Adaptive Lane Point Extraction in Rain Conditions

Prabhu Shankar Mahendran, Boon-Hee Soong
School of Electrical and Electronic Engineering
Nanyang Technological University
Block S2.1, 50 Nanyang Avenue
Singapore 639798
prab0010@e.ntu.edu.sg, ebhsoong@ntu.edu.sg

Jian-Gang Wang
Institute for Infocomm Research
*Agency for Science, Technology and Research (A*STAR)*
1 Fusionopolis Way, #20-10, Connexis South Tower
Singapore 138632
jgwang@i2r.a-star.edu.sg

***Abstract*—The lane detection systems in ADAS and self-driving applications need to be resilient to adverse weather like rain, to minimize the chance of serious accidents. However, it is still challenging to maintain high precision in these conditions. State-of-the-art segmentation-based detectors use a naive post-processing method to quickly extract lane points from the segmented output based on the local maxima, but it is prone to failure when the segmentation contains errors. In this paper, we present an Adaptive Lane point EXtractor (ALEX) which overcomes this limitation by harnessing statistical properties from local regions of the segmentation to implicitly identify and compensate for errors. ALEX merges both local maximum and local mean statistics with CNN attributes to build a holistic hybrid feature set. Lane points are predicted from this representation by estimating their location on a reduced-size confidence map. A lateral offset is predicted to compute the precise location on the full-sized scene, while a class label prediction denotes which lane class the point belongs to. Besides the segmentation output and lane point ground truth, no additional information is required by ALEX, making it effectively agnostic to weather conditions and segmentation methods. Evaluation on two publicly available clear-weather databases and one rain-translated database verified that our approach consistently improved accuracy while reducing false positive and false negative rates, regardless of weather or segmentation method.**

***Index Terms*—Lane Point Extraction, Post-processing, Rainy Conditions, Segmentation-based Lane Detection**

I. Introduction

One of the core perceptual elements in Autonomous Vehicles (AV) and Advanced Driver Assistance Systems (ADAS) is the lane detection system. It ensures that decision making elements receive timely updates on the road and lane characteristics. This includes information on the number of lanes, the curvature of the road, and the position of the vehicle on the road. It is a challenging process to build good and commercially practical lane detection systems because the road lane images are affected by outdoor conditions, e.g. varying illumination, adverse weather, occlusion, etc. Numerous factors including robustness of the detector to noisy input, lane tracking capabilities, and low latency or real time processability, should be considered in the development process. In this paper, our aim is to improve lane detection performance in rain conditions through post-processing.

A typical lane detection framework comprises of a pre-processor, detector, and a post-processing module. The pre-processing module is tasked with enhancing critical features [1], [2] and attenuating noise [3], [4] from the raw image signal. The detector then predicts lane instances and their properties from the enhanced signal. Lastly, the post-processing module improves the precision of the detection output [5]. It may also include tracking elements to boost the continuity of the detection between sequential frames.

Lanes are commonly interpreted as a series of precise points in lane perception applications. Each point represents the centroid of the lane marker at some distance from the camera. The points may be directly generated by the detector, or extracted through post-processing means from segmentation output. In some applications, these points may be projected to world coordinates. However, regardless of whether projection is done, the lane trajectory would be extrapolated from the lane points. We refer to this process as lane construction. In this paper, we refer to the process of extracting lane points from the lane segmentation output and extrapolating the lane trajectory as lane construction. The constructed lanes are an important input for path planning and lane departure warning systems as they define the lane decision boundary [6]. For this reason, it is important to ensure that the extracted lane points are precise and reliable under a variety of environmental conditions.

Deep learning has been a popular approach for lane detection in recent years, owing to its industry-leading performance. Segmentation is one such deep learning method where lane pixels are discriminated from non-lane elements in the image feed. The segmented output is typically coarse and requires some post-processing to improve the precision of the detected lanes. A common post-processing approach is to pass a 2D Gaussian smoothing kernel over the segmented output and select points corresponding to the local maxima [7]. As it naively selects the local maxima, we refer to this approach as Naive Lane point Extraction (NLE).

A major drawback is that it would naively transfer errors from the segmentation output to the lane construction output. Challenging situations like poor illumination, rain, and night scenes can lead some lane segmentors to produce patchy or noisy output. In such patchy regions, NLE may be prone to missing some lane point. One solution to this is to use an adaptive lane point extractor, which can compensate for noise in the segmented output (Fig. 1). Furthermore, ensuring that the extractor can operate proficiently in both clear and adverse weather conditions would help to extend lane detection

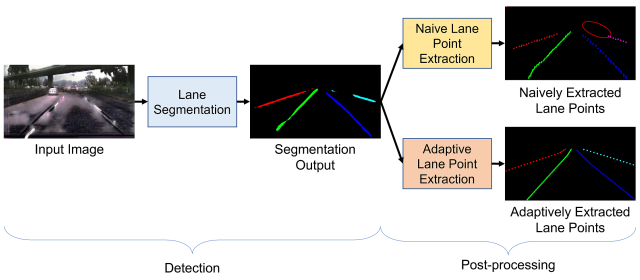

Fig. 1. Comparison of naive lane point extraction (NLE) with adaptive lane point extraction. NLE may carry false negatives from the segmentation output to the final post-processed output (circled in red). This can be rectified by using an adaptive post-processor.

frameworks to more environmental domains. In this paper, we address the abovementioned challenge of post-processing lane segmentation output by developing the adaptive lane point extractor (ALEX). The innovations of our work in this paper can be summarized as follows:

- We introduce Adaptive Lane point EXtractor (ALEX), a post-processing module for precise extraction of sparse lane points from lane segmentation maps. There is a high correlation in the positions of adjacent lane points. ALEX exploits this property by analysing the local statistics of the pixel confidences on the segmentation maps, including the peak and mean activation logits. Additional shallow CNN features are injected to the statistical properties to facilitate in compensating for errors made by the preceding lane detector. Lastly, careful regression of lane points ensure that at most one lane point is selected per row for each lane instance.
- Our study suggests that ALEX is weather-agnostic as it could produce improved results on both clear and rainy settings. In addition, the lane detection framework did not require sophisticated denoising algorithms to acclimate to rainy weather.
- Our method was thoroughly experimented in both rain and clear conditions, on commonly encountered driving scenes. Evaluation was performed on selected challenging scenes from the popular TuSimple [8] and CULane [9] databases. This includes artificially generated rain settings that were produced using established image-to-image translation technologies [16].

The rest of this paper is organized as follows: Section II reviews existing work on lane point extraction in lane detection applications. Next, Section III describes the developed method. Section IV then presents the experiments and results. Lastly, Section V concludes this paper.

II. Related Work

Lane detection has evolved from using simple intensity and edge-based hand-crafted features to harnessing powerful CNNs that are optimized to learn deep discriminative properties of the lane markings. Segmentation-based detectors produce coarse blobs to describe the identified lane elements. [5], [7], [9]–[12]. Lane construction is usually required during testing, to obtain the sparse lane points. This is commonly achieved using NLE as introduced above. In NLE, a 9×9 smoothing kernel is applied to reduce the coarse segmented blobs to a series of local maxima. Next, a lane point, which corresponds to the local maxima, is selected every 20 rows of pixels [7], [10]. The lane trajectory is then derived by fitting the selected lane points onto a cubic spline. This is frequently performed when evaluating on the TuSimple and CULane datasets [7], [10], [11].

Alternatively, EL-GAN [5] achieved higher precision in the lane segmentation maps by incorporating an adversarial-based embedding loss to the cost function. The resulting segmentation output is significantly finer and more precise. The map is then binarized to ensure that one lane point can be selected per row for each lane. This point typically corresponds to the mean x-index of the lane blob. The lane is then constructed by fitting a polyline on the selected points. Although simple and computationally cheap, the above methods are limited by their inability to adapt to noisy detection output. The smoothing kernel and point selection operations cannot compensate when sections of the lane are missed, such as in challenging scenarios like rainy weather.

Several groups have also proposed bypassing segmentation and directly regressing the lane points [13]–[15]. Qin [14] predicts lane points along the preselected rows of the image, while including an auxiliary segmentation task during training. A single lane point is then predicted per lane instance per anchor row during inference. However, as segmentation is leveraged for detecting lane elements, it would also be susceptible to missing lane points due to false negative segments. In Ko's approach [13], knowledge from a larger network is distilled to a clipped model. Similar to most object detectors, lane points are regressed using a combination of a confidence score, an offset, and a feature embedding loss. Nevertheless, the precision of this method is dependent on the size of the output grid. As such, its precision may diminish if the grid size is too small.

From the above review, it is evident that segmentation-based methods can still be prone to generating noisy output in some complex scenarios. Primarily, lane points from missed lane segments cannot be effectively recovered using the abovementioned methods. Furthermore, false positive detections can also contribute to false points or displacing of the predicted lane point from its true position. In view of this, an adaptive postprocessing module may greatly benefit the present segmentation methods by compensating for both false positive and false negative elements from the segmented output. Furthermore, it would extend the capabilities of these methods to more external conditions.

III. Our Approach

Most segmentation-based lane detection methods are still prone to noise present in challenging conditions. Factors such as large variations in illumination, shadows, rain, and occlusion from adjacent vehicles frequently result in either

missed lane segments, or false positives. In these situations, NLE may lead to further loss of detected lanes or propagation of the falsely detected regions.

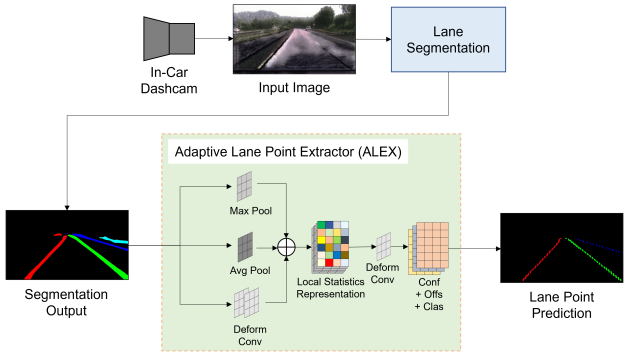

Fig. 2. Overview of ALEX in the lane detection framework. After segmenting lanes from the raw image, ALEX extracts shallow features and maps of the local maxima and local mean from the segmented output. These maps are fused to generate a local statistical representation of the lanes, which are then used to regress the confidence (Conf.), horizontal offsets (Offs.), and lane instance (Clas.) of the lane points.

A. Overview of ALEX

In this paper, the developed adaptive lane point extractor overcomes this by assessing the activation statistics of segmented pixels within local regions. An overview of ALEX and its placement in the lane detection framework is presented in Fig. 2. The size of the local region assessed is based on the typical width of segmented lanes. Following the TuSimple and CULane databases, most segmented lanes can be up to 30 pixels wide at each pixel row. In addition, we also consider the vertical displacement between lane points, which is related to the distance travelled by a vehicle between frames. Assuming an average frame rate of 25 Frames per second (FPS) and considering a vehicle speed of $60\,\mathrm{km/h}$, the distance travelled by the vehicle between frames is approximately 0.6 to 1 meter. On the image plane, this translates to about one point for every 5 to 10 pixel rows. Based on these properties, ALEX first dissects the segmentation map into grid cells of size 5×25. This ensures that at most 1 lane point would be predicted per lane every 5 rows of pixels. The larger grid width helps to accommodate processing of thicker lane segments.

B. Local Statistics Representation

The local statistics representation for each grid cell are produced using a combination of pooling and convolutional operations. ALEX utilizes Max and Average pooling operations to obtain the local maximum and local mean properties of each grid cell. The local maximum is used to locate the lane pixels at each row. In normal weather conditions, this information may be sufficient for identifying the lane point, which is identical to the procedure used in NLE. However, in rainy weather, the detection signal may be noisy and contain missing or false positive segments. The local mean information encodes the mean activation logits of surrounding pixels, and would be higher or lower depending on the presence of false positives and false negatives in that region, respectively. This property helps with identifying if the local region is affected by environmental noise. It also helps to tune and regulate the compensatory effect of ALEX. This property is not actively used in the NLE method, making it incapable of correcting errors from the lane detector.

A square kernel of size 25×25 is used by both pooling operations, while the horizontal and vertical strides of the operations are set to 25 and 5 respectively. This minimizes the number of operations along the horizontal axis, while maintaining a sufficiently large receptive field width. The vertical stride is made smaller to ensure that activation statistics between vertically adjacent grid cells can be encoded within each grid cell's intermediate features. This helps to provide more evidence that a lane exists at the grid cell's location, which is used to compensate for missed segments from the prior detection stage.

A series of two consecutive deformable convolution [17] operations are also performed to extract other local activation attributes. Both convolution operations utilize a 5×5 kernel size, thus achieving a relative receptive field size of 25×25. However, deformable convolution includes tunables offsets at each bin location of the kernel, which can vary the receptive field to suit the data. As a result, the true receptive field size may effectively be larger than 25×25. The convolution operations is also more sensitive to high-confidence lane segments. The local statistics representation is then built by concatenating the output of the pooling and convolutional operations along the depth of the tensor.

C. Generating Lane Point Predictions

Lane points are estimated from the local statistics representation by passing them through a third deformable convolution operation. This convolution operation encodes the three components used to define the lane point: 1) the lane point confidence (Conf.), 2) the horizontal offset (Offs.) of the point from the center of the grid cell, and 3) the lane instance (Clas.) the point corresponds to.

To generate lane points, the lane point locations are first identified by pixels on the Conf. map with a confidence value ($Conf(x, y)$) of at least 0.5. All other locations are discarded. Next, the lane class of the point (c) is extracted by referencing the indices of selected pixel regions in the Clas. map. A maximum of 5 lanes instances may be identified if ALEX is optimized on the TuSimple dataset. An additional background class is also included to facilitate training, however this is discarded during inference. Following this, the horizontal offset of the selected lane point ($Offs(x, y, c)$) is extracted from the Offs. map. The precise location of the lane point is derived by adding the horizontal offset to the horizontal index (x) of the point, and multiplying this value to the horizontal factor (G_x). The coordinates of the generated lane points are computed based on equations (1) and (2).

$$Coord_{x,gen}(x,y,c) = \begin{array}{l} G_x * (Coord_x(Clas(x,y,c)) \\ \quad + Offs(x,y,c)) \\ \text{if Conf(x,y)} \geq 0.5 \text{ and} \\ \text{Clas(x,y,c)} \geq 0.5 \end{array} \quad (1)$$

$$Coord_{y,gen}(x,y,c) = \begin{array}{l} G_y * (Coord_y(Clas(x,y,c))) \\ \text{if Conf(x,y)} \geq 0.5 \text{ and} \\ \text{Clas(x,y,c)} \geq 0.5 \end{array} \quad (2)$$

In these expression, G_x and G_y denote the size of the grid cell, which are fixed at 25 and 5 respectively. $Coord(.)$ represents the x or y coordinate on the output from the proposed model. $Coord_{gen}(.)$ refers to the generated coordinate in the input resolution. $Clas(x,y,c)$ and $Offs(x,y,c)$ are the class confidence and horizontal offset factor at pixel (x,y) for lane c, respectively. Lastly, $Conf(x,y)$ is the lane confidence score at pixel location (x,y).

D. Optimization Functions

ALEX is optimized using three losses that target the confidence, offset, and classification output. The confidence loss $\mathcal{L}_{Conf}$ is used to regress the lane point's location on the size-reduced map. It has separate components for lane and non-lane pixels, and uses a normalized squared error loss that is averaged over the total number of lane and non-lane points separately. Points further from true lane points were observed to converge the fastest. In addition, non-lane points that were very close to true lane points were the easiest to misclassify. An additional squared component is added to the non-lane component of $\mathcal{L}_{Conf}$ to compensate for this. $\mathcal{L}_{Conf}$ is displayed in equation (3).

$$\mathcal{L}_{Conf} = \begin{cases} \frac{1}{N_{lp}} \sum_{i=1}^{W} \sum_{j=1}^{H} (\hat{C_{ij}} - C_{ij})^2 & \text{if } \hat{C_{ij}} = 1 \\ \frac{1}{N_{bg}} \sum_{i=1}^{W} \sum_{j=1}^{H} (\hat{C_{ij}} - C_{ij})^2 & \\ + \frac{1}{10000} \times \sum_{i=1}^{W} \sum_{j=1}^{H} C_{ij}^2 & \text{otherwise} \end{cases} \quad (3)$$

$\hat{C_{ij}}$ and C_{ij} represent the ground truth and predicted lane point confidences at the coordinates (i,j), while N_{lp} and N_{bg} denote the number of lane point and non-lane point elements in the batch respectively.

Each pixel from the output grid represents a local window of the segmentation map consisting of 25 width pixels and 5 height pixels. An offset component is used to identify the horizontal offset of the lane point from the leftmost part of the window. The lane point is fixed on the central row of the local window, and does not require a vertical offset component. Non-lane points do not contribute to the lane construction, and are thus, ignored during training. The lateral offset X_{ij} is regressed using a normalized squared error loss as follows:

$$\mathcal{L}_{Offs} = \frac{1}{N_{lp}} \sum_{i=1}^{W} \sum_{j=1}^{H} (\hat{X_{ij}} - X_{ij})^2 \quad (4)$$

Here, W and H represent the width and height of the size-reduced output tensor respectively. $\hat{X_{ij}}$ is the ground truth lateral offset.

The lane point is also assigned to one of the available lane instances. The Softmax function is used to simulate a probability distribution of each pixel's class over the total number of lane classes N_{Lanes}. In this study, we used 1 background and 5 lane classes. The classification loss $\mathcal{L}_{Clas}$ is based on the standard cross-entropy loss, as depicted in equation (5).

$$\mathcal{L}_{Clas} = -\sum_{i=0}^{N_{Lanes}} \left[\hat{l_i} \times log \left(\frac{exp(l_i)}{\sum_{j=1}^{N_{Lanes}} exp(l_j)} \right) \right] \quad (5)$$

where, $\hat{l}$ and l represent the ground truth and predicted lane labels respectively.

The final cost function (equation (6)) is a weighted sum of the three loss components. The multipliers λ_1, λ_2, and λ_3, are all initialized as 1. However, λ_2 which corresponds to the offset loss $\mathcal{L}_{Offs}$, is increased to 1.5 after 50 epochs.

$$\mathcal{L}_{Total} = \lambda_1 \mathcal{L}_{Conf} + \lambda_2 \mathcal{L}_{Offs} + \lambda_3 \mathcal{L}_{Clas} \quad (6)$$

IV. Experiments and Results

In order to verify our approach proposed in this paper, we have evaluated our algorithm on some publicly available databases. The comparison with the state-of-the-art lane segmentation have shown that our method can achieve better accuracy and low false rate on large databases in rain conditions. This section documents the evaluation of the developed extractor module. Subsection A. describes all the databases used in this paper. Following this, Sections B. and C. describe the experimental settings and results obtained.

A. Databases

TuSimple: The TuSimple database features mainly highway scenes in clear and bright weather. However, it also contains scenes with vehicular occlusion of lanes and shadows. TuSimple contains a total of 6,498 images of which 3,626 are used for training and validation, while 2,782 are reserved for testing. Evaluation on TuSimple uses the lane accuracy metric, $Acc = N_{pd}/N_{gt}$. N_{pd} refers to the number of correctly predicted lanes, while N_{gt} is the actual number lanes. Besides this, the statistics of False Positive (FP) and False Negative (FN) are also computed.

CULane: CULane is a larger database that features a larger variety of challenging settings. This includes converging and diverging lanes, dazzling lights, and night scenes. A total of 88,880 images are used for training, while 34,680 images are kept for testing. The CULane evaluation first involves computing the Intersection-over-Union (IoU) between the predicted and ground truth maps. Predicted regions with an IoU greater than 0.5 are deemed True Positives (TP). The metric used is the F1 score which is computed following equation (9), which

TABLE I
DATABASE PROPERTIES

Database	#Frames	Train	Val	Test	Resolution	#Lanes	#Scenarios	Scene Type	Weather
TuSimple-Rain	6,408	3,268	358	2,782	1280×720	≤ 5	1	Highway	Rain
TuSimple [8]	6,408	3,268	358	2,782	1280×720	≤ 5	1	Highway	Clear
CULane [9]	133,235	88,880	9,675	34,680	1640×590	≤ 4	9	Urban and Highway	Clear
RainSG	4,003	-	-	4,003	1920×1080	≤ 5	≥ 6	Urban and Highway	Rain

is based on the Precision (equation (7)) and Recall (equation (8)) components.

$$Precision = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FP_i} \tag{7}$$

$$Recall = \frac{1}{N}\sum_{i=1}^{N}\frac{TP_i}{TP_i + FN_i} \tag{8}$$

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{9}$$

TuSimple-Rain: As both TuSimple and CULane mainly feature clear weather conditions, we generated an additional rainy weather setting, namely TuSimple-Rain, to evaluate the weather-agnostic capabilities of the developed module. The rain dataset is generated using the CycleGAN [16] image-to-image translation, with real rain images collected on Singapore roads used to represent the target style domain.

CycleGAN is an unsupervised solution for bidirectional style transfer which uses a system of two GANs. It gains its unsupervised learning capabilities by coupling this two-GAN framework with cycle consistency constraints to ensure that each GAN system is optimized to generate and audit translations in only one of the two directions. Through this, CycleGAN has achieved remarkable success in style transfer applications, which we harness for dataset generation.

The generated TuSimple-Rain database is identical in size and context to its parent database but features heavy rain artefacts. Ground truth information from TuSimple was used in conjunction with the rain-translated images. We present both quantitative and qualitative results for TuSimple-Rain. Table I summarizes the databases used in this articles, where we identify the set of real rain images used to generate TuSimple-Rain as RainSG. RainSG consists of 4,003 frames from more than 6 scenarios. It embodies a diverse range of rain intensities from light showers ($< 2.54mm/Hr$) to heavy downpour ($> 7.62mm/Hr$), and was recorded at a resolution of 1920×1080 using a GARMIN full HD dashcam.

B. Experimental Settings

As ALEX only uses raw lane segmentation maps for generating lane points, lane segmentation maps must first be obtained. ENet-SAD [7] and JESNet [18] are two methods selected to generate these maps on the TuSimple and TuSimple-Rain databases. We use these to compare the lane point estimation results between two different detection methods. ENet-SAD [7] is a forerunner in segmentation-based lane detection, and incorporates distilled self-attention to improve lane detection. On the other hand, JESNet [18] is a multi-task network of our design which harnesses weather-agnostic lane enhancement to boost lane detection in challenging conditions. JESNet is still under development and unpublished at the time of writing of this paper. Comparison on the CULane database is also made against prominent lane detectors.

By design, ENet-SAD requires TuSimple and TuSimple-Rain datasets to be resized to 368×640, while CULane is reduced to 288×800. JESNet uses a reduced input resolution of 352×1024 for all datasets. Nevertheless, the segmentation maps were resized to their original resolutions of 720×1280 for TuSimple and TuSimple-Rain, and 592×1640 for CULane, to match their respective ground truth resolutions. It is worth noting that the TuSimple and TuSimple-Rain databases feature up to 5 lane classes. However, the 5th lane class is not included during evaluation as it contains too few samples.

ALEX is optimized using the Adam optimizer, and uses a fixed learning rate of 0.0001. Regularization is also incorporated via spatial dropout and mixed precision training. Additionally, all experiments were conducted on a GTX TITAN X graphic card with 12 GB memory. Results from this study are presented in the next subsection.

C. Results

The quantitative results for TuSimple, TuSimple-Rain, and CULane are summarized in Tables II, III, and IV respectively. The comparison in both Table II, and Table III is between the TuSimple metrics obtained from using the existing NLE technique and that obtained with ALEX. This is performed for both the ENet-SAD and JESNet lane detectors. For the TuSimple database, ALEX reduced the FP rate for ENET-SAD from 0.0602 to 0.0511, as well as the FN rate from 0.0205 to 0.0189. The accuracy also increased from 0.9664 to 0.09678. Using ALEX, JESNet also achieved lower FP and FN rates of 0.0423 and 0.0258, respectively, and a 1.85% increase in accuracy.

A consistent reduction of FP and FN rates were also observed when ALEX was used on the TuSimple-Rain database. On TuSimple-Rain, ENet-SAD had higher base FP and FN rates of 0.3161 and 0.3319, respectively, as a result of interference contributed by rain artefacts. ALEX helped reduce these metrics to 0.2053 and 0.2232 for FP and FN, respectively. The network's accuracy also improved by 4.46% with ALEX. The rain artefacts also increased the base FP and FN rates for JESNet by 192% and 259%, respectively. Nevertheless, ALEX contributed to a reduction of the FP and FN rates on

TuSimple-Rain by 19.2% and 19.3%, respectively. We also note the increase in detection accuracy from 0.8829 to 0.9050. This consistency in improving the FP, FN, and Accuracy statistics for both clear and rain settings is a clear depiction of the weather-agnostic nature of ALEX. The same consistent improvements were also observed for both detectors, regardless of their approach. This indicates that ALEX is also model-agnostic, and can be integrated with different detection methods.

On the CULane database, we compared our detection approach ($JESNet + ALEX$) against published results from state-of-the-art methods. Table IV depicts the F1 scores for each test category in CULane, with the exception of the "Crossroad" class, which does not contain true positive lanes. Instead, the FP rates are reported for this category. Our method achieved comparable results against the state-of-the-art and produced improved F1 rates in the "Crowded", "Night", "No Line", "Dazzle Light", and "Crossroad" categories. We note that these are among the more challenging conditions encountered during driving. The overall score of our method of 72.2 also surpassed the overall scores of ENet-SAD [7] and SCNN [9], which were reportedly 70.8 and 71.6, respectively. In addition, our approach achieved a modest frame rate of 10.3 FPS on the GTX TITAN X. This is significantly faster than SCNN (7.5 FPS), but loses out to ENet-SAD and UFLDv2, which were both specifically designed to minimize computational costs. We attribute our modest frame rate to using a combination of smaller networks, mixed precision training, and carefully placed strided operations.

Qualitative results using ALEX and NLE on the TuSimple-Rain database are presented in Figs 3 and 4, respectively. In both figures, each lane is segmented using JESNet and depicted in green. The ground truth lane points are highlighted in blue, while the predicted lane points are in red. It is clear that the lane segmentation contains numerous FP and FN regions. Nevertheless, ALEX has shown to effectively avoid the FP regions, and produce lane points that are almost consistent with the ground truth (Fig 3). In comparison, the NLE method appears to miss detecting Lane 1, and wrongly generate lane points for Lane 3 at the FP regions (Circled in red in Fig 4). This clearly demonstrates the limitation of NLE with noisy segmentation output, and the advantages ALEX offers.

Analysis of the results indicates that ALEX can enhance the detection output, although its performance is contingent on the quality of the segmentation output it receives. Notably, ALEX exhibits an advantage in rectifying minor segmentation errors, such as the false positive segment of Lane 3 in Fig. 4. This capability stems from the fact that the majority of the segmented output for Lane 3 still aligns with the true lane location, thereby providing strong evidence of the lane's presence in that area.

We observed that ALEX's compensatory effect has its limitations, particularly when there is a substantial deviation between the segmented output and the actual lane regions. In cases where a significant portion of the lane is missed during the detection stage, ALEX may generate lane points sub-optimally. Additionally, if false positives from the segmentation output have high confidence scores, ALEX might produce false lane points in those regions. In such scenarios, segmentation errors can sufficiently taint the local statistical representation, leading to errors propagating through the post-processing. These limitations underscore the need for the lane detection framework to rely not solely on ALEX but also on reasonably accurate output from the segmentation network to achieve robust lane detection performance as a whole.

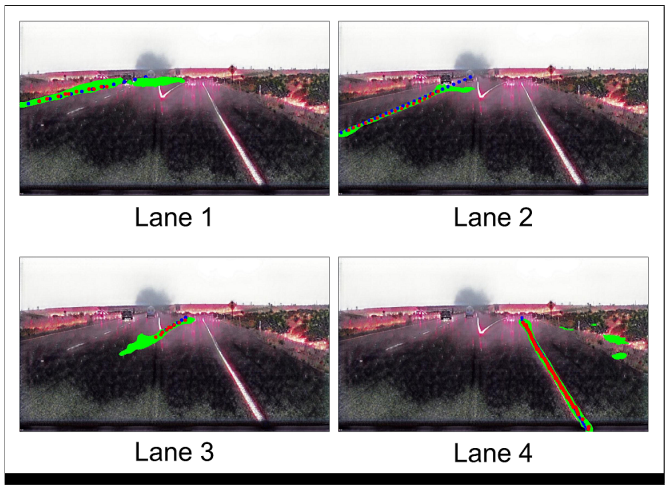

Fig. 3. Lane point predictions (red) using ALEX on a diverging-lane scene from the TuSimple-Rain database. Ground truths are colored blue, while the segmented lane is highlighted in green.

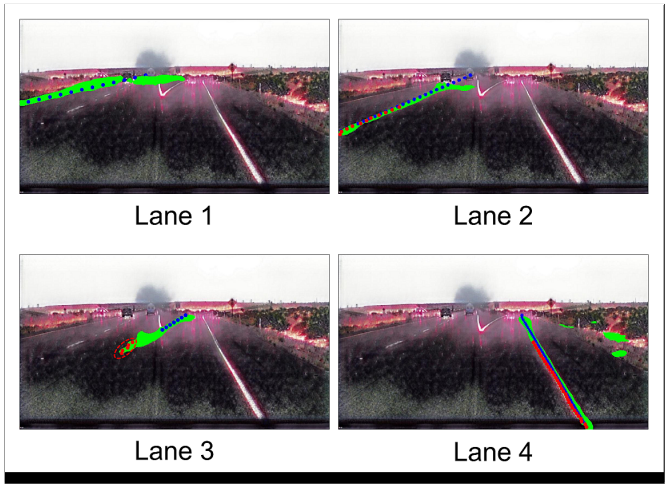

Fig. 4. Lane point predictions (red) using NLE on the same diverging-lane scene from the TuSimple-Rain database. Ground truth are colored blue, while the segmented lane is highlighted in green. False lane points generated by NLE (Lane 3) are circled in red, while a whole lane (Lane 1) was missed.

V. Conclusion

Segmentation-based lane detectors are often challenged by adverse weather like rain, resulting in more false detections. We address this by developing Adaptive Lane point EXtractor

TABLE II
EVALUATION RESULTS ON THE TUSIMPLE.

Method	Accuracy	False Positive (FP)	False Negative (FN)
ENet-SAD [7] + NLE	0.9664	0.0602	0.0205
ENet-SAD [7] + ALEX	**0.9678**	**0.0511**	**0.0189**
JESNet [18] + NLE	0.9489	0.0731	0.0441
JESNet [18] + ALEX	**0.9665**	**0.0423**	**0.0258**

TABLE III
EVALUATION RESULTS ON THE TUSIMPLE-RAIN.

Method	Acc.	FP	FN
ENet-SAD [7] + NLE	0.8560	0.3161	0.3319
ENet-SAD [7] + ALEX	**0.8942**	**0.2053**	**0.2232**
JESNet [18] + NLE	0.8829	0.2135	0.2337
JESNet [18] + ALEX	**0.9050**	**0.1725**	**0.1909**

(ALEX) as a post-processing module to compensate for minor irregularities in the output of lane segmentation networks in rainy weather. Unlike the current Naive Lane Point Extraction (NLE) method, which relies solely on local maxima for point extraction, ALEX utilizes a combination of local maxima, local mean, and shallow CNN features derived from the segmentation map logits. This approach generates a statistically-driven feature representation of the lanes. As a result, ALEX can effectively and adaptively suppress some false positive and false negative predictions from the segmentation output, mitigating the issues that NLE might propagate. Evaluation of ALEX on clear-weather and rain-translated databases demonstrated improved results, reaffirming ALEX's versatility across different weather conditions. However, ALEX's impact may be limited in cases of very poor segmentation quality. For instance, if a significant portion of the lane is missing in the segmentation output, the corresponding lane points may still be missed. Similarly, false lane points may be generated if their respective false positive segments exhibit excessively high confidence levels. Looking ahead, we consider exploring point tracking capabilities to address these limitations. Additionally, we look to investigate the incorporation of higher-order statistics into the local statistical representation to further enhance ALEX's performance.

REFERENCES

[1] J. -G. Wang, K. -W. Wan, C. -H. Pang and W. -Y. Yau, "Semi-supervised image-to-image translation for lane detection in rain," 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 2022, pp. 118-123, doi: 10.1109/ITSC55140.2022.9922367.

[2] T. Liu, Z. Chen, Y. Yang, Z. Wu and H. Li, "Lane detection in low-light conditions using an efficient data enhancement: Light conditions style transfer," 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, USA, 2020, pp. 1394-1399, doi: 10.1109/IV47402.2020.9304613.

[3] Y. Lee, J. Jeon, Y. Ko, B. Jeon and M. Jeon, "Task-driven deep image enhancement network for autonomous driving in bad weather," 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 2021, pp. 13746-13753, doi: 10.1109/ICRA48506.2021.9561076.

[4] O. Bailo, S. Lee, F. Rameau, J. S. Yoon and I. S. Kweon, "Robust road marking detection and recognition using density-based grouping and machine learning techniques," 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA, 2017, pp. 760-768, doi: 10.1109/WACV.2017.90.

[5] M. Ghafoorian, C. Nugteren, N. Baka, O. Booij, and M. Hofmann, "EL-GAN: Embedding loss driven generative adversarial networks for lane detection," In: Leal-Taixé, L., Roth, S. (eds) Computer Vision – ECCV 2018 Workshops, ECCV 2018, Lecture Notes in Computer Science(), vol 11129. Springer, Cham. https://doi.org/10.1007/978-3-030-11009-3_15.

[6] M. Fu, W. Song, Y. Yi and M. Wang, "Path planning and decision making for autonomous vehicle in urban environment," 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Gran Canaria, Spain, 2015, pp. 686-692, doi: 10.1109/ITSC.2015.117.

[7] Y. Hou, Z. Ma, C. Liu and C. C. Loy, "Learning lightweight lane detection CNNs by self attention distillation," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 1013-1021, doi: 10.1109/ICCV.2019.00110.

[8] TuSimple. Tusimple lane detection challenge. 2017. URL https://github.com/TuSimple/tusimplebenchmark/tree/master/doc/lane_detection.

[9] X. Pan, J. Shi, P. Luo, X. Wang, and X. Tang, "Spatial as deep: Spatial cnn for traffic scene understanding," In Conference on Artificial Intelligence (AAAI), February 2018.

[10] Y. Hou, Z. Ma, C. Liu, T. -W. Hui and C. C. Loy, "Inter-region affinity distillation for road marking segmentation," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 12483-12492, doi: 10.1109/CVPR42600.2020.01250.

[11] M. Lee, J. Lee, D. Lee, W. Kim, S. Hwang and S. Lee, "Robust lane detection via expanded self attention," 2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2022, pp. 1949-1958, doi: 10.1109/WACV51458.2022.00201.

[12] H. Xu, S. Wang, X. Cai, W. Zhang, X. Liang, Z. Li, "CurveLane-NAS: Unifying lane-sensitive architecture search and adaptive point blending." In Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12360. Springer, Cham. https://doi.org/10.1007/978-3-030-58555-6_41

[13] Y. Ko, Y. Lee, S. Azam, F. Munir, M. Jeon and W. Pedrycz, "Key points estimation and point instance segmentation approach for lane detection," in IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 7, pp. 8949-8958, July 2022, doi: 10.1109/TITS.2021.3088488.

[14] Z. Qin, H. Wang, and x. Li, "Ultra fast structure-aware deep lane detection," in European Conference on Computer Vision (ECCV), vol 12369. Springer, 2020. https://doi.org/10.1007/978-3-030-58586-0_17

[15] Z. Qu, H. Jin, Y. Zhou, Z. Yang and W. Zhang, "Focus on local: Detecting lane marker from bottom-up via key point," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 2021, pp. 14117-14125, doi: 10.1109/CVPR46437.2021.01390.

[16] J. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," In Proceedings of the IEEE International Conference on Computer Vision, pp. 2223–2232, 2017.

[17] J. Dai et al, "Deformable convolutional networks". CoRR, abs/1703.06211,2017. URL http://arxiv.org/abs/1703.06211.

[18] P. S. Mahendran, J.-G. Wang, and B.-H. Soong, "Joint enhancement and segmentation network for lane detection in rainy conditions" unpublished.

[19] T. Zheng, H. Fang, Y. Zhang, W. Tang, Z. Yang, H. Liu, and D. Cai, "RESA: Recurrent Feature-Shift Aggregator for Lane Detection," In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 4, pp. 3547-3554, May 2021, doi: 10.1609/aaai.v35i4.16469.

[20] Z. Qin, P. Zhang and X. Li, "Ultra Fast Deep Lane Detection With Hybrid Anchor Driven Ordinal Classification," in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2022, doi: 10.1109/TPAMI.2022.3182097.

TABLE IV
COMPARISON OF RESULTS ON CULANE (F1-SCORE, %).

Category	Proportion, %	**JESNet [18] + ALEX**	ENet-SAD [7]	SCNN [9]	RESA-34 [19]	UFLDv2 (RESNet-18) [20]
Normal	27.7	90.4	90.1	**90.6**	**91.9**	91.8
Crowded	23.4	70.3	68.8	69.7	72.4	**73.3**
Night	20.3	66.7	66.0	66.1	69.8	**70.7**
No Line	11.7	43.8	41.6	43.4	46.3	**47.6**
Shadow	2.7	66.2	65.9	66.9	72.0	**75.1**
Arrow	2.6	83.1	84.0	84.1	**88.1**	87.9
Dazzle Light	1.4	60.3	60.2	58.5	**66.5**	65.3
Curve	1.2	64.7	65.7	64.4	**68.6**	68.5
Crossroad	9.0	**1893**	1998	1990	1896	2075
Total	-	72.2	70.8	71.6	74.5	**75.0**
Graphic Card	-	GTX TITAN X	GTX TITAN X	GTX TITAN X	RTX 2080Ti (× 4)	RTX 3090
Frame rate (FPS)	-	10.3	74.6	7.5	45.5	**310**