them combine the features of a current frame with those of several past frames to detect lanes in the current frame, as in Fig. 2(a). To exploit temporal correlation, Zou *et al.* [44] and Zhang *et al.* [38] employed recurrent neural networks. Zhang *et al.* [39] aggregated features of a current frame and multiple past frames based on the attention mechanism [20, 30]. Tabelini *et al.* [27] fused global features of lanes in video frames after extracting them using the anchor-based detector in [28]. Wang *et al.* [33] modified the feature aggregation module in [40] to exploit spatiotemporal information in neighboring video frames. However, these video detectors do not reuse the aggregated features in future frames. Recently, Jin *et al.* [11] developed the RVLD method, which uses only a single previous frame but propagates the state of the current frame to the next frame recursively. As in Fig. 2(b), RVLD estimates a motion field, warps the previous output to the current frame, refines the feature map of the current frame, and passes it to the subsequent frame. Despite promising results, RVLD often misses or incorrectly detects unobvious lanes, especially highly occluded lanes. In contrast, the proposed algorithm effectively processes those lanes by detecting latent obstacles and utilizing both memory information and previous output, as shown in Fig. 2(c).

3 Proposed Algorithm

Given a video sequence, we perform lane detection, which is composed of encoding, latent obstacle detection, feature refinement, and decoding steps. Fig. 3 shows an overview of the proposed algorithm. For clarity, we describe the encoding and decoding processes in advance. Notice that the proposed OMR module is performed in the feature refinement step.


I
Backbone
Conv
Conv
× 4
× 2
Conv
F
Ⓘ : Bilinear interpolation
Ⓒ : Channel-wise concatenation
P
B
Deformable conv
C
E^1
E^2
σ : Sigmoid function
⊕ : Element-wise addition

(a) Encoder (b) Decoder

Fig. 4: Architecture of the encoder and the decoder: (a) Given an image I, three coarsest feature maps are extracted using a backbone network. After matching their channel dimensions and resolutions, they are encoded into a combined feature map F. (b) From a feature map F, a lane probability map P is estimated. Then, by applying a deformable convolution, a lane coefficient map C is predicted from P.

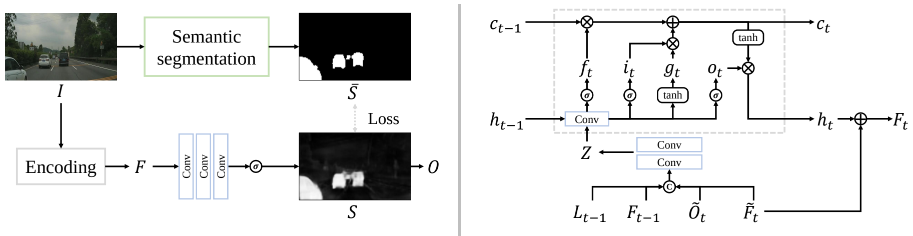
3.1 Encoding

Given an image I, we extract a convolutional feature map $F \in \mathbb{R}^{H\times W\times K}$, as done in [12, 23]. Fig. 4(a) shows the encoding process. First, we extract multi-scale feature maps using ResNet18 [6] as the backbone. Then, we combine the three coarsest maps, which have 1/8, 1/16, and 1/32 of the resolution of I, respectively. Specifically, we match the channel dimensions of the feature maps to K. We then match the resolutions of the two coarser maps to the finest one via bilinear interpolation and concatenate them. From the concatenated feature map, we obtain F via convolutional and up-sampling layers. We set K to 64.

3.2 Decoding

From a feature map F, we produce two output maps to determine lanes in I, as shown in Fig. 4(b). We obtain a lane probability map $P \in \mathbb{R}^{H\times W\times 1}$ using a series of convolutional layers and a sigmoid function. Let $\mathbf{x}$ denote the position vector of a pixel, and let $P(\mathbf{x})$ be the probability that pixel $\mathbf{x}$ belongs to a lane. Also, we estimate a lane coefficient map $C \in \mathbb{R}^{H\times W\times M}$. Each element in C is a coefficient vector in the M-dimensional eigenlane space [12], in which lanes are represented compactly with M basis vectors. Since $C(\mathbf{x})$ represents geometric information for a lane containing $\mathbf{x}$, we use a positional bias [30] to regress the coefficient vector more accurately. To this end, we obtain a sinusoidal positional bias $B \in \mathbb{R}^{H\times W\times K}$ and combine it with F via element-wise addition. From the combined feature map, we generate an offset map $E^1 \in \mathbb{R}^{H\times W\times 50}$, a weight map $E^2 \in \mathbb{R}^{H\times W\times 25}$, and a transformed feature map, and then perform deformable convolution [43] with a 5×5 kernel to regress C.

Using the probability map P and the coefficient map C, we determine reliable lanes through non-maximum suppression (NMS), as done in [11]. First, we select the optimal pixel $\mathbf{x}^*$ with the highest probability in P. Then, we form the corresponding lane $\mathbf{r}$ by linearly combining M eigenlanes with the coefficient



(a) Latent Obstacle Detection

(b) OMR Module

Fig. 5: Block diagrams of the latent obstacle detection and OMR: (a) From the encoded feature map F, a binary probability map S for latent obstacles is predicted. By thresholding S, a binary obstacle mask O is determined. To obtain its ground-truth $\bar{S}$, SegFormer [35], which is a semantic segmentation algorithm, is employed. (b) In OMR, four input maps L_{t-1}, F_{t-1}, $\tilde{O}_t$, and $\tilde{F}_t$ are aggregated to Z. Then, using the combined feature map Z, ConvLSTM [26] is used to update (h_{t-1}, c_{t-1}) to (h_t, c_t) via (4). Then, h_t is added to $\tilde{F}_t$ to refine it into F_t. Blue boxes represent a series of 2D convolution operations with batch-normalization and ReLU function.

vector $C(\mathbf{x}^*)$, which is given by

$$\mathbf{r} = \mathbf{U}C(\mathbf{x}^*) = [\mathbf{u}_1, \cdots, \mathbf{u}_M]C(\mathbf{x}^*) \tag{1}$$

where $\mathbf{u}_1, \cdots, \mathbf{u}_M$ are the M eigenlanes [12]. Note that $\mathbf{r}$ is a column vector containing the horizontal coordinates of lane points, which are uniformly sampled vertically. After dilating the lane curve $\mathbf{r}$, we construct a mask and remove the pixels within it to prevent their selection in the remaining iterations. We iterate this NMS process until $P(\mathbf{x}^*)$ is higher than 0.5.

Finally, using the selected lanes, we output a lane mask $L \in \mathbb{R}^{H \times W \times 1}$: $L(\mathbf{x}) = 1$ if $\mathbf{x}$ belongs to a lane, and $L(\mathbf{x}) = 0$ otherwise.

3.3 Latent Obstacle Detection

Various objects appear in road environments, such as trucks on highways or pedestrians on crossroads. These objects hinder the visibility of lanes, posing significant challenges in lane detection. To address such occlusion, we treat them as latent obstacles and detect them accordingly. Fig. 5(a) shows the obstacle detection process. More specifically, we predict a binary probability map $S \in \mathbb{R}^{H \times W \times 1}$ from the feature map F of I by

$$S = \sigma(w_1(F)), \tag{2}$$

where σ is the sigmoid function and w_1 is composed of 2D convolutional layers. $S(\mathbf{x})$ is the probability that pixel $\mathbf{x}$ belongs to an obstacle on a road surface. Then, we obtain an obstacle mask $O \in \mathbb{R}^{H \times W \times 1}$, in which $O(\mathbf{x})$ is assigned to 1 if $S(\mathbf{x})$ is higher than a threshold, and 0 otherwise. We set the threshold to 0.3.

Since there is no ground-truth (GT) segmentation of those obstacles in existing lane datasets, we generate their pseudo-labels by employing a semantic segmentation algorithm. In this work, we adopt SegFormer [35], which is efficient and yields high performance on the Cityscapes dataset [4]. Eight of the 19 categories in the dataset, including 'car,' 'bus,' and 'rider' classes, are regarded as potential lane-occluding obstacles. Then, we perform SegFormer to produce a GT binary segmentation mask $\bar{S} \in \mathbb{R}^{H \times W \times 1}$ for those candidates in I. Using the predicted map S and its GT mask $\bar{S}$, the obstacle detector in (2) is trained. The training process is detailed in Section 3.5.

3.4 Occlusion-Aware Memory-Based Refinement

In a current frame I_t, some lanes may be unobvious due to the occlusions by neighboring obstacles, as mentioned previously. Furthermore, various factors such as poor lighting and adverse weather conditions affect the visibility of lanes. To deal with these issues, we utilize the obstacle detection results from I_t, previous output from I_{t-1}, and memory information through the proposed OMR module. Fig. 5(b) shows the structure of the OMR module.

Let $\tilde{F}_t$ be the feature map of I_t obtained by the encoder. In $\tilde{F}_t$ and the following notations, *tilde* represents output produced from a still image. Thus, the probability map $\tilde{P}_t$, the coefficient map $\tilde{C}_t$, the lane mask $\tilde{L}_t$, and the obstacle mask $\tilde{O}_t$ are decoded from $\tilde{F}_t$. Using the OMR module, we aim to refine $\tilde{F}_t$ to F_t and improve the detection results from $\tilde{L}_t$ to L_t. To this end, we first obtain a feature map $Z \in \mathbb{R}^{H \times W \times K}$ by aggregating $\tilde{O}_t$ and $\tilde{F}_t$ with the previous output L_{t-1} and F_{t-1} via

$$Z = w_4([w_2(L_{t-1}), F_{t-1}, w_3(\tilde{O}_t), \tilde{F}_t]) \tag{3}$$

where $[\cdot]$ is channel-wise concatenation, and w_2, w_3, and w_4 are 2D convolution layers. Also, notice that L_{t-1} and F_{t-1} are already refined in the previous step and recursively used in the current step. Then, we exploit memory information by employing a variant of ConvLSTM [26]. Specifically, from the mixed feature map Z, we perform a series of ConvLSTM operations by

$$\begin{aligned}
f_t &= \sigma(w_5(Z) + w_6(h_{t-1})), \\
i_t &= \sigma(w_7(Z) + w_8(h_{t-1})), \\
g_t &= \sigma(w_9(Z) + w_{10}(h_{t-1})), \\
o_t &= \tanh(w_{11}(Z) + w_{12}(h_{t-1})), \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t, \quad h_t = o_t \odot c_t.
\end{aligned} \tag{4}$$

Here, $w_5, \dots, w_{12}$ are 2D convolutional layers, and $\odot$ is element-wise multiplication, respectively. Also, h_t and c_t are a hidden state and cell state, and f_t, i_t, g_t, and o_t are a forget gate, input gate, control gate, and output gate, respectively. The four gates are used to update the cell state and hidden state sequentially. h_1 and c_1 are initialized by learnable parameters. Also, we do not use the cell vectors for estimating the gate parameters. Then, we produce the refined feature map F_t by

$$F_t = \tilde{F}_t + h_t, \tag{5}$$

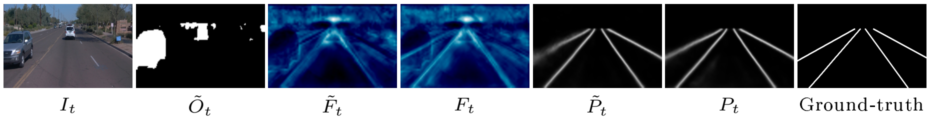


Fig. 6: Visualization of the obstacle mask $\tilde{O}_t$, the feature map $\tilde{F}_t$, the probability map $\tilde{P}_t$, and their enhanced ones F_t and P_t. In the current frame I_t, some lane parts are occluded by nearby vehicles. The visible lane parts of $\tilde{F}_t$ are sufficiently discriminative for identifying them. In contrast, the features for the occluded parts are not so informative. Thus, $\tilde{P}_t$ is poorly estimated around the occlusions. However, in F_t and P_t, the lane features and lane probabilities for the occluded regions are restored faithfully using the proposed OMR module. To visualize these feature maps, min-max normalization is done.

Fig. 6 illustrates that the OMR module refines $\tilde{F}_t$ to F_t reliably, even though some lane parts are obstructed by vehicles.

Lastly, using the refined feature map F_t, we produce a reliable lane mask L_t by performing the decoding process, as described in Section 3.2.

3.5 Training

Data configuration: For each image I, we generate a GT lane probability map $\bar{P} \in \mathbb{R}^{H\times W\times 1}$, a GT coefficient map $\bar{C} \in \mathbb{R}^{H\times W\times M}$, and a GT obstacle mask $\bar{S} \in \mathbb{R}^{H\times W\times 1}$. First, $\bar{P}(\mathbf{x}) = 1$ if pixel $\mathbf{x}$ belongs to a lane, and $\bar{P}(\mathbf{x}) = 0$ otherwise. To obtain $\bar{C}$, M eigenlanes are extracted by processing all lanes in a training set, as done in [12]. Then, each lane in an image is transformed to an M-dimensional coefficient vector. In the image, $\bar{C}(\mathbf{x})$ is assigned the coefficient vector if $\mathbf{x}$ belongs to one of the lanes. Otherwise, if $\mathbf{x}$ does not belong to any lane, $\bar{C}(\mathbf{x})$ is assigned the zero vector. To obtain $\bar{S}$, we adopt Segformer [35], which predicts the semantic segmentation mask for 19 categories. Thus, $\bar{S}(\mathbf{x})$ is set to 1 if $\mathbf{x}$ belongs to eight of those categories, such as 'car,' 'bicycle,' 'bus,' 'truck,' 'train,' 'motorcycle,' 'person,' and 'rider,' and 0 otherwise.

Loss function: We perform training in two steps. First, we define the loss for training the encoder and decoder as

$$\ell_{\text{step1}} = \ell_{\text{cls}}(\tilde{P}, \bar{P}) + \ell_{\text{reg}}(\tilde{C}, \bar{C}) + \ell_{\text{cls}}(\tilde{S}, \bar{S}). \tag{6}$$

Here, $\tilde{P}$, $\tilde{C}$, and $\tilde{S}$ are the decoded output from the encoded feature map $\tilde{F}$ of I. Also, ℓ_{cls} is the focal loss [14] over binary classes, and ℓ_{reg} is the LIoU loss [41] between a predicted lane contour $\mathbf{r}$ in (1) and its ground-truth $\bar{\mathbf{r}}$. Then, we define the loss for training the proposed OMR module as

$$\ell_{\text{step2}} = \ell_{\text{cls}}(P, \bar{P}) + \ell_{\text{reg}}(C, \bar{C}) \tag{7}$$

where P and C are the output from the refined feature map F. Also, during the training of the OMR module, we freeze the parameters of the pretrained encoder and decoder.

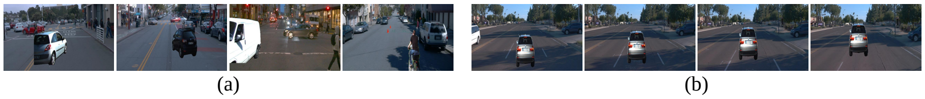


Fig. 7: (a) In a training set, each image is synthesized by overlaying new objects, such as vehicles or cyclists, from the KINS dataset. (b) Additionally, video sequences are regenerated by linearly varying the sizes and positions of these objects over frames. The resulting images appear natural because fully shaped objects are extracted from KINS.

Data augmentation: In real-world environments, lanes unexpectedly disappear and reappear due to occlusions by nearby objects. To cope with such challenging scenarios reliably, we introduce a data augmentation scheme, which is applied to the training of the OMR module. To this end, we employ the KINS dataset [22]. It is an amodal instance segmentation dataset, in which a fully-shaped mask per each object is given involving its occluding parts. Given an original video sequence, we randomly select an object from the KINS dataset and then attach its full shape to the video frames. We also vary the size and position of the object linearly over frames. Fig. 7 shows some examples of the synthetically generated frames.

4 Experimental Results

4.1 Implementation Details

We adopt ResNet18 [6] as a backbone. We use AdamW optimizer [18] with an initial learning rate of 10^{-1} and halve it after every 100,000 iterations four times. Also, we use a batch size of four for 400,000 iterations. We resize an input image to 384×640. As the default setting, we set $H = 96$, $W = 160$, $K = 64$, and $M = 6$. We also employ SegFormer-B5 [35] for the semantic segmentation.

4.2 Datasets

VIL-100 [39] 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 includes some challenging scenes in which some lanes are highly occluded by big trucks or buses. In each frame, 2D lane coordinates up to 6 road lanes are annotated.

OpenLane-V [11], which is modified from OpenLane [2], is a huge and diverse video lane dataset. It 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 [21], up to 4 road lanes are annotated in each image, corresponding to ego and alternative lanes. OpenLane-V is more difficult for lane detection than VIL-100, because of various challenging factors: lane occlusions, severe weather conditions, poor illumination, or lack of lane marking on crossroads.

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

	Approach	mIoU(↑)	F1(↑)	R_F(↓)	R_M(↓)
LaneNet [19]	Image-based	0.633	0.721	-	-
ENet-SAD [10]		0.616	0.755	-	-
LSTR [17]		0.573	0.703	-	-
RESA [40]		0.702	0.874	-	-
LaneATT [28]		0.664	0.823	-	-
MFIALane [24]		-	0.905	0.047	0.128
ADNet [34]		0.781	0.920	0.039	0.043
MMA-Net [39]	Video-based	0.705	0.839	0.042	0.127
LaneATT-T [27]		0.692	0.846	-	-
TGC-Net [33]		0.738	0.892	-	-
RVLD [11]		**0.787**	0.924	0.038	0.050
Proposed	Video-based	0.774	**0.936**	**0.026**	**0.038**

4.3 Evaluation Metrics

Image metrics: For lane detection, image-based metrics are generally employed. Each lane is regarded as a thin stripe with a 30-pixel width [21]. Then, a predicted lane is declared correct if its IoU ratio with GT is greater than 0.5. The precision and the recall are computed by

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{8}$$

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 is defined as

$$\text{F1} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision+Recall}}. \tag{9}$$

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

Video metrics: In autonomous driving systems, achieving temporally stable lane detection is crucial to prevent hazardous situations caused by the sudden detection or absence of a lane within a frame. To assess the temporal stability of detected lanes, two video metrics [11] are employed. 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 detected successfully in both frames. In a flickering case, a lane is detected in one frame but missed in the other. A missing case is the worst one in which both frames miss a lane consecutively.

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, the flickering and missing rates are defined as

$$R_F = \frac{N_F}{N}, \quad R_M = \frac{N_M}{N}, \tag{10}$$

where the IoU threshold for correct detection is 0.5.

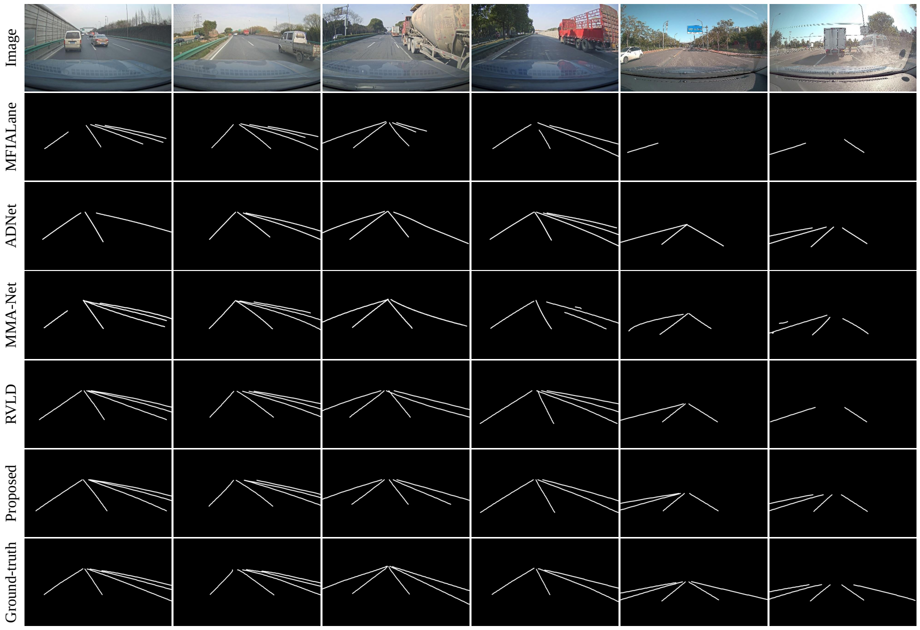


Fig. 8: Comparison of lane detection results on the VIL-100 dataset.

4.4 Comparative Assessment

VIL-100: We compare the proposed algorithm with conventional image lane detectors [10,17,19,24,28,34,40] and video ones [11,27,33,39] on VIL-100. Table 1 lists the mIoU, F1 scores, $\mathrm{R_F}$ rates, and $\mathrm{R_M}$ rates. The proposed algorithm outperforms the existing techniques in every metric, except for mIoU. Especially, the proposed algorithm is better than the state-of-the-art video lane detector RVLD by the same margins of 0.012 in F1, $\mathrm{R_F}$, and $\mathrm{R_M}$. RVLD improves the detection results in a current frame using a single previous frame only based on motion estimation and feature refinement. However, it may fail to detect implied lanes occluded by neighboring vehicles. This is because the motion estimator in RVLD tends to produce inaccurate motion field for occluded regions. In contrast, the proposed algorithm detects those lanes more reliably by exploiting the obstacle masks and memory information. ADNet [34], a recent image-based detector, yields decent performances, but the scores are inferior to those of the proposed algorithm in most metrics.

Fig. 8 presents some detection results. Both MFIALane and ADNet fail to detect unobvious lanes precisely, for it is image-based. MMA-Net does not detect those lanes, even though it uses several past frames as input. RVLD is better than these techniques, but it also processes the occluded lanes poorly. In contrast, the proposed algorithm provides better results based on the obstacle reasoning effectively.

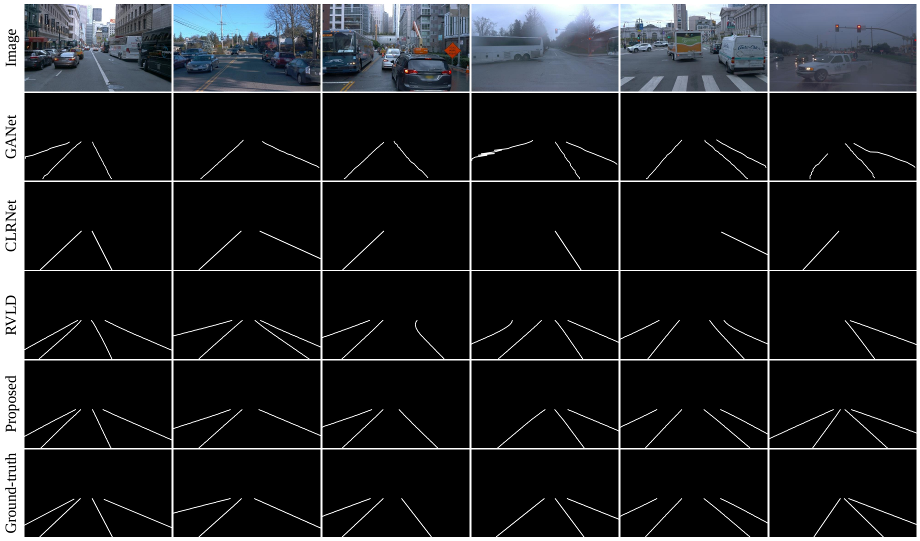


Fig. 9: Comparison of lane detection results on the OpenLane-V dataset.

Table 2: Comparison on OpenLane-V.

	Approach	mIoU(↑)	F1(↑)	R_F(↓)	R_M(↓)
MFIALane [24]	Image-based	0.697	0.723	0.061	0.300
CondLaneNet [15]		0.698	0.780	0.047	0.239
GANet [32]		0.716	0.801	0.048	0.198
CLRNet [41]		0.735	0.789	0.054	0.224
ConvLSTM [44]	Video-based	0.529	0.641	0.058	0.282
ConvGRUs [38]		0.540	0.641	0.064	0.288
MMA-Net [39]		0.574	0.573	0.044	0.461
RVLD [11]		0.727	0.825	**0.014**	0.167
Proposed	Video-based	**0.742**	**0.836**	0.016	**0.162**

OpenLane-V: Table 2 compares the proposed algorithm with the image lane detectors [15,24,32,41] and the video lane detectors [11,38,39,44] on OpenLane-V. We see that the proposed algorithm outperforms the existing techniques in most metrics. GANet [32] and CLRNet [41], which are recent image-based detectors, perform well in image metrics. But, their flickering rates R_F and missing rates R_M are relatively high. RVLD [11] yields the lowest R_F, but it is inferior to the proposed algorithm in other metrics. Specifically, the proposed algorithm outperforms RVLD by margins of 0.015, 0.011, and 0.005 in mIoU, F1, and R_M, respectively. Note that reducing R_M is more critical than reducing R_F because missing lanes consecutively at both frames represents the worst scenario in video lane detection. These experimental results indicate that the proposed algorithm is temporally more stable than RVLD.

Fig. 9 shows detection results. Image-based techniques inaccurately detect implied lanes or simply miss them in challenging scenes. RVLD is better than

Table 3: Ablation studies of the proposed algorithm on VIL-100.

	Obstacle mask	Memory	Synthetic data	F1	$\mathrm{R_F}$	$\mathrm{R_M}$
I		✓	✓	0.933	0.026	0.043
II	✓		✓	0.932	0.024	0.041
III	✓	✓		0.929	0.036	0.043
IV	✓	✓	✓	0.936	0.026	0.038

these detectors but underperforms for highly occluded lanes. In contrast, the proposed algorithm detects those lanes reliably.

4.5 Ablation Studies

We conduct ablation studies to analyze the efficacy of the proposed algorithm and its components. Table 3 compares several ablated methods on VIL-100. Method I detects lanes in a current frame I_t without exploiting the latent obstacle mask in the proposed OMR module. In other words, it excludes $\tilde{O}_t$ in (3). Method II does not use the memory information by removing the ConvLSTM block in the OMR module. Thus, Z directly becomes F_t in (5). In Method III, the OMR module is not trained using synthetically generated data. Method IV, the proposed algorithm, applies the OMR module along with the data augmentation scheme.

Efficacy of obstacle mask: As compared with the proposed algorithm (Method IV), Method I yields inferior scores in terms of F1 and $\mathrm{R_M}$. This indicates that utilizing obstacle masks is beneficial for accurate and temporally stable lane detection. Some detected obstacle masks are presented in the second column in Fig. 10.

Efficacy of memory information: Compared to Method IV, Method II yields slightly lower scores of $\mathrm{R_F}$, but its F1 score and missing rate $\mathrm{R_M}$ become worse. This indicates that, rather than using previous output only, it is more effective to exploit memory information for reliable lane detection.

Efficacy of synthetic training data: Without using synthetic data in Method III, the performances drop significantly in every metric, especially for $\mathrm{R_F}$ and $\mathrm{R_M}$. The synthetic data augmentation is helpful for enhancing the temporal stability of lanes.

Efficacy of OMR module: Fig. 10 visualizes the obstacle mask $\tilde{O}_t$, the feature map $\tilde{F}_t$ and the lane probability map $\tilde{P}_t$ of a current frame I_t, and their refined ones F_t and P_t. Some lane parts in I_t are occluded by nearby obstacles, and thus their features and probabilities are erroneous. These results, however, are restored faithfully through the proposed OMR module.

Runtime: Table 4 lists the runtime for each stage of the proposed algorithm. The processing speed is about 105 frames per second, surpassing RVLD [11]. RVLD demands high computational costs due to local correlation in the motion estimator. In contrast, the proposed algorithm consists of simpler operations. The key parts, the latent obstacle detection and OMR, take less time to process. The decoding part requires the longest time, containing the NMS process.

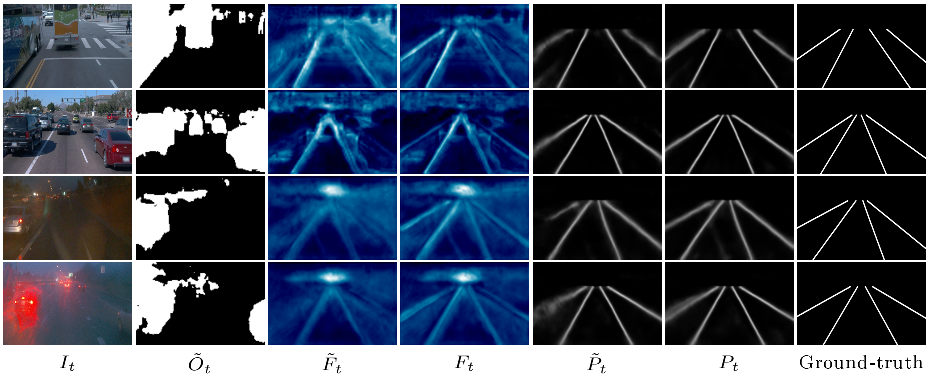


Fig. 10: Visualization of the obstacle mask $\tilde{O}_t$, the feature map $\tilde{F}_t$, and the probability map $\tilde{P}_t$, and their enhanced ones.

Table 4: Runtime analysis and comparison of the proposed algorithm with RVLD. LOD refers to latent obstacle detection. The processing times are reported in seconds per frame.

RVLD [11]	Proposed				
	Encoding	LOD	OMR	Decoding	Total
0.0125s	0.0026s	0.0002s	0.0010s	0.0057s	0.0095s

5 Conclusions

We proposed a novel video lane detector. First, the proposed algorithm extracts a feature map for a current frame and detects latent obstacles obstructing lane visibility. Then, it enhances the feature map using the occlusion-aware memory-based refinement (OMR) module, which takes the detected obstacle mask and the feature map from the current frame, the previous output, and the memory information as input. The enhanced feature map is used for more reliable lane detection. Moreover, we developed a data augmentation scheme for training the OMR module robustly. Experimental results demonstrated that the proposed algorithm outperforms existing techniques meaningfully.

Acknowledgements

This work was conducted by Center for Applied Research in Artificial Intelligence (CARAI) grant funded by DAPA and ADD (UD230017TD) and was supported by the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (No. NRF-2022R1A2B5B03002310 and No. RS-2024-00397293).

References

1. Aly, M.: Real time detection of lane markers in urban streets. In: Intelligent Vehicles Symposium (2008)
2. Chen, L., Sima, C., Li, Y., Zheng, Z., Xu, J., Geng, X., Li, H., He, C., Shi, J., Qiao, Y., Yan, J.: PersFormer: 3D lane detection via perspective transformer and the OpenLane benchmark. In: Proc. ECCV (2022)
3. Chen, Z., Liu, Q., Lian, C.: PointLaneNet: Efficient end-to-end CNNs for accurate real-time lane detection. In: Intelligent Vehicles Symposium (2019)
4. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The Cityscapes dataset for semantic urban scene understanding. In: Proc. IEEE CVPR (2016)
5. Feng, Z., Guo, S., Tan, X., Xu, K., Wang, M., Ma, L.: Rethinking efficient lane detection via curve modeling. In: Proc. IEEE CVPR (2022)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. IEEE CVPR (2016)
7. He, Y., Wang, H., Zhang, B.: Color-based road detection in urban traffic scenes. IEEE Trans. Intel. Transp. Syst. **5**(4), 309–318 (2004)
8. Hillel, A.B., Lerner, R., Levi, D., Raz, G.: Recent progresss in road and lane detection: A survey. Mach Vis. Appl. **25**(3), 727–745 (2014)
9. Hou, Y., Ma, Z., Liu, C., Hui, T.W., Loy, C.C.: Inter-region affinity distillation for road marking segmentation. In: Proc. IEEE CVPR (2020)
10. Hou, Y., Ma, Z., Liu, C., Loy, C.C.: Learning lightweight lane detection CNNs by self attention distillation. In: Proc. IEEE ICCV (2019)
11. Jin, D., Kim, D., Kim, C.S.: Recursive video lane detection. In: Proc. IEEE ICCV (2023)
12. Jin, D., Park, W., Jeong, S.G., Kwon, H., Kim, C.S.: Eigenlanes: Data-driven lane descriptors for structurally diverse lanes. In: Proc. IEEE CVPR (2022)
13. Li, X., Li, J., Hu, X., Yang, J.: Line-CNN: End-to-end traffic line detection with line proposal unit. IEEE Trans. Intel. Transp. Syst. **21**(1), 248–258 (2019)
14. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proc. IEEE CVPR (2017)
15. Liu, L., Chen, X., Zhu, S., Tan, P.: CondLaneNet: A top-to-down lane detection framework based on conditional convolution. In: Proc. IEEE ICCV (2021)
16. Liu, R., Yuan, Z., Liu, T., Xiong, Z.: End-to-end lane shape prediction with transformers. In: Proc. IEEE WACV (2021)
17. Liu, R., Yuan, Z., Liu, T., Xiong, Z.: End-to-end lane shape prediction with transformers. In: Proc. IEEE WACV (2021)
18. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: Proc. ICLR (2019)
19. Neven, D., De Brabandere, B., Georgoulis, S., Proesmans, M., Van Gool, L.: Towards end-to-end lane detection: An instance segmentation approach. In: Intelligent Vehicles Symposium (2018)
20. Oh, S.W., Lee, J.Y., Xu, N., Kim, S.J.: Video object segmentation using space-time memory networks. In: Proc. IEEE ICCV (2019)
21. Pan, X., Shi, J., Luo, P., Wang, X., Tang, X.: Spatial as deep: Spatial CNN for traffic scene understanding. In: Proc. AAAI (2018)
22. Qi, L., Jiang, L., Liu, S., Shen, X., Jia, J.: Amodal instance segmentation with KINS dataset. In: Proc. IEEE CVPR (2019)

23. Qin, Z., Wang, H., Li, X.: Ultra fast structure-aware deep lane detection. In: Proc. ECCV (2020)
24. Qiu, Z., Zhao, J., Sun, S.: MFIALane: Multiscale feature information aggregator network for lane detection. IEEE Trans. Intel. Transp. Syst. (2022)
25. Qu, Z., Jin, H., Zhou, Y., Yang, Z., Zhang, W.: Focus on local: Detecting lane marker from bottom up via key point. In: Proc. IEEE CVPR (2021)
26. Shi, X., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.c.: Convolutional LSTM Network: A machine learning approach for precipitation nowcasting. In: Proc. NeurIPS (2015)
27. Tabelini, L., Berriel, R., De Souza, A.F., Badue, C., Oliveira-Santos, T.: Lane marking detection and classification using spatial-temporal feature pooling. In: International Joint Conference on Neural Networks (2022)
28. Tabelini, L., Berriel, R., Paixao, T.M., Badue, C., De Souza, A.F., Oliveira-Santos, T.: Keep your eyes on the lane: Real-time attention-guided lane detection. In: Proc. IEEE CVPR (2021)
29. Tabelini, L., Berriel, R., Paixao, T.M., Badue, C., De Souza, A.F., Oliveira-Santos, T.: PolyLaneNet: Lane estimation via deep polynomial regression. In: Proc. IEEE ICPR (2021)
30. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Proc. NeurIPS (2017)
31. Wang, B., Wang, Z., Zhang, Y.: Polynomial regression network for variable-number lane detection. In: Proc. ECCV (2020)
32. Wang, J., Ma, Y., Huang, S., Hui, T., Wang, F., Qian, C., Zhang, T.: A keypoint-based global association network for lane detection. In: Proc. IEEE CVPR (2022)
33. Wang, M., Zhang, Y., Feng, W., Zhu, L., Wang, S.: Video instance lane detection via deep temporal and geometry consistency constraints. In: Proc. ACM Multimedia (2022)
34. Xiao, L., Li, X., Yang, S., Yang, W.: ADNet: Lane shape prediction via anchor decomposition. In: Proc. IEEE ICCV (2023)
35. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J.M., Luo, P.: SegFormer: Simple and efficient design for semantic segmentation with transformers. In: Proc. NeurIPS (2021)
36. Xu, H., Wang, S., Cai, X., Zhang, W., Liang, X., Li, Z.: CurveLane-NAS: Unifying lane-sensitive architecture search and adaptive point blending. In: Proc. ECCV (2020)
37. Xu, S., Cai, X., Zhao, B., Zhang, L., Xu, H., Fu, Y., Xue, X.: RCLane: Relay chain prediction for lane detection. In: Proc. ECCV (2022)
38. Zhang, J., Deng, T., Yan, F., Liu, W.: Lane detection model based on spatio-temporal network with double convolutional gated recurrent units. IEEE Trans. Intel. Transp. Syst. **23**(7), 6666–6678 (2021)
39. Zhang, Y., Zhu, L., Feng, W., Fu, H., Wang, M., Li, Q., Li, C., Wang, S.: VIL-100: A new dataset and a baseline model for video instance lane detection. In: Proc. IEEE ICCV (2021)
40. Zheng, T., Fang, H., Zhang, Y., Tang, W., Yang, Z., Liu, H., Cai, D.: RESA: Recurrent feature-shift aggregator for lane detection. In: Proc. AAAI (2021)
41. Zheng, T., Huang, Y., Liu, Y., Tang, W., Yang, Z., Cai, D., He, X.: CLRNet: Cross layer refinement network for lane detection. In: Proc. IEEE CVPR (2022)
42. Zhou, S., Jiang, Y., Xi, J., Gong, J., Xiong, G., Chen, H.: A novel lane detection based on geometrical model and Gabor filter. In: Intelligent Vehicles Symposium (2010)

43. Zhu, X., Hu, H., Lin, S., Dai, J.: Deformable ConvNets v2: More deformable, better results. In: Proc. IEEE CVPR (2019)
44. Zou, Q., Jiang, H., Dai, Q., Yue, Y., Chen, L., Wang, Q.: Robust lane detection from continuous driving scenes using deep neural networks. IEEE Trans. Veh. Technol. **69**(1), 41–54 (2019)