AI Innovations and Applications

DOI URL: https://doi.org/10.63944/aia.Vol.2-2
Vol.2 No.2

A Review of Deep Learning-Based Lane Detection Methods

Ting Liu [1*], Megat Norulazmi Megat Mohamed Noor [2], Mohammad Faizuddin Bin Md Noor [3]

[1] Department of Computer Engineering Technology, Universiti Kuala Lumpur – Malaysian Institute of Information Technology, Kuala Lumpur-50250, MALAYSIA | liu.ting@s.unikl.edu.my

[2] Department of Computer Engineering Technology, Malaysian Institute of Information Technology, MALAYSIA | megatnorulazmi@unikl.edu.my

[3] Department of CACS - Center for Alumni & Career Services, Universiti Kuala Lumpur – Malaysian Institute of Information Technology, Kuala Lumpur-50250, MALAYSIA | mfaizuddin@unikl.edu.my

[*] Corresponding Author: Ting Liu, Department of Computer Engineering Technology, Universiti Kuala Lumpur – Malaysian Institute of Information Technology, Kuala Lumpur-50250, MALAYSIA | liu.ting@s.unikl.edu.my

Abstract: Lane detection serves as a critical environmental perception application designed to identify lane markings utilizing onboard cameras or LiDAR systems. In recent years, concurrent with the advancement and deployment of computer vision applications, lane detection tasks have garnered substantial attention, leading to the emergence of a diverse array of lane detection methodologies. This article presents a comprehensive overview of lane detection techniques predicated on the acquisition of two-dimensional images via onboard cameras. Initially, it outlines the fundamental task of lane detection. Subsequently, it introduces datasets pertinent to lane detection. Following this, it delineates both traditional lane detection approaches and those grounded in deep learning. Ultimately, it provides an in-depth discussion of deep learning-based methods. Lane detection methodologies leveraging deep learning can be systematically categorized into four primary types: segmentation-based approaches, detection-based approaches, parameter curve-based approaches, and key point-based approaches.

Keywords: lane line detection; Instance segmentation; deep learning

1. Introduction

Autonomous driving, a hot topic in both academia and industry, has attracted considerable attention from researchers. To ensure vehicle safety, autonomous driving systems must precisely control the vehicle to travel stably along lane lines, a goal heavily reliant on accurate lane line perception. Lane detection plays a crucial role in autonomous driving systems, especially Advanced Driver Assistance Systems (ADAS). Given a two-dimensional front-view image captured by an onboard camera, the core task of lane detection is to accurately acquire the precise shape of each lane line on the road. This requires not only accurately capturing the direction and shape of the lane lines but also clearly distinguishing each lane line instance. From this perspective, lane detection and instance segmentation tasks share similarities.

However, lane detection faces numerous challenging difficulties. On the one hand, lane lines are inherently long and thin, and in real-world, complex scenarios, they vary greatly and often encounter occlusion issues, such as being blocked by other vehicles or obstacles. Changes in lighting conditions also significantly impact detection; strong light, backlighting, or shadows can all interfere with detection. Furthermore, semantic ambiguity is common, such as worn or blurred lane lines. These factors make instance-level recognition extremely difficult. On the other hand, for lane detection algorithms to be applied to in-vehicle systems, they must be able to process data in real time, placing extremely stringent demands on their real-time performance. Given limited hardware resources, improving algorithm performance becomes another crucial challenge that lane detection tasks must overcome. Therefore, developing a scientifically sound and reasonable lane detection method is particularly critical.

2. Detection Dataset

Building a lane line detection dataset requires data balancing across various scenario categories, such as highways, auxiliary roads, mountain roads, nighttime driving, and rainy weather, to simulate real-world driving environments. Open-source lane line datasets include:

2.1. TuSimple Dataset

A total of 7208 images were taken on a highway. The weather was clear and the lane lines were well-defined. The lane lines were marked with dots. The image size was 1280x720. As shown in Fig 1.

Figure 1. TuSimple dataset

The TuSimple dataset is a publicly available road dataset of American highways, specifically focusing on lane detection. The dataset encompasses a wide range of road conditions, traffic situations, and weather conditions, ensuring that the model can adapt to

various real-world driving scenarios.

2.2. CULane Dataset

There are approximately 133,235 images in total, covering eight difficult-to-detect scenarios including congestion, nighttime, missing lane markings, and shadows. A maximum of four lane lines can be marked. Image size: 1640x590; as shown in Fig 2.

Figure 2. CULane dataset

The CULane dataset is a public dataset specifically developed for lane detection tasks by a joint research team from Tsinghua University and the Chinese Academy of Sciences. It contains lane markings in various complex scenarios, such as curves, intersections, and occlusions.

2.3. Caltech Dataset

There are approximately 1200 pictures in total., with relatively simple scenes and good perspectives; image size: 640x480.

2.4. VPGNet Dataset

There are approximately 2,000 images in total, including daytime and nighttime data, as well as various lane line types and other lane markings such as left-turn arrows, straight-ahead arrows, and zebra crossings.

2.5. BDD100k Dataset

There are approximately 100,000 images in total, including various daytime and nighttime weather conditions in four regions of the United States, and 2D lane lines in eight categories; image size: 1280x720.

2.6. ApolloScape Dataset

There are approximately 140,000 images in total, characterized by lane lines marked in the form of masks, including 28 categories in 2D or 3D; image size: 3384x2710.

2.7. CurveLanes Dataset

Huawei's curve detection dataset contains approximately 150,000 images. All lane lines were manually labeled using cubic spline curves. It includes many complex scenarios, such as S-shaped roads, Y-shaped roads, as well as nighttime and multi-lane scenarios. The dataset is divided into a training set of 100,000 images, a validation set of 20,000 images, and a test set of 30,000 images; image size: 2650x144.As shown in Fig 3.

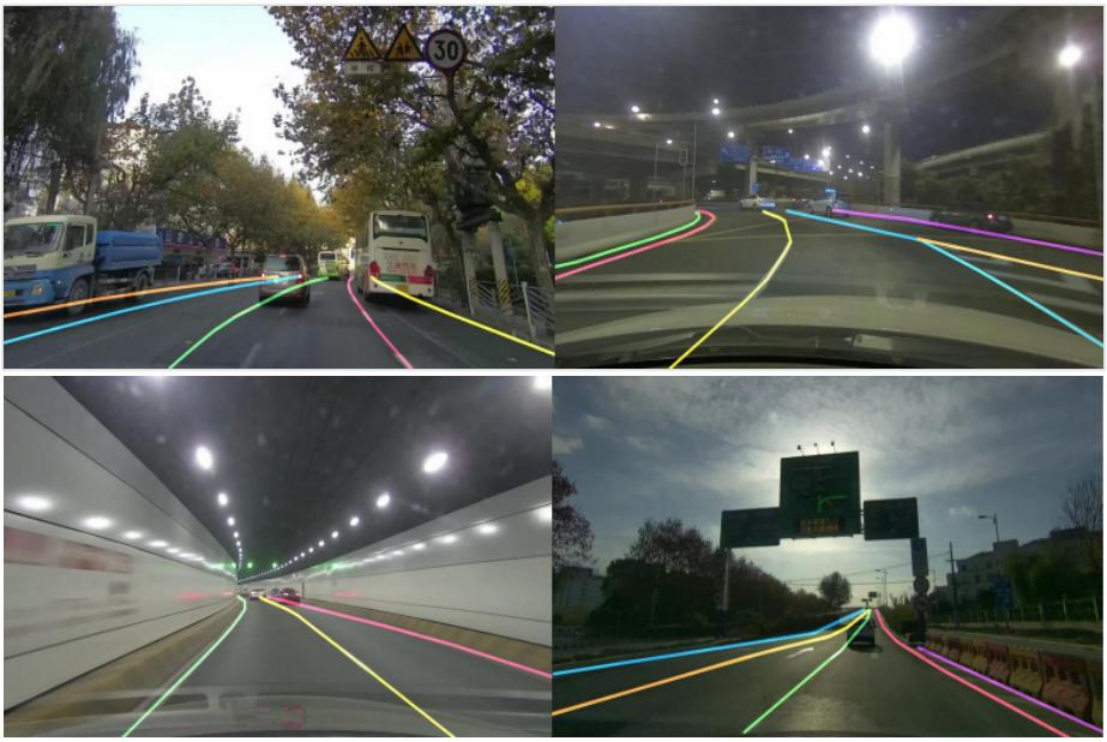


Figure 3. CurveLanes dataset

The CurveLanes dataset is a public dataset jointly released by Huawei Noah's Ark Laboratory and Sun Yat-sen University, specifically designed for lane detection tasks. Each image contains at least one curved lane, covering complex scenarios such as S-shaped curves, Y-shaped intersections, nighttime scenes, and multi-lane roads.

Among these datasets, TuSimple, CULane, and CurveLanes are the most frequently used datasets in lane detection research. TuSimple is less challenging, with most scenarios being highways. In contrast, CULane presents more complex scenarios, many located in urban Beijing, making it more difficult.

3. Detection Methods

3.1. Traditional Image Detection Methods

Traditional image processing methods segment lane line regions using edge detection and filtering, then combine them with algorithms such as Hough transform and RANSAC for lane line detection. These algorithms require manual tuning of filtering operators and adjustment of parameter curves based on the characteristics of the street scene, resulting in a large workload and poor robustness. When the driving environment changes significantly, lane line detection performance deteriorates. The mainstream approach is as follows.

3.1.1. Lane Detection Based on Hough Transform

This method is based on the idea of transforming line detection in image space to Hough parameter space. In image space, a line is represented by the polar coordinate equation ρ = xcosθ + ysinθ, where ρ is the distance from the line to the origin and θ is the angle between the line normal and the x-axis. A line in image space corresponds to a point in Hough parameter space, as shown in Fig 4. , while a point corresponds to a line in polar coordinates, as shown in Fig 4. The specific steps are as follows: First, the input image is processed using an edge detection algorithm (such as Canny edge detection) to obtain a binary image containing lane line edge information. Then, each edge point in the edge image is traversed, and a vote is taken in Hough parameter space, i.e., a count is added at the corresponding ρ and θ values. Next, the peak point with a high count is found in Hough parameter space, and its corresponding ρ and θ values represent lines in image space. Finally, the detected lines are filtered based on prior knowledge such as lane line length and angle range to obtain the final lane lines. This method is simple in principle, easy to implement, and has a certain robustness to noise, but it is computationally intensive, parameter-sensitive, and difficult to handle curved lane lines.

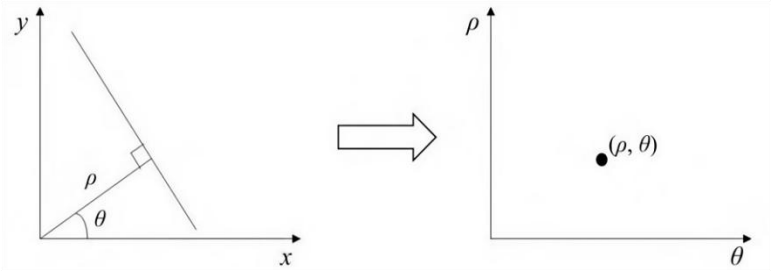


Fig 4. A straight line in polar coordinates corresponds to a point in Hough space.

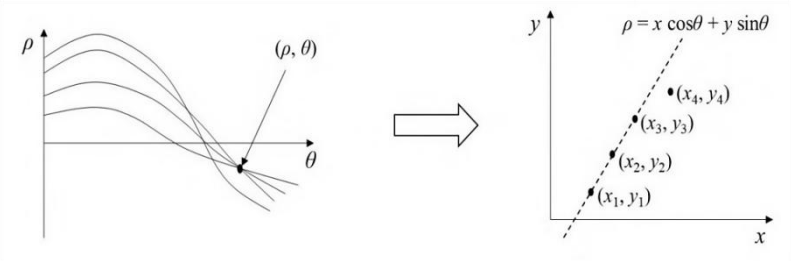
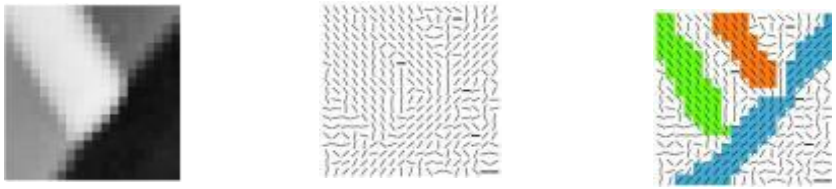


Figure 4. A point in Hough space corresponds to a straight line in polar coordinates

3.1.2. Lane Detection Based on LSD Lines

The LSD (Line Segment Detector)algorithm calculates the level-line angle of each pixel in the image, which is perpendicular to the gradient angle at that pixel. Adjacent pixels with level-line angles within a certain threshold form a line-support region. As shown in Fig 5, each line-support region is a candidate region for generating a straight line. A rectangle is fitted to this region, with the principal axis of the rectangle representing the principal axis direction of the region. The width, height, total number of pixels within the region, and the number of pixels with level-line angles within ±22.5 are integrated to calculate the NFA (Number of False Alarms) value for each rectangular region. When the NFA is less than 1, the rectangle is considered a line segment. Compared to traditional line detection algorithms, LSD has the advantages of accurate results, controllable false detections, and no need for parameter adjustment. This algorithm is used to detect straight lines in the image, providing a baseline direction for subsequent search of lane candidate points. When using LSD to detect straight lines in the lower half of the image, slope control was applied to the detected lines,

and some obviously erroneous lines outside the lane lines were removed. The direction of the longest remaining line was set as the initial search direction. As shown in Fig 6, the green lines represent the filtered line portion, and the red lines represent the determined initial search direction.



(a)Original image (b) Level-line region (c) Generated line-support region

Figure 5. Schematic diagram of generating line-support area

Figure 6. Selected lane lines and initial search direction

3.1.3. Lane Line Detection Based on Top-View Transformation

This method utilizes perspective transformation to convert an image from a perspective view to a top-view view. We convert the image into a bird's-eye view, where lane lines are mostly parallel straight lines, simplifying the detection process. The principle is to project the image from one viewpoint to another using a 3×3 transformation matrix. The steps are as follows: First, calculate the perspective transformation matrix based on four corresponding points in the image, such as lane line vanishing points and specific points on the image edge, and four corresponding points in the top-view view. Then, transform the input image using this matrix to obtain the top-view view. Next, detect lane lines in the top-view view using a simple straight-line detection algorithm. Finally, convert the detected lane line parameters back to the original image coordinate system using inverse perspective transformation to obtain the final lane lines, as shown in Figure 7. This method simplifies the detection model in the top-view view, improving accuracy and stability, and has some ability to handle curved lane lines. However, the calculation of the perspective transformation matrix depends on accurate corresponding points, and the transformation may not be entirely ideal in complex scenes.

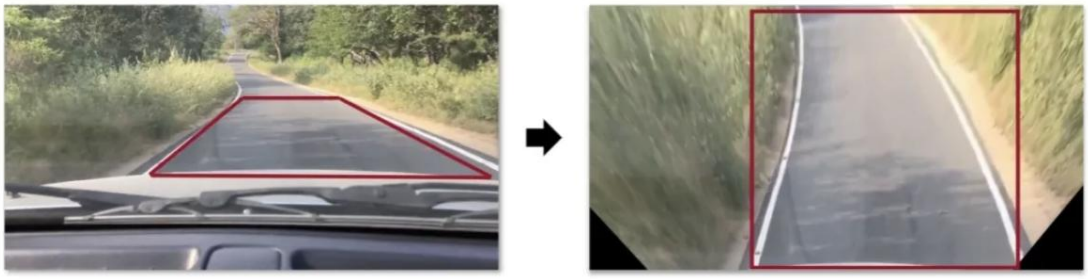


Figure 7. Left: Before is original image;

Right: After picture is transform the original image into a bird's-eye view

3.1.4. Lane Line Detection Based on Fitting

The core principle of this method is based on the difference in grayscale gradient between lane lines and the road surface. Since lane lines are white and the road surface is gray, there is a stable grayscale gradient difference between them. By setting a reasonable threshold, the edges of the lane lines can be extracted. There are many edge points extracted. The method involves finding the edge points of adjacent lane lines at the same horizontal position and taking their midpoint as a point on the lane line. This process is repeated to obtain points for the entire lane line. Because the grayscale values of the lane lines and the road surface are affected by color changes, the edge points segmented by a single threshold are not located in the middle of the lane line, but rather within a region. The set of extracted midpoints of the lane lines is not on a straight line. Instead of being on top, these points are distributed on both sides of the straight line. To obtain the final lane line, these points need to be fitted. For adjacent left and right edge points at the same horizontal position, the midpoint is taken as the projection point of the lane line to eliminate positioning deviation. However, when affected by changes in lighting or road surface color, the edge points of the threshold segmentation will be scattered in the area on both sides of the real lane line, resulting in the midpoint set showing a discrete distribution on both sides of the straight line instead of an ideal straight line, as shown in Fig 8. Therefore, it is necessary to use fitting functions such as least squares, RANSAC, or spline curves to smooth the discrete points and finally generate a continuous lane line equation. The advantage of this method is that the computation is small and it can fit lane lines with curvature. The disadvantage is that it has poor environmental adaptability, is greatly affected by lighting, and has poor stability.

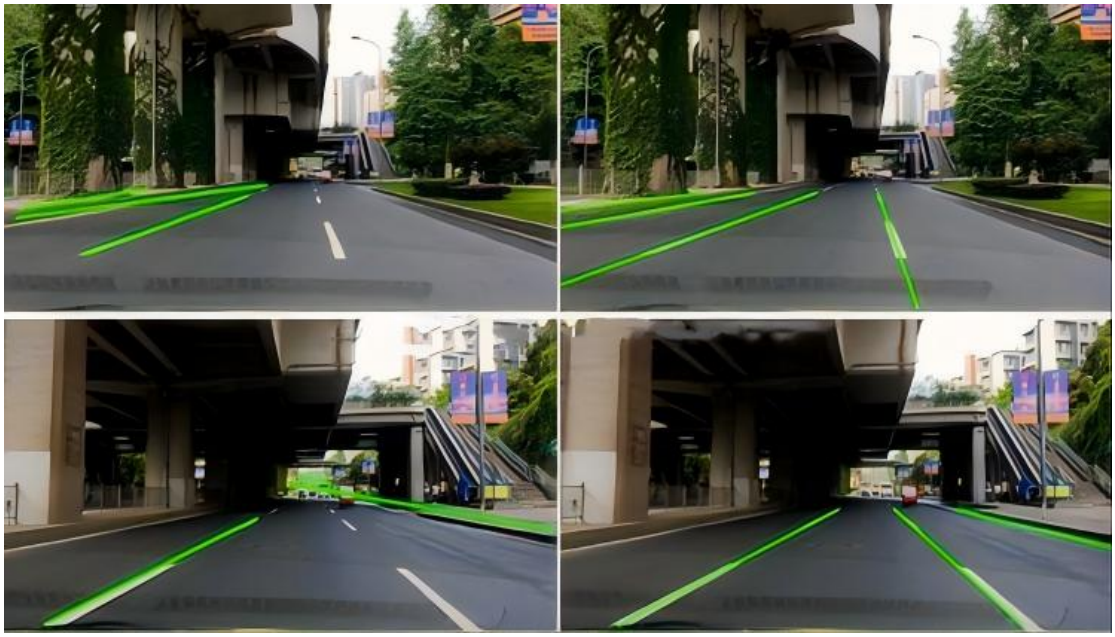


Figure 8. Lane line detection features based on fitting

3.1.5. Lane Line Detection Based on Parallel Perspective Vanishing Points

In perspective projection, a set of parallel lines intersect at a vanishing point on the image plane. Lane lines intersect at a vanishing point on both sides at a distance and exhibit perspective relationships. This method utilizes this characteristic to detect lane lines. The steps are as follows: First, detect straight line segments in the image and estimate the vanishing point position of the lane line using their intersection points; then, determine the lane line direction based on the vanishing point position and prior knowledge of the lane line; next, search for lane line edge points in the image along the determined direction; finally, fit the searched edge points using a fitting method to obtain the lane line equation. This method utilizes the perspective geometry of lane lines to improve detection accuracy and robustness, and has a certain adaptability to complex scenes. However, the accuracy of vanishing point estimation affects the detection results, and the algorithm implementation is also relatively complex.

Of course, the above mainstream methods also have drawbacks: First, their application scenarios are limited; the Hough line detection method is accurate but cannot detect curves; the fitting method can detect curves but is unstable; affine transformation can perform multi-lane detection but suffers from severe interference under occlusion and other conditions. Perspective transformation operations have certain specific requirements for the camera. The image needs to be adjusted before the transformation, and the camera installation and the tilt of the road itself will affect the transformation effect(Cieślik *et al.,* 2024). Secondly, these methods cannot meet the real-time requirements. Although the mainstream traditional methods still have some applications in specific scenarios, they have been gradually replaced by end-to-end detection schemes based on deep learning.

3.2. Deep Learning Methods

Deep learning methods have rapidly gained attention due to their robustness and real-time performance, and can be broadly categorized into four types: segmentation-based methods, detection-based methods, parametric curve-based methods, and keypoint-based methods. A brief introduction to these four types follows.

3.2.1. Segmentation-Based Methods

Segmentation-based lane detection techniques model lane detection as a pixel-by-pixel classification problem. The core principle is to classify each pixel of the input image using a neural network to determine whether it belongs to a lane region. These methods are typically based on semantic segmentation model architectures, extracting image features and generating high-resolution segmentation masks in an encoder-decoder structure. The encoder progressively reduces spatial resolution through convolutional and pooling layers to capture contextual information, while the decoder gradually recovers spatial details through upsampling and feature fusion, ultimately outputting a segmentation result of the same size as the input image. Lane pixel values are labeled as specific categories, while background pixels are labeled as other categories. Some methods also add an instance discrimination head to distinguish different lane instance values through clustering or embedding vectors.

Among these, mask-based modeling treats lane detection as an instance segmentation task. Its core features are reflected in three aspects: first, it possesses pixel-level instance discrimination capabilities, accurately separating lane lines from the background and effectively identifying different lane instance values within the same image. Typical methods like LaneNet employ a dual-branch structure: the lane segmentation branch generates semantic segmentation masks to distinguish lane line regions, while the lane embedding branch outputs pixel-level feature vectors. Clustering algorithms like Mean-Shift are used to aggregate pixels with similar features into different instances, achieving instance-level segmentation. Secondly, it supports high-precision boundary localization, accurately depicting lane line boundary contours through pixel-by-pixel classification, making it particularly suitable for scenarios with stringent requirements for lane line shape accuracy, such as autonomous driving path planning. Thirdly, it exhibits adaptability to complex scenes, effectively handling complex situations such as partially occluded or intersecting lane lines through instance-level feature differences, ensuring accurate differentiation and detection of different lane lines.

Meanwhile, Grids-based modeling, by dividing the image into a regular grid and predicting the position or existence of lane lines within each grid cell, offers a unique technological advantage. Its core features are first reflected in its structured output. This method transforms lane detection into a classification or regression task of grid cells, predicting whether a lane line passes through a horizontal strip and regressing its specific location. This design naturally supports the differentiation of multiple lane line instances, with different lane lines producing differentiated prediction results in different grid cells. Second, gridding significantly improves computational efficiency. By decomposing the global detection problem into independent predictions of local grid cells, the computational load is greatly reduced. For example, the UFAST method predicts the position of the lane line in each horizontal row, achieving fast row selection based on global features, which is particularly suitable for scenarios with stringent real-time requirements, such as autonomous driving. Finally, this method fully utilizes the geometric prior knowledge of lane lines. Its grid division method is highly consistent with the spatial structure of lane lines extending from near to far from the driver's perspective. Through gridding design, the model's ability to model spatial features such as the continuity and directionality of lane lines is effectively enhanced.

Classic lane detection techniques based on segmentation include SCNN(Pan *et al.,*2024), RESA[3], LaneNet[4], etc.

SCNN detection technology，to distinguish different lane lines, SCNN(Pan *et al.,*2024) treats different lane lines as different categories, thus transforming lane detection into a multi-class segmentation task. A slice CNN structure is proposed to enable message transmission

across rows and columns.Its schematic diagram is shown in Fig 9.

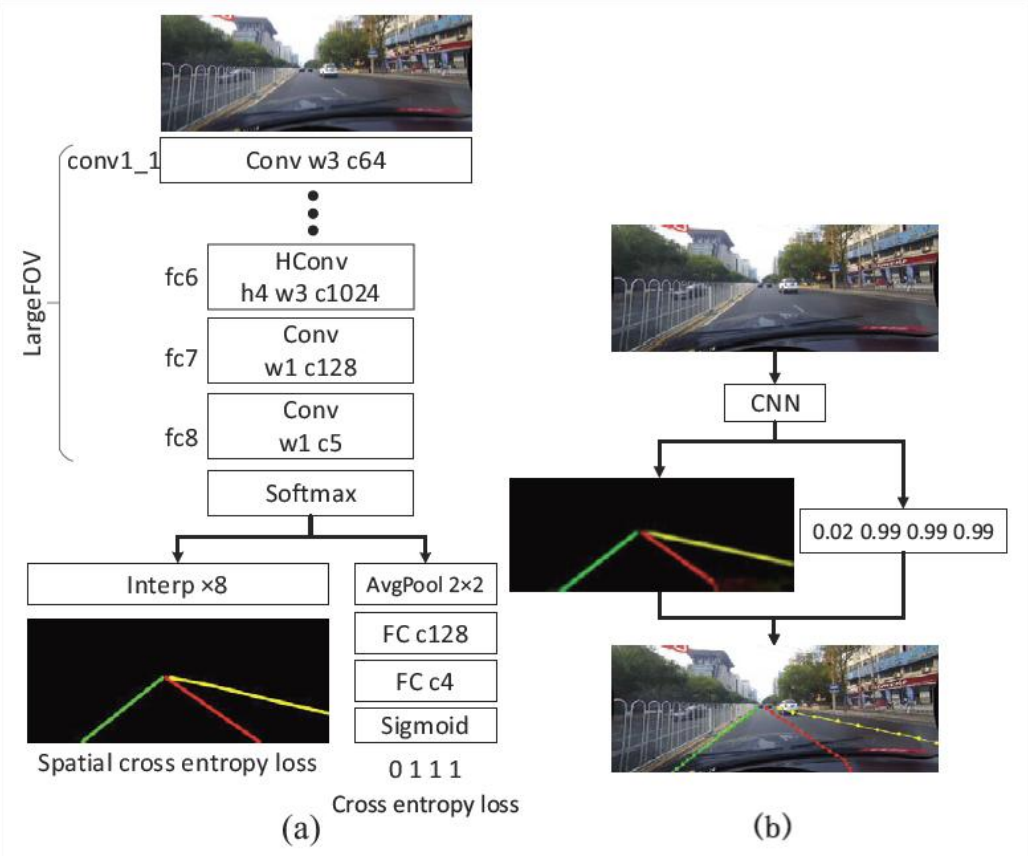


Figure 9. SCNN multi-class segmentation task

RESA detection technology improves the SCNN slice CNN structure, as shown in Fig 10. It adds information transmission of different amplitude sizes between slices, while decoupling the temporal dependencies between adjacent layers and increasing parallel processing capabilities .The structure is shown in Fig 11.

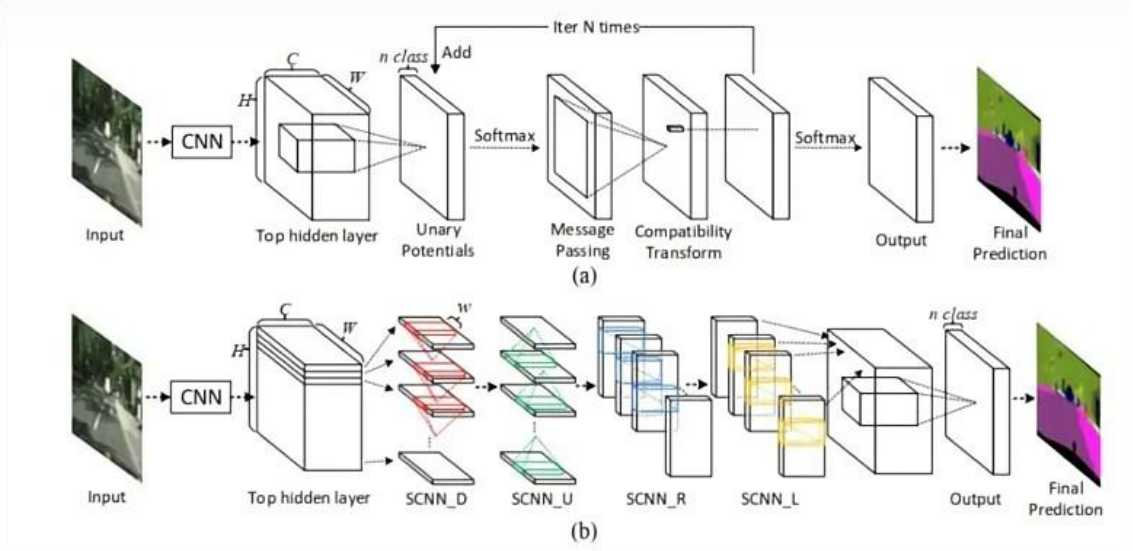


Figure 10. Slice CNN structure in SCNN

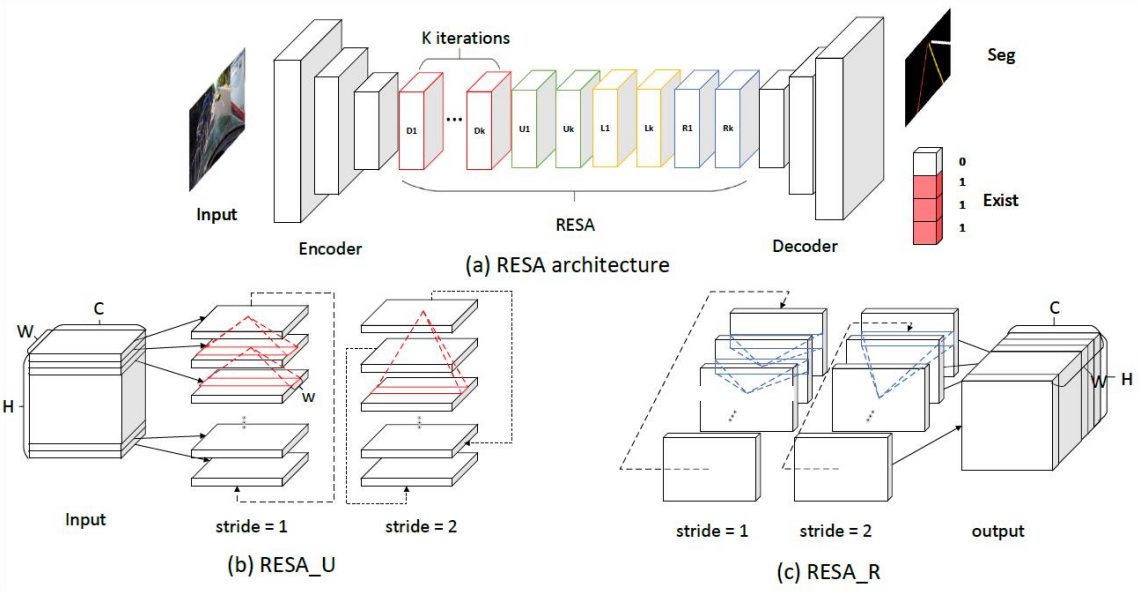


Figure 11. Schematic diagram of RESA structure

LaneNet is a lane detection method based on instance segmentation. Its core innovation lies in the dual-branch network design: the lane segmentation branch uses lightweight semantic segmentation models such as ENet to output lane line probability maps, and the lane embedding branch outputs pixel-level feature vectors. Pixels with similar feature vectors are aggregated into different lane line instances through the Mean-Shift clustering algorithm, achieving instance-level segmentation of lane lines. This method can handle any number of lane lines, but its performance degrades under severe occlusion, and the post-clustering processing steps increase computational complexity.

Lane detection based on segmentation, a classic paradigm in deep learning, achieves fine-grained differentiation between lane lines and the background through pixel-level classification. SCNN technology effectively captures the long-range continuity of lane lines by introducing spatial convolutions on feature maps to propagate contextual information; however, its multi-stage training process and dense convolutional operations result in a massive number of model parameters, such as hundreds of MB in the ResNet-101 backbone network, and inference speeds typically below 10 FPS, making it difficult to meet real-time requirements. While RESA reduces computational redundancy through recursive feature translation, it still relies on high-resolution input in complex scenes, further exacerbating the computational burden. LaneNet uses a dual-branch structure for instance segmentation, but its encoder-decoder architecture consumes significant hardware resources and has low deployment efficiency on embedded devices.

These methods share common drawbacks: high model complexity, relying on deep networks to extract high-level semantic features, leading to a surge in parameters and computational cost; significant real-time bottlenecks, especially when processing high-resolution images, where the frame rate is difficult to reach the 30 FPS or higher required by autonomous driving systems; insufficient robustness to occlusion scenarios, where pixel-level segmentation is prone to breakage or false detections due to missing local information when lane lines are severely occluded by vehicles, shadows, or road markings; and a lack of prior knowledge utilization, failing to explicitly model the geometric constraints or topological relationships of lane lines, requiring reliance on large amounts of data-driven learning, resulting in a sharp drop in performance when data distribution shifts. For example, in the curved lane detection task on the CurveLanes dataset, SCNN's F1 score is 12% lower than

curve-fitting-based methods because it does not explicitly encode curve priors; and LaneNet's false detection rate is 23% higher than improved schemes incorporating attention mechanisms in low-light nighttime scenes because it does not incorporate illumination invariance constraints.

3.2.2. Detection-based methods

Detection-based methods typically employ a top-down approach to predict lane lines. These methods utilize prior knowledge of lane lines extending from near to far in the driver's viewpoint to construct lane line instances. Anchor-based methods design linear anchors and regress the offset between sampling points and predefined anchors. Non-maximum suppression (NMS) is applied to select the lane line with the highest confidence. Related works include LineCNN[5] and LaneATT[6].

The LineCNN detection method uses straight rays emanating from the image boundary in a specific direction as a set of anchors. The model structure diagram is shown in Fig 12.

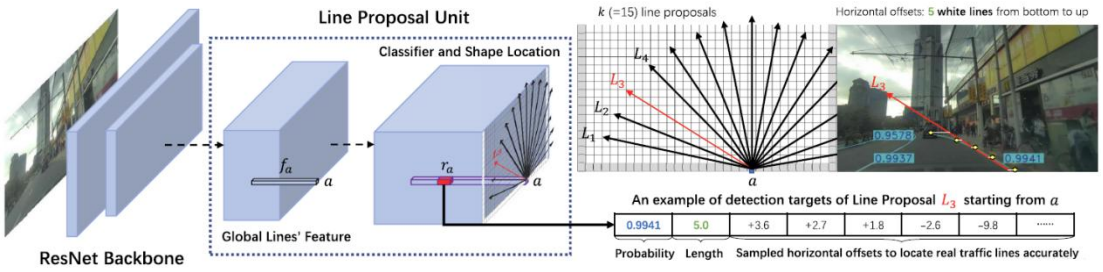


Figure 12. LineCNN model structure diagram

The LaneATT detection method proposes a pooling method based on linear anchors combined with an attention mechanism to obtain more global information. The LaneATT model is shown in Fig 13.

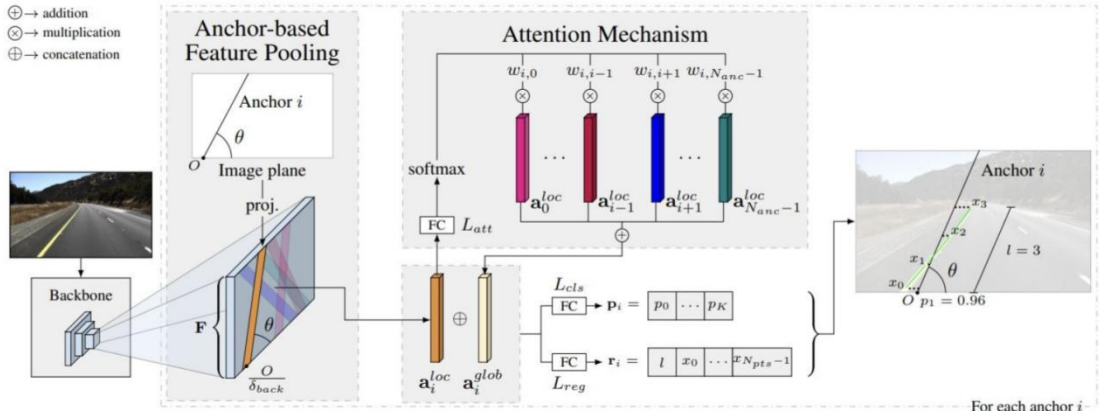


Figure 13 . Schematic diagram of LaneATT model

The method based on instance detection divides the horizontal stripes of the image into equidistant segments and detects the position of each lane line in the horizontal stripes. Related works include CondLaneNet [7], UFAST [8], etc.

The CondLaneNet detection method is a top-down lane detection framework. It first

detects lane instances and then dynamically predicts the lane shape for each instance. A schematic diagram of the CondLaneNet model is shown in Fig14.

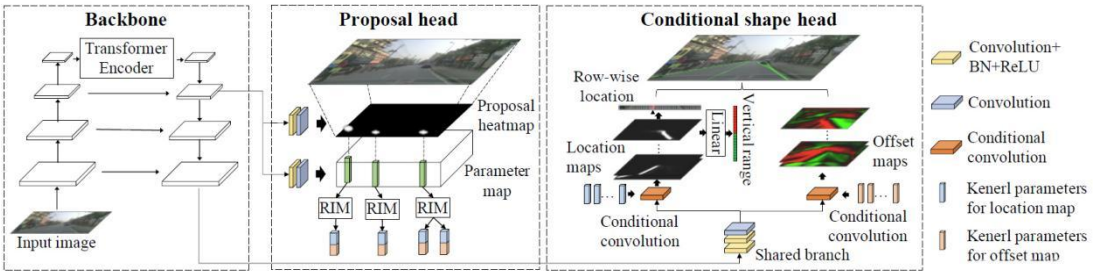


Figure 14. Schematic diagram of CondLaneNet model

The UFAST detection method treats the lane detection process as a row selection problem based on global features, which effectively reduces computational cost and increases computational speed. A lightweight version can achieve 300+ FPS. A schematic diagram of the UFAST row selection problem is shown in Fig 15.

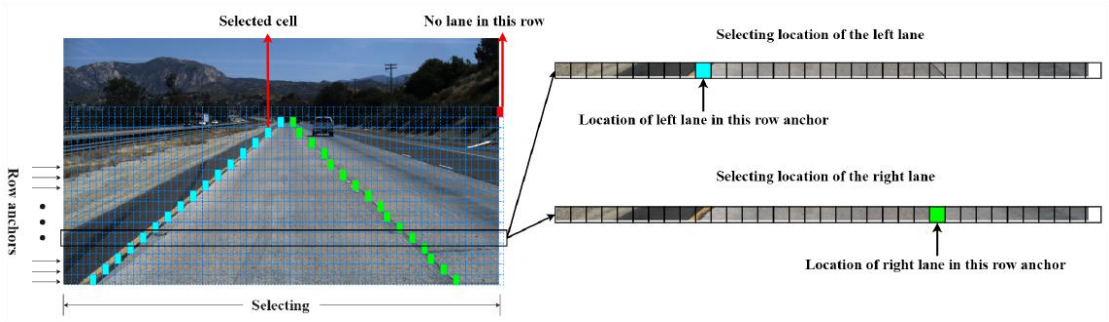


Figure 15 . Schematic diagram of the UFAST row selection problem

Detection-based methods, employing a top-down design paradigm, fully leverage the geometric priors of lane lines under perspective projection, transforming the detection task into a structured parameter prediction problem. These methods typically use pre-defined line anchors as spatial benchmarks, directly generating lane line instances by regressing keypoint offsets or geometric parameters. This significantly reduces search space complexity, effectively handling severely occluded scenarios while maintaining high real-time performance. Even with missing information in local areas, they can still generate continuous lane line structures based on global geometric constraints. However, their core limitation lies in the strong dependence on anchor design: the shape, number, and distribution of pre-defined anchors must closely match the real lane lines; otherwise, regression difficulty increases or detection flexibility decreases. To alleviate this issue, recent research has explored dynamic anchor generation, anchor-free keypoint clustering, or parametric curve modeling, further enhancing the methods' adaptability to diverse scenarios while maintaining the advantage of utilizing prior knowledge.

3.2.3. Key Point-Based Detection Method

Keypoint-based methods directly detect lane line instances and use post-processing to segment the instances. Related works include FOLOLane[9] and GANet[10].

The FOLOLane detection method models local patterns and predicts the global structure in a bottom-up manner. The FOLOLane inference framework is shown in Fig16.

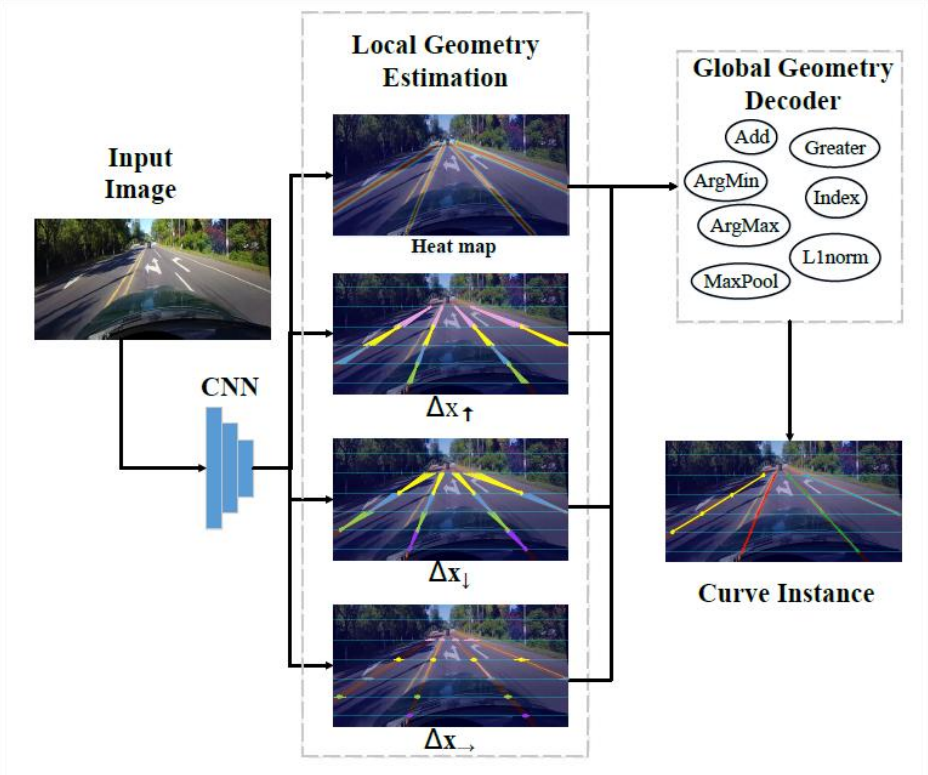


Figure 16. FOLOLane's Reasoning Framework

The GANet detection method directly regresses each keypoint to the starting point of the lane line for instance segmentation, rather than extending point by point. A schematic diagram of the GANet model is shown in Fig17.

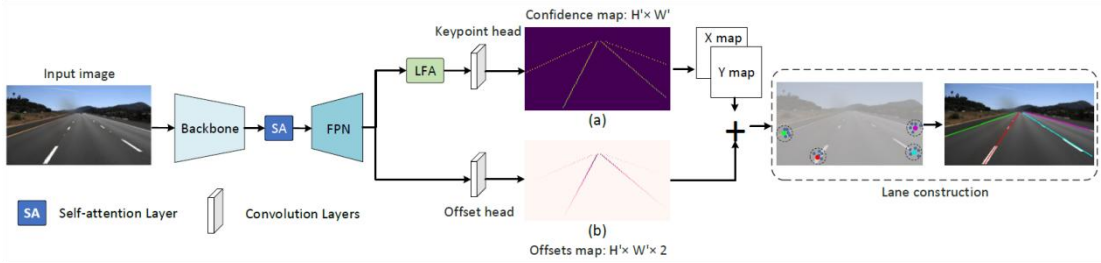


Figure 17. Schematic diagram of GANet model

Keypoint-based lane detection methods achieve a balance between flexibility and real-time performance by predicting keypoints along lane lines and constructing global relationships. Their core advantages include: high flexibility, requiring no pre-defined anchors of fixed shapes, adapting to lane lines with different curvatures, lengths, or topologies, and exhibiting stronger generalization ability in complex scenes; strong real-time performance, as the keypoint prediction head is typically a lightweight fully connected layer or a 1×1 convolution, with computationally significantly lower than segmentation or bounding box-based methods, enabling real-time inference when combined with an efficient backbone network; and local-global information fusion, where spatial constraints or attention mechanisms between keypoints allow for the inference of the overall lane line shape using undisturbed keypoints even in cases of partial occlusion.

However, a key challenge for such methods in handling severe occlusion lies in global information construction: if a large number of keypoints are missing due to occlusion, relying

solely on local features can easily lead to lane line breaks or misconnections. Existing solutions include: geometric constraint modeling, such as introducing priors like lane line parallelism and curvature continuity, and forcing keypoint distribution to conform to geometric rules through loss functions; contextual information fusion, utilizing vehicle and curb-surrounding road elements or multi-scale feature maps to supplement occluded area information; and graph structure reasoning, treating keypoints as graph nodes and propagating relationships between nodes through graph neural networks (GNNs) to achieve implicit completion of occluded areas. For example, PINet predicts keypoint heatmaps using stacked hourglass networks and designs a keypoint grouping module to construct lane line instances using spatial distance; LaneATT aggregates global features through an attention mechanism to enhance the correlation between keypoints. Future directions could further explore unsupervised/self-supervised learning to reduce dependence on labeled data, or combine the long-term dependency modeling capabilities of Transformers to improve robustness in extreme occlusion scenarios.

3.2.4. Parametric Curve-Based Methods

Parallel curve-based methods use pre-defined parametric curves to detect lane line shapes. Related works include PolyLaneNet[11] and B'ezierLaneNet[12].

The PolyLaneNet detection method uses polynomial curve regression to output a polynomial representing each lane line in the image. It maintains an efficiency of 115 FPS. A schematic diagram of the PolyLaneNet model is shown in Fig 18.

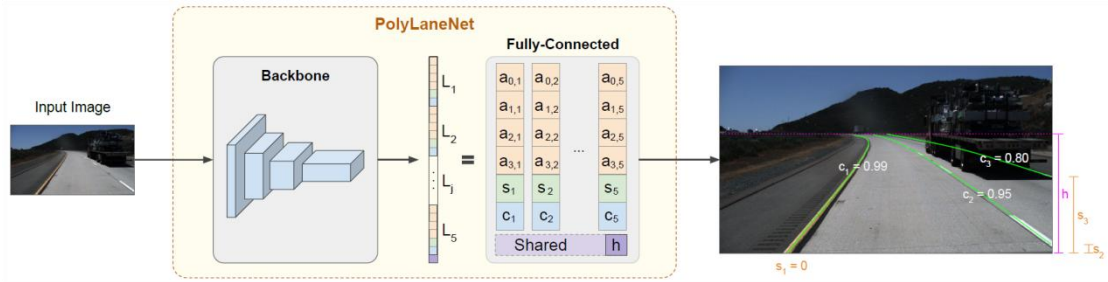


Figure 18. Schematic diagram of PolyLaneNet model

The B'ezierLaneNet detection method addresses the optimization difficulties of existing polynomial curve methods by using B'ezier curves to fit lane lines. A schematic diagram of the B'ezierLaneNet model is shown in Fig 19.

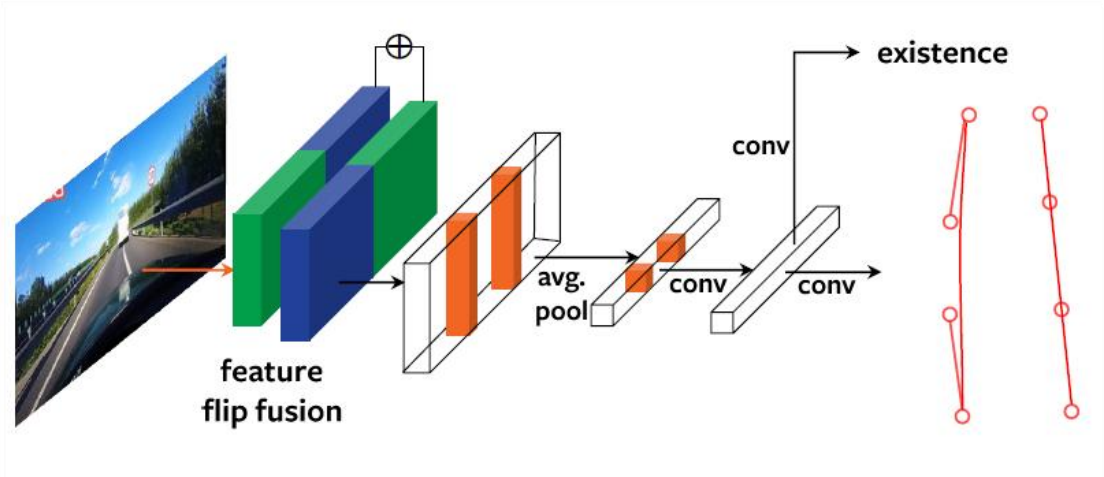


Figure 19. Schematic diagram of B'ezierLaneNet model

Curve-based lane detection methods directly fit the overall shape of lane lines using parametric curve models. Their core advantage lies in naturally modeling the global structure: curve parameters continuously represent the curvature, direction, and length of the lane line, avoiding the local fragmentation problems associated with keypoint or segmented detection, making them particularly suitable for complex scenarios such as curves and ramps. Furthermore, the computational complexity of curve fitting is typically lower than dense pixel segmentation or instance segmentation, and combined with lightweight networks, it can achieve high inference speeds. However, the accuracy of such methods is limited: on the one hand, a single curve model may struggle to accurately fit the local details of real lane lines, leading to endpoint offsets or curvature errors; on the other hand, curve parameters are sensitive to noise, and feature extraction errors are easily amplified in the parameter space by occlusion, lighting changes, or background interference, causing fitting bias. In addition, some methods require pre-setting the number of curves or their initial shape, further limiting their adaptability to varied scenarios.

4. Conclusion

The deployment of autonomous driving systems places fundamental requirements on the accuracy and real-time performance of lane detection algorithms. The development of deep learning-based lane detection algorithms has provided a practical solution for their application. From segmentation-based methods based on traditional instance segmentation to detection-based methods that combine prior knowledge for top-down modeling, the accuracy and speed of lane detection algorithms have continuously improved. How to better utilize prior knowledge of lane shape is a direction worthy of further exploration.

References

[1] Cieślik, K., Krogul, P., Muszyński, T., Przybysz, M., Rubiec, A., & Typiak, R. K. (2024). Influence of Camera Placement on UGV Teleoperation Efficiency in Complex Terrain. Applied Sciences, 14(18), 8297.

[2] Pan, Xingang, Jianping Shi, Ping Luo, Xiaogang Wang, and Xiaoou Tang. “Spatial as Deep: Spatial CNN for Traffic Scene Understanding.” Proceedings of the AAAI Conference on Artificial Intelligence 32, no. 1 (April 27, 2018). https://ojs.aaai.org/index.php/AAAI/article/view/12301.

[3] Zheng, Tu, Hao Fang, Yi Zhang, Wenjian Tang, Zheng Yang, Haifeng Liu, and Deng Cai.

"RESA: Recurrent Feature-Shift Aggregator for Lane Detection." ArXiv: 2008.13719 [Cs], March 25, 2021. http://arxiv.org/abs/2008.13719.

[4] Neven, Davy, Bert De Brabandere, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. "Towards End-to-End Lane Detection: An Instance Segmentation Approach." In 2018 IEEE Intelligent Vehicles Symposium (IV), 286–91, 2018. https://doi.org/10.1109/IVS.2018.8500547.

[5] Li, Xiang, Jun Li, Xiaolin Hu, and Jian Yang. "Line-CNN: End-to-End Traffic Line Detection With Line Proposal Unit." IEEE Transactions on Intelligent Transportation Systems 21, no. 1 (January 2020): 248–58. https://doi.org/10.1109/TITS.2019.2890870.

[6] Tabelini, Lucas, Rodrigo Berriel, Thiago M. Paixão, Claudine Badue, Alberto F. De Souza, and Thiago Oliveira-Santos. "Keep Your Eyes on the Lane: Real-Time Attention-Guided Lane Detection." ArXiv:2010.12035[Cs], November 17, 2020. http://arxiv.org/abs/2010.12035.

[7] Liu, Lizhe, Xiaohao Chen, Siyu Zhu, and Ping Tan. "CondLaneNet: A Top-to-down Lane Detection Framework Based on Conditional Convolution." ArXiv:2105.05003 [Cs], June 10, 2021. http://arxiv.org/abs/2105.05003.

[8] Qin, Zequn, Huanyu Wang, and Xi Li. "Ultra Fast Structure-Aware Deep Lane Detection." ArXiv:2004.11757 [Cs], August 4, 2020. http://arxiv.org/abs/2004.11757.

[9] Qu, Zhan, Huan Jin, Yang Zhou, Zhen Yang, and Wei Zhang. "Focus on Local: Detecting Lane Marker from Bottom Up via Key Point." arXiv, May 28, 2021. https://doi.org/10.48550/arXiv.2105.13680.

[10] Wang, Jinsheng, Yinchao Ma, Shaofei Huang, Tianrui Hui, Fei Wang, Chen Qian, and Tianzhu Zhang. "A Keypoint-Based Global Association Network for Lane Detection." ArXiv:2204.07335 [Cs], April 15, 2022. http://arxiv.org/abs/2204.07335.

[11] Tabelini, Lucas, Rodrigo Berriel, Thiago M. Paixão, Claudine Badue, Alberto F. De Souza, and Thiago Oliveira-Santos. "PolyLaneNet: Lane Estimation via Deep Polynomial Regression." ArXiv:2004.10924[Cs], July 14, 2020. http://arxiv.org/abs/2004.10924.

[12] Feng, Zhengyang, Shaohua Guo, Xin Tan, Ke Xu, Min Wang, and Lizhuang Ma. "Rethinking Efficient Lane Detection via Curve Modeling." ArXiv:2203.02431 [Cs], March 4, 2022. http://arxiv.org/abs/2203.02431.