ELaneNet: Using Lane Parameters for Better Detection of Lanes in Autonomous Driving Systems

Elikem Buertey
Dept. of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada
email: ebuertey@uwaterloo.ca

Kshirasagar Naik
Dept. of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada
email: snaik@uwaterloo.ca

Nitin Naik
School of Computer Science and Digital Technologies
Aston University
Birmingham, United Kingdom
email: n.naik1@aston.ac.uk

Sriram Sivaraman
Dept. of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada
email: s36sivaraman@uwaterloo.ca

***Abstract*—Lane detection is crucial for an Autonomous Driving System (ADS). While traditional lane detection methods have limitations, machine learning has shown promise, though many deep learning networks struggle with variable lane detection. High Definition (HD) Maps provide comprehensive road information but are expensive and inflexible. This research proposes eLaneNet, a flexible, cost-effective, and robust lane detection system that adapts to diverse driving scenarios. By incorporating the number of lanes into the network, we demonstrate improved adaptability and potential advancements in autonomous driving technologies. We also introduce new evaluation metrics, namely, capacity, lost capacity and unsafe driving measure to assess lane detection techniques more comprehensively. We also propose evaluation of lane detection techniques by using a lane abstraction approach instead of the traditional line abstraction method. Through extensive evaluation and comparisons, we showcase the superiority of eLaneNet over LaneNet in detecting lanes. Using the TuSimple dataset, we show that eLaneNet performs better than LaneNet in detecting lanes. This research contributes to bridging the gap between ML techniques and HD maps, offering a viable solution for effective and efficient lane detection in an ADS.**

***Keywords*—*Convolutional Neural Network*; *Enhanced LaneNet; lane detection*; *ELaneNet*; *semantic segmentation*; *autonomous driving*; *knowledge guided machine learning*.**

I. Introduction

Astonishingly, 94% of road accidents are caused by human error, highlighting the potential for significant reduction in human error [1]. Addressing this issue, an Autonomous Driving System (ADS) emerges as a possible solution to decrease human error. The anticipated benefits of autonomous vehicles include crash prevention, reduced travel times, improved fuel efficiency, and parking benefits, with estimated savings of up to $2000 per year per autonomous vehicle and potentially reaching nearly $4000 when considering comprehensive crash costs [2].

Lane detection is a crucial vision problem in the context of autonomous vehicles and Advanced Driver Assistance Systems (ADAS). It involves the identification and tracking of lane markings on the road to determine the vehicle's position within its lane. In order to ensure safe and accurate travel on roads and highways, autonomous driving systems heavily depend on accurately detecting lane lines. The ultimate aim is to achieve accuracy in locating and tracking lane markings, even in challenging environmental conditions.

There is no shortage of methods, which have been suggested for lane detection [3]–[5]. Traditional methods which rely on handcrafted features and heuristics were initially proposed [6]. In challenging scenarios, including adverse weather conditions, occlusions from other vehicles, and complex urban road networks, traditional lane detection techniques often fail, highlighting the imperative for robust and adaptable solutions as emphasized in [6].

With the advent of machine learning and deep learning techniques, researchers have explored their application to the lane detection problem. However, numerous existing models have limitations when it comes to detecting an arbitrary number of lanes. This is because they are typically designed to detect a maximum of 'n' lanes, where 'n' is a specific, predefined number. Although deep learning networks such as LaneNet [6], can detect arbitrary number of lanes, there is room for improvement in this area.

Another approach to solving lane detection involves the use of HD maps. The authors in study [7] define an HD map as a map which contains all critical static properties (for example, roads, buildings, traffic lights, and road markings) of the road/environment necessary for autonomous driving, including the object that sensors cannot appropriately detect due to occlusion. HD maps, while an attractive solution,

 ISBN: 978-1-68558-140-4

are not entirely free of challenges. In particular, there is a prohibitive cost associated with producing and modifying HD maps.

In this paper, we seek to enhance lane detection accuracy by leveraging additional data while avoiding the prohibitive costs associated with HD maps. Ultimately, we aim to create a robust and adaptable lane detection system that can effectively navigate complex driving scenarios and contribute to the advancement of autonomous driving technologies.

To establish a reliable baseline for lane detection experiments, LaneNet [6] was chosen. State-of-the-art lane detection networks, such as LaneNet, take only the driving scene as input and produce an output representing the detected lane lines. However, other information on road systems is usually readily available. For example, some governments have information on road width and the number of lanes on the road on their website. We decided to explore the impact of readily available information, such as the number of lanes (NoL), on the impact of lane detection networks. We chose the NoL as the input parameter because the NoL on the road is usually constant for a long time. To do this, we modified an existing lane detection network, LaneNet, to incorporate the input image and the number of lanes on the road. We call this modified network eLaneNet. LaneNet and eLaneNet were evaluated on the TuSimple dataset to compare their performance. New metrics were also introduced to evaluate the performance of lane detection networks because of the limitations of existing metrics.

The contributions introduced in this paper are summarized as follows:

(C1) By integrating the NoL associated with a driving scene into the lane detection process, we propose eLaneNet as a new approach to address lane detection challenges.

(C2) Due to the limitations of conventional line abstracting methods, we propose that a lane abstracting method should be used to assess the performance of lane detection algorithms. In addition, we introduce capacity, lost capacity and unsafe driving measures as performance metrics since they are more specific to lane detection than general metrics, such as recall.

Experiments comparing eLaneNet to LaneNet across metrics such as capacity, lost capacity, unsafe driving measure, and accuracy consistently showed eLaneNet's superior effectiveness. This was observed in both the lane and line abstraction approaches, where entities for metric calculations were lanes and lane lines, respectively.

The structure of this paper is outlined as follows. Section II provides a review of related work.In Section III, we discuss the original LaneNet implementation and then delve into its enhanced version, eLaneNet, explaining the introduced modifications. Section IV of the paper introduces the new evaluation metrics proposed to assess the performance of lane detection systems. In Section V, we quantitatively and visually compare the performance of LaneNet and eLaneNet. Section VI concludes our work and briefly discusses possible future work.

II. Related work

In this section, we provide a comprehensive overview of pertinent literature and frequently employed datasets pertaining to Lane Detection in Autonomous Vehicles.

A. Lane Detection Methods

Various studies [8]–[11] have outlined diverse approaches for detecting and predicting lane lines.These methods can be generally classified into two categories: Traditional Methods and Deep Learning Methods.

Traditional Methods: In the era predating the rise of deep learning, lane detection relied on conventional geometric modeling techniques, such as line detection or fitting. The lane detection process usually involved four main stages: image preprocessing, feature extraction, model fitting, and lane tracking. Feature extraction encompasses the utilization of attributes, such as texture, gradients, and colors to discern essential features crucial for the identification of lane lines. Image preprocessing involved tasks such as converting colored RGB images to grayscale, reducing noise, selecting the Region of Interest (ROI), and conducting edge detection [12].

ROI selection involves vanishing point detection, perspective analysis with a projective model, and sub-sampling [13]. The idea behind using the vanishing point is that a correctly estimated vanishing point provides a strong clue about the region to localize. The authors in study [14] tackled road detection, by estimation of the vanishing point associated with the main (straight) part of the road, followed by the segmentation of the corresponding road area based on the detected vanishing point. Perspective analysis with a projective model, leverages the concept that parallel lane markings within the real-world plane converge at a vanishing point within the image plane. This approach frequently employs perspective analysis to refine the scope of detection to a precise region, which is then identified as the ROI. Through the establishment of a cohesive projection that interconnects the image plane, real-world plane, and camera plane, the process of extracting the ROI is streamlined. In study [15], a perspective projection model connects the camera and road plane, projecting lane marker edge points onto a road-space grid. The central lane line is defined by points on the grid's upper and lower edges, with each grid segment described by its offset from the lower-left point and the horizontal deviation between endpoints. In subsampling either a predefined or an adaptive region of the image is used to determine the ROI. Examples are given in [16].

Edge detection operators can be classified into Gradient and Laplacian operators, although there are additional operators that do not strictly adhere to these categories [17]. The gradient method detects edges by looking for the maximum and minimum in the first derivative of the image. The Laplacian method uses zero crossings in the second derivative of the image to find edges. Gradient based edge detectors include Roberts, Sobel and Prewitt operators while Laplacian based edge detectors include Marrs-Hildreth edge detector. The authors in study [17] studied various edge detectors and

 ISBN: 978-1-68558-140-4

concluded that under noisy conditions, Canny, LoG, Sobel, Prewitt, Roberts's exhibited better performance, respectively. They also concluded that Canny's edge detection algorithm has a better performance compared to the others on images.

Generally, two kinds of features exist for extracting lane lines: colors and edges [18]. Lane detection techniques can be grouped into edge-based methods, color-based methods and hybrid (edge and color) methods [13]. The Hough transform and its variants, such as, Adaptive Hough Transform and Probablistic Hough Transform are the most popular edge-based methods [13]. Steerable filter is also an edge-based technique that has been applied in many research [19] [20] [21] [22] with good results especially when road markings exhibit a clear and uniformly smooth appearance. Color-based methods have the limitation of being influenced by lighting and hence are not widely used by researchers. An example of a colour based method, HSILMD, was proposed by the authors in [23]. In HSILMD, full-color images are transformed into HSI color representation within a region of interest (ROI) to detect the road surface on the host vehicle. Using the Fuzzy c-Means algorithm, intensity distribution differences within an ROI row of pixels are recorded and clustered, enabling lane marking detection via selected intensity and saturation thresholds. Hybrid methods usually combine width, length, and location of lines with gray levels and brightness values of pixels, which improve the extraction results. An example is given in [24].

Images captured by vehicle cameras are captured in a continuous sequence. This sequential nature of image acquisition allows for an overlap between lanes detected in the current frame and those from the preceding frame. By leveraging information from both the current and previous frames, we can anticipate lane positions and track their evolution over time, enabling a more robust and accurate lane tracking process. Common trackers include Kalman filters and Particle filters [12].

Deep-learning-based methods: Deep learning lane detection methods can be grouped into: encoder-decoder CNN, Fully-Convolutional Neural (FCN) networks with optimization algorithms, CNN+RNN, and GAN model [25].

1) Encoder-decoder CNN: The encoder-decoder CNN architecture is frequently employed in semantic segmentation tasks [25]. Two examples worth discussing are LaneNet [6] and IBN-Net [26]. In the original LaneNet, which we would improve upon in this paper, an encoder-decoder network was used for binary and instance segmentation. Binary segmentation consists of segmenting the pixels into lanes and background. Instance segmentation consists of generating embeddings for lane pixels. IBN-Net improves on LaneNet by using an attention-based encoder-decoder network for lane detection. IBN-Net's encoder-decoder network also generates a binary and instance embedding. The difference between IBN-Net and LaneNet in the encoder-decoder network is that the encoder and decoder are connected by a self-attention layer.

2) FCN with optimization algorithms: In this architecture, lane detection, lane-marking identification, and vanishing-point extraction are achieved using optimization algorithms [25]. Commonly used optimization algorithms include clustering and subsampling. The architecture employs the concept of a vanishing point to guide the predictions of lane markings and road regions. VPGNet [27] leverages the vanishing point as a guiding factor and optimizes the joint prediction of lane information and road layout. This results in improved accuracy and robustness in handling complex road scenes. Deep learning methods for lane detection involving clustering is dominated by semantic segmentation algorithms. Image pixels are classified by the deep neural network, and the lane line information is extracted by clustering and other post-processing methods. An example of a deep learning method involving clustering is LaneNet.

3) CNN+RNN: Lane detection methods employing CNN+RNN architectures operate on the premise that the combination of Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) allows for the utilization of spatial and sequential information present in road scenes respectively. Through a series of convolutional and pooling layers, CNNs are adept at extracting progressively abstract features, which are crucial for accurate lane detection. However, the analysis of road scenes necessitates the consideration of not only spatial features but also the evolution of lane configurations over time. RNNs excel at modeling sequential data by incorporating memory of prior inputs, thus capturing the dynamic nature of lane movements over frames. The unified CNN+RNN architecture therefore acquires the capacity to understand both immediate lane contexts and the evolution of the lane over time [25] [28]. This can be helpful in estimating the positions of lanes that have been obscured in the current driving scene.

4) GAN model: Given that certain lane detection methods rely on semantic segmentation, and given that Generative Adversarial Networks (GANs) are equipped to perform semantic segmentation tasks, GANs can also serve a purpose in lane detection applications. Two loss functions are used to guide the process of semantic segmentation in GANs. The generator loss function is responsible for ensuring that the GAN produces correct predictions while the discriminator loss term is more concerned with the overall image being segmented. It ensures that the individual pixel-wise predictions are consistent with each other. Thus, the discriminator evaluates the authenticity and quality of the predictions generated by the generator in a GAN. Examples of this approach are given in [29]–[31].

B. Datasets

Some common lane detection datasets include: TuSimple, BDD100K and Unsupervised LLAMAS.

1) TuSimple dataset - The TuSimple dataset was obtained from US highways and display a range of weather conditions. It consists of 358 images for validation, 2,782 images for testing and 3,626 for training, totalling 6,408 images.

2) BDD100K dataset - This dataset is drawn from more than 50,000 rides across New York and the San Francisco Bay Area city from streets, residential areas, and highways.

 ISBN: 978-1-68558-140-4

It contains 100K driving videos, each lasting 40 seconds. The videos are split into training (70K), validation (10K) and testing (20K) sets. The dataset is made of 720p high resolution images, with a frame rate of 30 fps and GPS/IMU recordings to preserve the driving trajectories. Ten tasks are associated with the dataset: image tagging, lane detection, drivable area segmentation, road object detection, semantic segmentation, instance segmentation, multi-object detection tracking, multi-object segmentation tracking, domain adaptation, and imitation learning.

3) Unsupervised LLAMAS dataset - Comprising of 100,042 labeled lane marker images, the unsupervised LLAMAS dataset stems from approximately 350 kilometers of recorded drives. The image labels are automatically generated, first by projecting markers into camera images, and then through further optimization to enhance label accuracy. The dataset annotations include pixel-level annotations for dashed lane markers, as well as the 3D and image space endpoints for individual markers, along with lane associations for each marker. The challenges presented within this dataset encompass a pixel-level binary segmentation problem, a segmentation problem intertwined with lane association, and a lane estimation task.

III. Methodology

In this section, we present the well-established LaneNet network. Subsequently, we introduce the enhanced version of the LaneNet network, eLaneNet.

A. LaneNet

LaneNet is structured as a two-step lane detection network as illustrated in Figure 1. Two-step lane detection methods are composed of a feature extracting step and a post-processing step [32]. LaneNet's feature extraction stage comprises the use of deep learning techniques to segment an image into two categories: binary segmentation and instance segmentation. The post-processing phase involves clustering, which groups lane pixels into clusters. Lane pixels belonging to the same lane will be in the same cluster. Finally, the fitting operation employs mathematical models to precisely define the trajectory of each lane, further enhancing the accuracy of lane boundary representation. The details of each part of LaneNet's architecture are explained below. Similar processes in both eLaneNet and LaneNet have the same number in Figure 1 and Figure 2.

Input Image (input 1) and Resize Image (process 2): In the image processing pipeline, input images are resized from their original resolution of $\alpha m \times \beta n \times c$ pixels to a reduced resolution of $(m \times n \times c)$, where $\alpha m, \beta n, m, n, c \in \mathbb{N}$. Resizing the images allows for faster computation and accommodates GPU and memory constraints.

Shared Encoder (process 3): LaneNet's shared encoder architecture is based on the ENet encoder-decoder network [33]. Two modifications to ENet's architecture was introduced in LaneNet's shared encoder. Firstly, the output of ENet was adapted to create a two-branched network, accommodating both binary segmentation and instance segmentation branches within LaneNet. Secondly, in LaneNet, only the first two stages (stages 1 and 2) of ENet's encoder are shared between the two branches, while the full ENet decoder (stages 4 and 5) serves as the backbone for each separate branch. This means that stage 3 of ENet's encoder is not used in LaneNet.

The binary segmentation branch produces a one-channel image while the instance segmentation branch produces an N-channel where N represents the embedding dimension. In this context, a k-channel image indicates that information about a specific attribute of a pixel is stored in k-dimensions. As an example, consider a color image with three channels: red, green, and blue. Each channel encodes the intensity of its respective color at each pixel, allowing for the representation of a wide spectrum of colors in the image. In the binary segmentation map of the binary segmentation branch, an output of 1 indicates that a pixel belongs to a lane instance while an output of 0 indicates that the pixel belongs to a background. In the N-channel image produced by the instance segmentation branch, an embedding dimension encodes information about which lane instance a pixel belongs to.

Segmentation Branch (process 5): The segmentation branch of the network is designed to produce a binary segmentation map, which classifies the pixels into either lane or background categories. The class weighted cross entropy loss [33] is used to account for imbalance between the lane pixels and the background pixels. As stated earlier, the output of the segmentation branch is a binary segmentation map which classifies pixels into either lane or background. Since the background pixels far exceed the lane pixels, there is an imbalance between the lane pixels and the background pixels. To address this, the class weighted cross entropy loss [33] is used to account for the imbalance between the lane pixels and the background pixels.

Embedding Branch (process 4): Embeddings produced by the embedding branch have the characteristic that lane pixels belonging to the same lane have similar embeddings while lane pixels belonging to different lanes have different embeddings. This phenomenon is achieved by using the clustering loss function in (1). In (1), the L_v term minimizes the distance between pixel embeddings belonging to the same lane. Another way to interpret this phenomenon is that the L_v term is a variance term that applies a pull force on each embedding towards the mean embedding of pixels belonging to that lane.

In order to ensure, we can distinguish between lane pixels belonging to different lanes, a distance term (L_d) is introduced. The L_d term pushes the cluster centers away from each other. In this context, a cluster center refers to the mean embedding of pixels belonging to a particular lane. Both terms are hinged. That is, the L_v term activates when an embedding is at a distance of more than δ_v from its cluster center. The pushing force, L_d between the cluster centers only activates when the centers are at a distance less than δ_d from each other. In this context, δ_d is the minimum distance allowed between cluster centers while δ_v is the maximum distance allowed between an embedding and the mean embedding of its corresponding cluster. Let K denote the number of clusters (lanes), N_k the

 ISBN: 978-1-68558-140-4

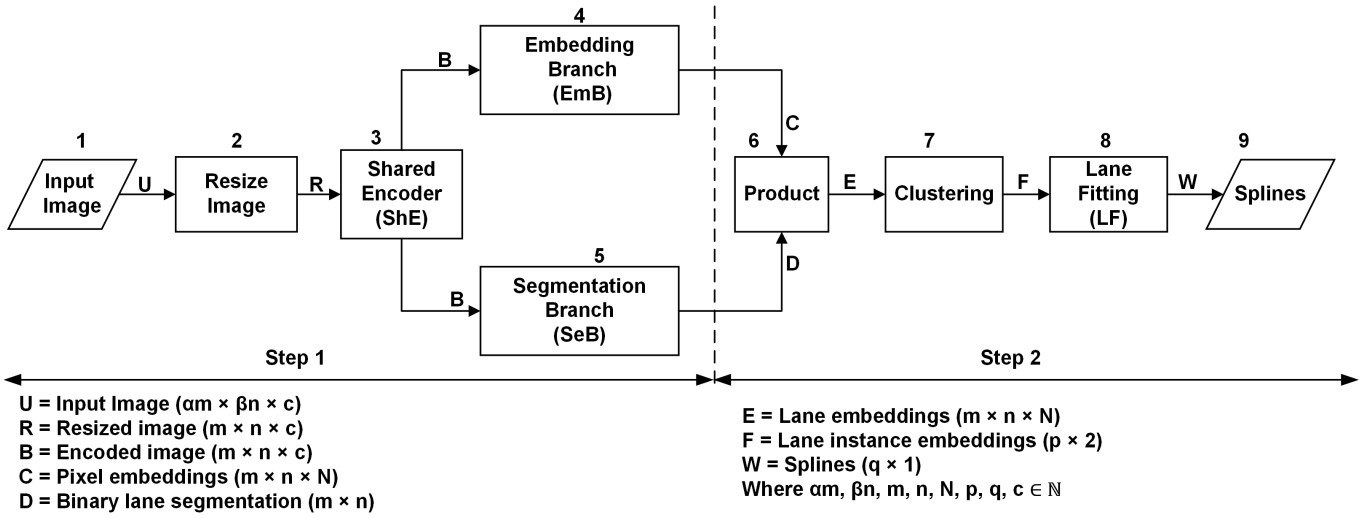


Figure 1. LaneNet architecture [6].

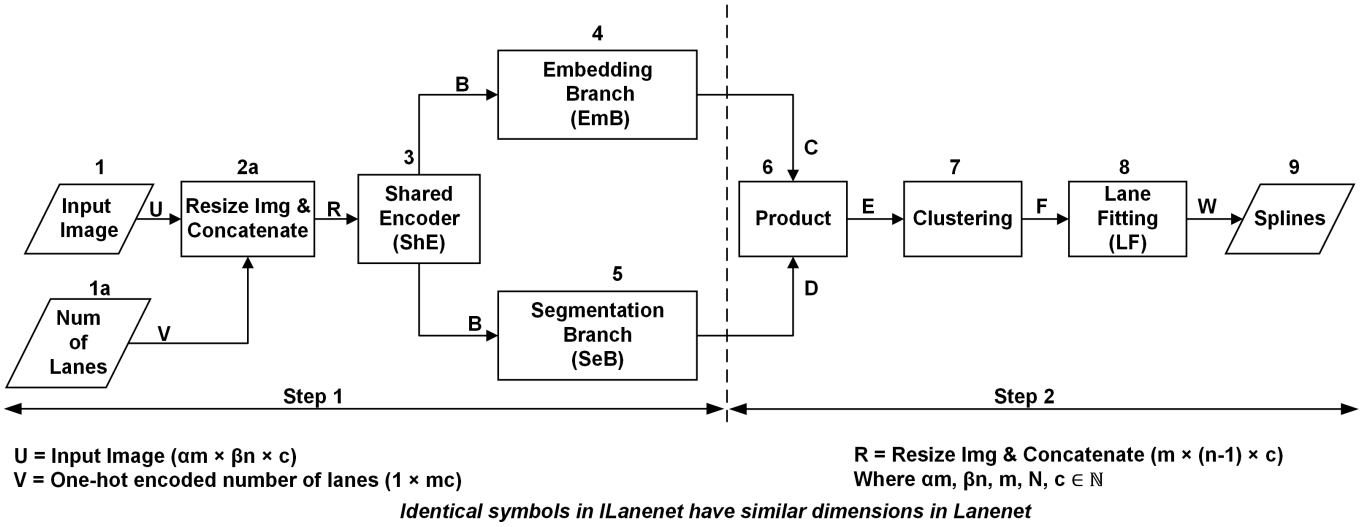


Figure 2. ELaneNet architecture.

number of elements in cluster k where $1 \leq k \leq K$, x_i is the i^{th} pixel embedding in cluster k, μ_k the embedding of cluster k, $\|\cdot\|$ the L2 distance, and $[x]_+ = \max(0, x)$ the hinge, the total loss L is equal to $L_v + L_d$. The quantities L_v and L_d are defined in (1).

$$\begin{cases} L_v = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{N_k}\sum_{i=1}^{N_k}[\|\mu_k - x_i\| - \delta_v]_+^2 \\ L_d = \frac{1}{K(K-1)}\sum_{k_A=1}^{K}\sum_{\substack{k_B=1 \\ k_A \neq k_B}}^{K}[\delta_d - \|\mu_{k_A} - \mu_{k_B}\|]_+^2 \end{cases} \tag{1}$$

The LaneNet clustering process is performed iteratively with the loss function in (1) until the network converges. Upon convergence of the network, clusters will emerge in the embeddings of lane pixels. These clusters will exhibit a separation distance larger than δ_d from adjacent clusters, with each cluster possessing a radius smaller than δ_v.

Product (process 6) and Clustering (process 7): In the image's binary segmentation map, 0s represent background pixels, while 1s represent lane pixels. Hence, to specifically extract embeddings related to the lane pixels, we multiply the results from both the embedding and segmentation branches. This step effectively eliminates all non-lane pixels, leaving us exclusively with embeddings associated with the lanes. The subsequent application of clustering techniques helps identify distinct lane pixels corresponding to a specific lane instance. The LaneNet clustering process is executed iteratively, with the condition $\delta_d > 6\delta_v$. This condition mandates that the relationship $\delta_d > 6\delta_v$ is satisfied by the parameters δ_d and δ_v.

 ISBN: 978-1-68558-140-4

A two step approach is used when selecting a lane embedding to threshold on. First, the selected point is shifted closer to the cluster center using the mean shift algorithm. This step helps to refine the initial selection. After this shift, the thresholding process is applied, which results in the accurate identification of lane embeddings within the specified radius. This addresses the concern of accidently selecting an outlier lane embedding to threshold on.

Lane Fitting (process 8) and Splines (output 9): The process of fitting the lane in LaneNet is described as follows: First, assume we have matrix H which is the perspective transform for converting from the driving scene to bird's eye view (BEV).

Assume we have a lane pixel $\mathbf{p_i} = [x_i, y_i, 1]^T \in \mathbf{P}$ where $\mathbf{P}$ is the set of pixels belonging to a particular lane. To transform the pixel to BEV, we use H to perform the operation $\mathrm{H}\mathbf{p_i}$ where H is the perspective transformation matrix and $\mathbf{p_i'} = [x_i', y_i', 1]^T \in \mathbf{P'}$, the transformed pixels. After transforming the pixels to BEV, the least squares algorithm is then used to fit a low order polynomial,$f(y')$, through $\mathbf{P'}$.

To evaluate the x-value, x_i^*, of a lane at a given y-coordinate y_i, the point $\mathbf{p_i} = [-, y_i, 1]^T$ is first transformed to BEV using the expression $\mathbf{p_i'} = \mathrm{H}\mathbf{p_i} = [-, y_i', 1]^T$. Since the current x-value in point $\mathbf{p_i}$ is irrelevant, we represent it with a -. After transformation, we use the low order polynomial with which we fitted the lane curve to evaluate the x-value of a lane at the given y position. This step is shown as follows: $x_i'^* = f(y_i')$. We then reproject the point from BEV back to the original image space using $\mathbf{p_i^*} = \mathrm{H}^{-1}\mathbf{p_i'^*}$ where H^{-1} is the inverse perspective transformation matrix. Using this approach, we can evaluate the x-values at different y-positions.

In LaneNet, the perspective transform matrix, H, is gotten by training a neural network called HNet. In converting the image from Bird's Eye View (BEV) back to the original image space, we utilized the equation $\mathbf{p_i^*} = \mathrm{H}^{-1}\mathbf{p_i'^*}$. However, while implementing H-Net, we encountered an issue during its training phase where non-invertible matrices were outputted. This complication prevented the conversion from BEV to the original space, impeding loss calculation and leading to the suspension of network training. To address this obstacle, we chose to skip the lane fitting step altogether. Despite this omission, both LaneNet and eLaneNet exhibited satisfactory performance.

B. Improved LaneNet

ELaneNet is enhanced by a simple modification to LaneNet's architecture in step 1 and concatenating it with the resized image in step 2, while keeping the rest of the network unchanged. To augment the input image with the number of lanes (NoL), we first performed one-hot encoding on the lane count. The resulting one-hot encoded representation was then passed through a fully connected (FC) layer, which processed the lane information and produced an output ready for reshaping. This FC layer establishes connections between every input neuron and every output neuron. After reshaping, the output from the FC layer was concatenated with the original image. The resulting combined input served as the input for the LaneNet model, giving rise to a modified network known as eLaneNet.

Assume we have an initial image with dimensions $1280 \times 720 \times 3$ (αm width $\times$ βn height $\times$ c channels) and want to rescale it to a target size of $512 \times 256 \times 3$ (m width $\times$ n height $\times$ c channels). There are two different approaches with regards to LaneNet and eLaneNet.

In LaneNet, the process of rescaling is straightforward. We directly rescale the image to the target size of $512 \times 256 \times 3$.

On the other hand, the eLaneNet approach takes a slightly different route. The image is first rescaled to $512 \times 255 \times 3$ (m width $\times$ $(n-1)$ height $\times$ c channels). Additionally, the number of lanes is one-hot encoded such that the possible number of lanes ranging from 1 to $m \cdot c$ has a unique representation. To represent a lane uniquely, each position in the array corresponds to the number of lanes associated with the driving scene. Since there is only one number of lanes associated with a driving scene, one position in the array is "hot" (set to 1) while the others are "cold" (set to 0). For example, 1 lane can be encoded as $[1, 0, 0, \ldots, 0]$, 2 lanes can be encoded as $[0, 1, 0, \ldots, 0]$, 3 lanes can be encoded as $[0, 0, 1, \ldots, 0]$ and $m \cdot c$ lanes can be encoded as $[0, 0, 0, \ldots, 1]$.

This encoded lane information undergoes further processing: it passes through a fully connected layer and is reshaped into a tensor of size $512 \times 1 \times 3$ (m width $\times$ 1 height $\times$ c channels). The next step involves combining this reshaped lane information with the resized image of $512 \times 255 \times 3$. The concatenation of these two tensors results in an output tensor of size $512 \times 256 \times 3$, resembling the shape used in the LaneNet approach. This concatenated tensor, which includes both the image and the encoded lane information, serves as the input for the LaneNet network.

Figure 2 illustrates the eLaneNet architecture, while Figure 3 and Figure 4 provide a detailed depiction of the process involved in merging the inputs. The algorithm in Figure 4 first resizes the input image, then one-hot encodes the lane information, extracts lane information using a Fully Convolutional Network (FCN), reshapes the output, and finally concatenates it with the resized image to produce the output image with added lane information.

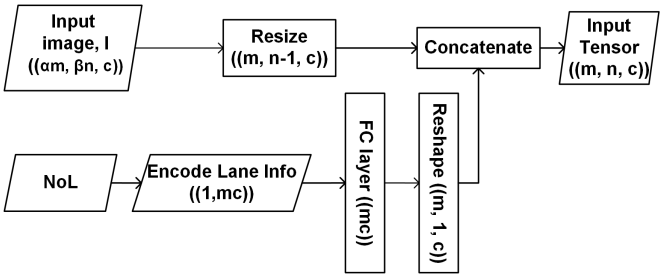


Figure 3. Concatenating image with lanes.

Algorithm 1 Add Lane Information

Input:
I: Input image (αm $\times$ βn $\times$ c)
NoL: Number of Lanes where NoL > 0
m: Width of the target image
c: Number of channels in input image
n: Height of the target image
Output:
outImg: Image with added lane information (m $\times$ n $\times$ c)

procedure
    // Resize the input image to m $\times$ (n $-$ 1) $\times$ c
    $resizedImg \leftarrow$ ResizeImage(I, (m, n-1, c))
    // One-hot encode lane information
    $laneInfo \leftarrow$ EncodeLaneInformation(NoL)
    // Extract lane information using FCN network
    $extractedInfo \leftarrow$ FCN(laneInfo, size=mc)
    // Reshape output
    $laneInfo \leftarrow$ Reshape(extractedInfo, (m, 1, c))
    // Concatenate lane information with the image
    $outImg \leftarrow$ Concatenate(resizedImg, laneInfo)
    return $outImg$
end procedure

Figure 4. Algorithm for adding lane information to an image.

IV. Performance Metrics

Within the LaneNet framework [6], the evaluation of lane detection accuracy, represented as the average correct number of points per image, is conducted using the formula outlined in (2).

$$Accuracy = \sum_{im} \frac{C_{im}}{S_{im}} \tag{2}$$

with im denoting a driving scene in the dataset, C_{im} the total number of correctly predicted points in im and S_{im} the total number of ground-truth points in im. A correct point is defined as one where the disparity between the predicted point and the ground truth falls below a specific threshold. Equations (3) and (4) provide the formulas for computing the false positive and false negative scores, respectively.

$$\textit{False Positive Score } (FPS^l) = \frac{F^l_{pred}}{N^l_{pred}} \tag{3}$$

$$\textit{False Negative Score } (FNS^l) = \frac{M^l_{pred}}{N^l_{gt}} \tag{4}$$

with F^l_{pred} denoting the total number of falsely predicted lane lines, N^l_{pred} denoting the total number of correctly predicted lane lines, M^l_{pred} denoting the total number of missed ground-truth lane lines and N^l_{gt} denoting the total number of all ground-truth lane lines.

As described in contribution C2 of Section I, we propose the introduction of a novel performance metric referred to as the *capacity* of the lane detection system. The mathematical expression for capacity is provided in (5). To elucidate this concept, consider a scenario with two lanes on the road. When the lane detection system accurately identifies both lanes, it indicates that more ADSs can traverse the road smoothly. Essentially, this implies that the lane detection system exhibits a higher capacity by facilitating the full utilization of available lanes, thereby enhancing overall traffic flow efficiency.

On the flip side, when the system accurately identifies only one lane but fails to detect the other, it suggests the possibility of confining all vehicles to a sole lane on the road. This limitation could result in less-than-optimal utilization of the road, potentially causing traffic congestion. In these scenarios, the lane detection system is considered to have reduced capacity as it cannot efficiently utilize all available lanes, consequently hampering overall traffic efficiency. We therefore define the term "capacity" as the system's ability to detect and effectively utilize existing lane markings/lanes on the road. Conversely, lost capacity refers to the system's inability to effectively utilize existing lane markings/ lanes on the road.

We introduce a novel metric called the unsafe driving measure, alongside capacity and lost capacity considerations. The unsafe driving measure assesses the lane detection system's potential to yield inaccurate lane predictions, thereby influencing drivers to make unsafe decisions. False positives generated by the system wrongly indicate the presence of a lane. This misinformation may lead the ADS to perceive a road section as a legitimate lane, prompting an autonomous vehicle to attempt unsafe maneuvers. The corresponding expressions for capacity, lost capacity, and the unsafe driving measure are provided in (5), (6), and (7), respectively.

$$\text{Capacity}^l = \frac{TP^l}{TP^l + FN^l} \tag{5}$$

$$\text{Lost Capacity}^l = 1 - \text{Capacity}^l \tag{6}$$

$$\text{Unsafe Driving Measure}^l = \frac{FP^l}{TP^l + FN^l} \tag{7}$$

with TP^l representing the total count of accurately predicted lanes, FP^l denoting the total count of erroneously predicted lanes, and FN^l indicating the total count of missed ground-truth lanes. It's important to highlight the similarity between the expressions for capacity and the expressions for recall and false positive score, respectively.

When conceptualizing a lane as a mere line, it may lack the practical details necessary for effective use. Picture a scenario where the network outputs only a singular lane line. In this scenario, crucial information about the lane's boundaries is missing, making it difficult for a vehicle to ascertain appropriate passing areas. To overcome this limitation, we propose a more advanced lane abstraction approach.

 ISBN: 978-1-68558-140-4

In the proposed lane abstraction approach, lanes are considered as distinct entities as opposed to individual lane lines. This method proves beneficial by imparting a clearer understanding of the road configuration, enabling vehicles to make more informed decisions regarding lane changes and secure navigation.

To better align with the lane abstraction approach, adjustments were implemented in the following equations to account for lanes rather than lines. The expressions for the false positive score and false negative score in the lane abstraction approach are presented in Equations (8) and (9), respectively.

$$FPS^L = \frac{F^L_{pred}}{N^L_{pred}} \tag{8}$$

$$FNS^L = \frac{M^L_{pred}}{N^L_{gt}} \tag{9}$$

with F^L_{pred} as the count of incorrectly predicted lanes, N^L_{pred} as the total number of predicted lanes, M^L_{pred} as the count of missed ground-truth lanes, and N^L_{gt} as the total number of ground-truth lanes. The formulations for capacity, lost capacity, and the unsafe driving measure in the lane abstraction approach are provided in (10), (11), and (12), respectively:

$$\text{Capacity}^L = \frac{TP^L}{TP^L + FN^L} \tag{10}$$

$$\text{Lost Capacity}^L = 1 - \text{Capacity}^L \tag{11}$$

$$\text{Unsafe Driving Measure} = \frac{FP^L}{TP^L + FN^L} \tag{12}$$

with TP^L denoting the total count of accurately predicted lanes, FP^L representing the overall count of incorrectly predicted lanes, and FN^L denoting the total count of ground-truth lanes that were missed.

V. Experiments and Results

This section extensively explores experiments comparing LaneNet and eLaneNet. A thorough analysis of the results supports the claim that eLaneNet outperform LaneNet, offering valuable insights into its enhanced performance.

A. Setup

LaneNet and Improved LaneNet: The TuSimple dataset was utilized for training, and both networks were trained with an embedding dimension (N) of 4. Additionally, δ_v was set to 0.5, and δ_d was set to 3. The images underwent rescaling to 512×256. The training of the network involved using the Adam optimizer with a batch size of 32 and a learning rate of 5e-4 until convergence.

B. Performance and comparison

Quantitative Analysis: The results for the network using the line abstraction approach is given in Table I while the results for the lane abstraction approach is given in Table II.

Line Abstraction In the realm of line abstraction, the comparison between eLaneNet and LaneNet reveals that eLaneNet surpasses LaneNet across various performance metrics, establishing a subtle yet significant advantage of eLaneNet over LaneNet. The prowess of eLaneNet becomes particularly evident in its capacity to mitigate both false positives and false negatives, ultimately resulting in a higher capacity for lane detection, and an accuracy of 93.1%. This highlights eLaneNet's superior ability to accurately identify existing lane markings and effectively accommodate vehicles in all lanes, thereby minimizing instances of lost capacity on the road.

One of the noteworthy strengths of eLaneNet lies in its exceptional reduction of false positives, with a score of 13.9%, as opposed to LaneNet's 23.0%. This discrepancy underscores eLaneNet's effectiveness in minimizing the likelihood of erroneously identifying non-existent lanes as real. In the context of an ADS that leverages eLaneNet, this translates to safer and more reliable driving maneuvers. The reduced false positive rate implies a decreased risk of the system misinterpreting irrelevant features as valid lane markings, contributing to enhanced precision and reliability in autonomous navigation.

In essence, the comparative analysis demonstrates that eLaneNet not only outperforms LaneNet in terms of overall accuracy but also excels in specific aspects crucial for robust lane detection. The higher capacity and lower unsafe driving measure collectively underscore eLaneNet's advanced capabilities in identifying and interpreting lane information, making it a favorable choice for applications demanding precision and reliability in autonomous driving scenarios.

Lane Abstraction In the comparison between eLaneNet and LaneNet, the focus was on evaluating various metrics that are crucial for assessing the performance of lane detection systems. Unlike traditional approaches that assess metrics based on individual lines, this evaluation considered a more comprehensive approach by analyzing metrics at the level of entire lanes. The results, presented in Table I, shed light on the superiority of eLaneNet over LaneNet in several key aspects.

Firstly, the metric of used capacity, representing the accuracy of identifying lanes, was found to be significantly higher for eLaneNet (87.5%) compared to LaneNet (80.4%). This indicates that eLaneNet is more proficient at recognizing and delineating lanes, contributing to a more accurate representation of the road environment.

Moreover, the assessment of lost capacity, which reflects instances where the system fails to identify lanes correctly, also favored eLaneNet. The lower lost capacity of eLaneNet suggests that it experiences fewer instances of missing lanes compared to LaneNet.

In terms of safety, the metric of unsafe driving measure was introduced, and eLaneNet demonstrated a notably lower score (27.3%) compared to LaneNet (38.5%). This implies that an Autonomous Driving System (ADS) utilizing eLaneNet is

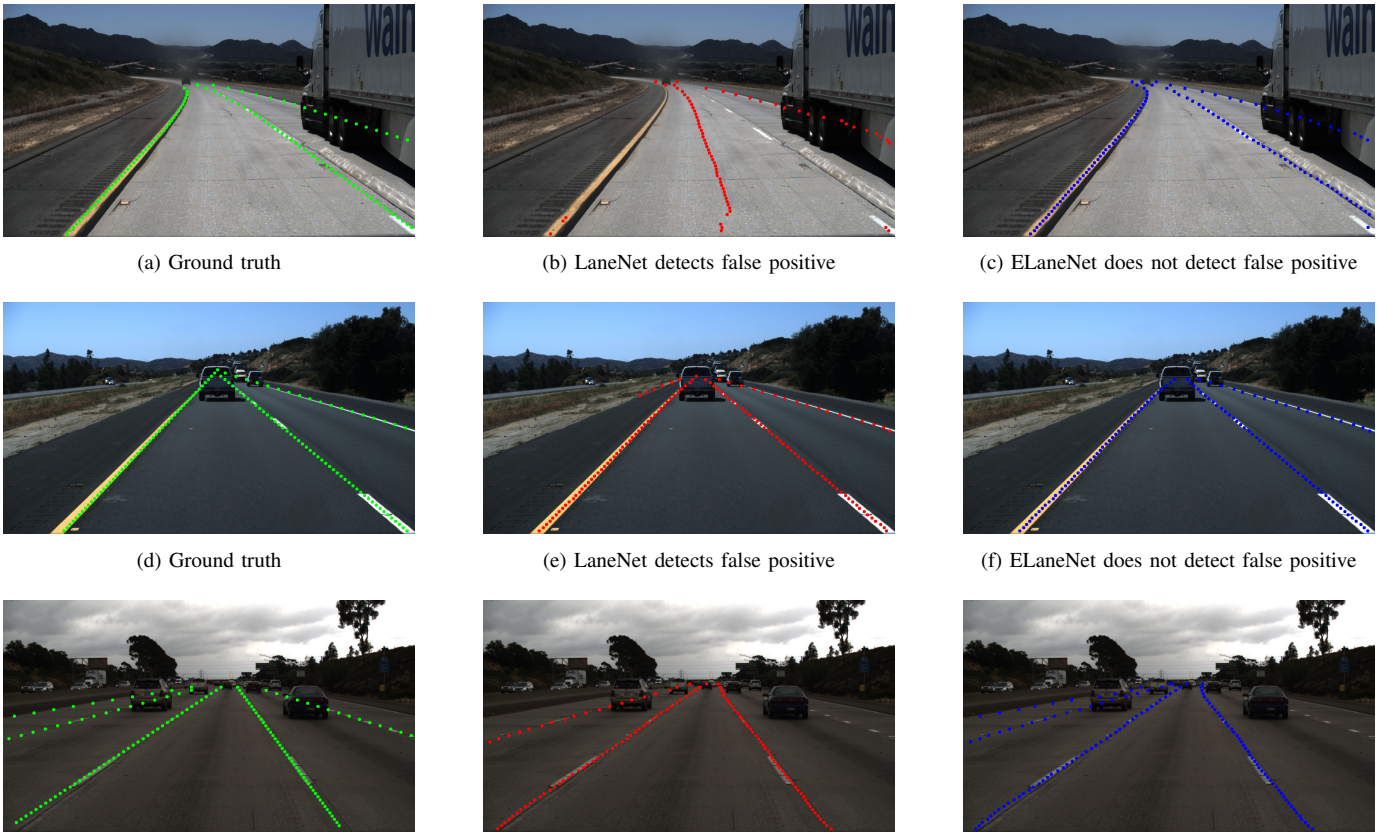
 ISBN: 978-1-68558-140-4

TABLE I. LANE ABSTRACTION

NETWORK	USED CAPACITY (RECALL)	LOST CAPACITY (FN SCORE)	UNSAFE DRIV. MEASURE (FPS SCORE)
ELaneNet	87.5 %	12.5 %	27.3 %
LaneNet	80.4 %	19.6 %	38.5 %

TABLE II. LINE ABSTRACTION

NETWORK	USED CAP (RECALL)	LOST CAPACITY (FN SCORE)	UNSAFE DRIVING MEASURE	ACC
ELaneNet	93.1 %	6.9 %	13.9 %	94.5 %
LaneNet	88.9 %	11.1 %	23.0 %	92.3 %



(a) Ground truth (b) LaneNet detects false positive (c) ELaneNet does not detect false positive

(d) Ground truth (e) LaneNet detects false positive (f) ELaneNet does not detect false positive

(g) Ground truth (h) LaneNet misses a lane (i) ELaneNet identifies one missed lane

Figure 5. Lane Detection in LaneNet and eLaneNet.

less likely to result in unsafe driving conditions compared to its counterpart LaneNet. The lower unsafe driving measure underscores the importance of accurate lane detection in enhancing the safety of autonomous vehicles.

Visual Analysis: From Figure 5 presented above, we can gain some insight into the lane detection capabilities of LaneNet and ELaneNet.

In the context of false positive detection, the ground truth images (Figures 5a and 5d) serve as the baseline, representing the actual lane markings. LaneNet's performance, as shown in Figures 5b and 5e, reveal that it tends to detect additional, false positive lane markings not present in the ground truth. This suggests that LaneNet may have a tendency to over-detect lanes in certain scenarios. Conversely, eLaneNet's results in Figures 5c and 5f demonstrate that it is more conservative in its lane detection approach. ELaneNet does not detect these false positive lane markings, which is advantageous when accuracy and avoiding false alarms are paramount.

Additionally, when considering missed lane detection, Fig. 5g represents the ground truth with all the lane markings correctly annotated. However, Figure 5h shows that LaneNet misses two of the lane markings present in the ground truth. This indicates that LaneNet may have limitations in accurately identifying all lane markings. In contrast, Figure 5i illustrates eLaneNet's ability to successfully identify one of the two missed lane marking, showcasing its strength in capturing lane markings that may be overlooked by eLaneNet.

These sample observations reveal that LaneNet exhibits a higher false positive rate and often misses lane markings, whereas eLaneNet excels in capturing missed lane markings while minimizing false positives. This makes eLaneNet a better model compared to LaneNet overall. Our LaneNet

 ISBN: 978-1-68558-140-4

results closely align with the original paper [6], especially considering the absence of accounting for conditional homography.

VI. CONCLUSION

This paper introduced an enhanced version of LaneNet designed for robust lane detection in driving scenarios. The improved architecture incorporated multiple inputs, namely, the driving scene and the number of lanes. A fully connected layer was employed to extract information from the NoL, which was then combined with the input image to create the input for LaneNet. The results demonstrated that by reducing false positives and false negatives, eLaneNet exhibited better performance compared to LaneNet. Future work aims to further enhance the model by utilizing lane count information to extrapolate missing lanes and eliminate false positives. Additionally, the effectiveness of the eLaneNet will be assessed using other datasets [34].

REFERENCES

[1] National Highway Traffic Safety Administration, "Traffic Safety Facts 2015 Data: Pedestrians," U.S. Department of Transportation, Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115, April 2015, retrieved: December, 2023.

[2] D. J. Fagnant, K. Kockelman, "Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations," *Transp. Res. Part A: Policy Pract.*, vol. 77, pp. 167–181, 2015. Available: https://www.sciencedirect.com/science/article/pii/S0965856415000804, retrieved: December, 2023.

[3] Z. Yang, et al., "CANet: Curved Guide Line Network with Adaptive Decoder for Lane Detection," in *ICASSP 2023-2023 IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, pp. 1–5, 2023.

[4] J. Wang, et al., "A keypoint-based global association network for lane detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 1392–1401, 2022.

[5] T. Zheng, et al., "CLRNet: Cross Layer Refinement Network for Lane Detection," in *2022 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pp. 888–897, 2022. Available online: Available: https://doi.org/10.1109/CVPR52688.2022.00097, retrieved: December, 2023.

[6] D. Neven, B. De Brabandere, S. Georgoulis, M. Proesmans, L. Van Gool, "Towards End-to-End Lane Detection: an Instance Segmentation Approach," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 286–291, 2018. Available online: Available: https://doi.org/10.1109/IVS.2018.8500547, retrieved: December, 2023.

[7] Z. Bao, S. Hossain, H. Lang, X. Lin, "A review of high-definition map creation methods for autonomous driving," *Eng. Appl. Artif. Intell.*, vol. 122, pp. 106125, 2023. Available online: Available: https://doi.org/10.1016/j.engappai.2023.106125, retrieved: December, 2023.

[8] L. Chen, Q. Li, Q. Mao, Q. Zou, "Block-constraint line scanning method for lane detection," in *2010 IEEE Intelligent Vehicles Symposium*, pp. 89–94, 2010.

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

[10] S. Yenikaya, G. Yenikaya, E. Düven, "Keeping the vehicle on the road: A survey on on-road lane detection systems," *ACM Comput. Surv.*, vol. 46, no. 2, pp. 1–43, 2013.

[11] Q. Li, L. Chen, M. Li, S.-L. Shaw, A. Nüchter, "A Sensor-Fusion Drivable-Region and Lane-Detection System for Autonomous Vehicle Navigation in Challenging Road Scenarios," *IEEE Trans. Veh. Technol.*, vol. 63, pp. 540–555, 2014.

[12] N. B. Chetan, J. Gong, H. Zhou, D. Bi, J. Lan, L. Qie, "An Overview of Recent Progress of Lane Detection for Autonomous Driving," in *2019 6th Int. Conf. Dependable Syst. Their Appl. (DSA)*, pp. 341–346, 2020. Available: https://doi.org/10.1109/DSA.2019.00052, retrieved: December, 2023.

[13] A. Mammeri, A. Boukerche, Z. Tang, "A real-time lane marking localization, tracking and communication system," *Comput. Commun.*, vol. 73, pp. 132–143, 2016. Available: https://www.sciencedirect.com/science/article/pii/S0140366415003096, retrieved: December, 2023.

[14] H. Kong, J.-Y. Audibert, J. Ponce, "Vanishing point detection for road detection," in *2009 IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 96–103, 2009.

[15] Q. Li, J. Zhou, B. Li, Y. Guo, J. Xiao, "Robust Lane-Detection Method for Low-Speed Environments," *Sensors*, vol. 18, no. 12, p. 4274, 2018. Available: https://www.mdpi.com/1424-8220/18/12/4274, retrieved: December, 2023.

[16] A. M. Shihavuddin, K. Ahmed, M. S. Munir, K. R. Ahmed, "Road boundary detection by a remote vehicle using radon transform for path map generation of an unknown area," *Int. J. Comput. Sci. Network Secur.*, vol. 8, no. 8, pp. 64–69, 2008.

[17] G. T. Shrivakshan, C. Chandrasekar, "A comparison of various edge detection techniques used in image processing," *Int. J. Comput. Sci. Issues (IJCSI)*, vol. 9, no. 5, p. 269, 2012.

[18] J. Niu, J. Lu, M. Xu, P. Lv, X. Zhao, "Robust Lane Detection using Two-stage Feature Extraction with Curve Fitting," *Pattern Recognit.*, vol. 59, pp. 225-233, 2016. Available: https://www.sciencedirectz.com/science/article/pii/S0031320315004690, retrieved: December, 2023.

[19] R. K. Satzoda, M. M. Trivedi, "Drive analysis using vehicle dynamics and vision-based lane semantics," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 9–18, 2015.

[20] S. Sivaraman, M. M. Trivedi, "Integrated lane and vehicle detection, localization, and tracking: A synergistic approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 906–917, 2013.

[21] J. C. McCall, M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: survey, system, and evaluation," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 20–37, 2006.

[22] J. M. Wang, Y. C. Chung, S. L. Chang, S. W. Chen, "Lane marks detection using steerable filters," in *Proc. 16th IPPR Conf. Comput. Vis., Graphics Image Process.*, pp. 858–865, 2003.

[23] T. Y. Sun, S. J. Tsai, V. Chan, "HSI color model based lane-marking detection," in *2006 IEEE Intell. Transp. Syst. Conf.*, pp. 1168-1172, 2006.

[24] N. Apostoloff, A. Zelinsky, "Vision In and Out of Vehicles: Integrated Driver and Road Scene Monitoring," *I. J. Robotic Res.*, vol. 23, pp. 513-538, 2004.

[25] Q. Zou, et al., "Robust Lane Detection From Continuous Driving Scenes Using Deep Neural Networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 41–54, Jan. 2020.

[26] W. Li, F. Qu, J. Liu, F. Sun, Y. Wang, "A lane detection network based on IBN and attention," *Multimed. Tools Appl.*, vol. 79, pp. 16473–16486, 2020.

[27] S. Lee, et al., "VPGNet: Vanishing Point Guided Network for Lane and Road Marking Detection and Recognition," in *Int. Conf. Comput. Vision*, 2017, pp. 1965-1973.

[28] Y. Kortli, et al., "Deep embedded hybrid CNN–LSTM network for lane detection on NVIDIA Jetson Xavier NX," *Knowledge-Based Syst.*, vol. 240, p. 107941, 2022. Available: https://doi.org/10.1016/j.knosys.2021.107941, retrieved: December, 2023.

[29] Y. Liu, J. Wang, Y. Li, C. Li, W. Zhang, "Lane-GAN: A Robust Lane Detection Network for Driver Assistance System in High Speed and Complex Road Conditions," *Micromachines*, vol. 13, no. 5, p. 716, 2022. Available: https://www.mdpi.com/2072-666X/13/5/716, retrieved: December, 2023.

[30] M. Ghafoorian, C. Nugteren, N. Baka, O. Booij, M. Hofmann, "EL-GAN: Embedding Loss Driven Generative Adversarial Networks for Lane Detection," in *Proc. Eur. Conf. Comput. Vision (ECCV) Workshops*, 2019, pp. 256-272.

[31] Y. Zhang, Z. Lu, D. Ma, J.-H. Xue, Q. Liao, "Ripple-GAN: Lane Line Detection With Ripple Lane Line Detection Network and Wasserstein GAN," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1532-1542, 2021.

[32] J. Tang, S. Li, P. Liu, "A review of lane detection methods based on deep learning," *Pattern Recognit.*, vol. 111, p. 107623, 2021. Available: https://doi.org/10.1016/j.patcog.2020.107623, retrieved: December, 2023.

[33] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation," *CoRR*, abs/1606.02147, June 2016.

 ISBN: 978-1-68558-140-4

[34] K. Wolf, et al., “Compensation mechanism in tumor cell migration: mesenchymal–amoeboid transition after blocking of pericellular proteolysis,” *J. Cell Biol.*, vol. 160, pp. 267–277, 2003.

 ISBN: 978-1-68558-140-4