Project Report

Deep Learning

[CSE4007]

Image Caption Generation

By

Ridit Jain

210471

Ashi Jain

210454

Department of Computer Science and Engineering

School of Engineering and Technology

BML Munjal University

Nov 2023

Declaration by the Candidates

We hereby declare that the project entitled **"Lane Detection"** has been carried out to fulfill the partial requirements for completion of the core-elective course on Deep Learning offered in the 5th Semester of the Bachelor of Technology (B.Tech) program in the Department of Computer Science and Engineering during AY-2023-24 (odd semester). This experimental work has been carried out by us and submitted to the course instructor **Dr. Soharab Hossain Shaikh**. Due acknowledgments have been made in the text of the project to all other materials used. This project has been prepared in full compliance with the requirements and constraints of the prescribed curriculum.

Ridit Jain

Ashi Jain

Place: BML Munjal University

Date: 18 November 2023

Contents

Introduction

The intersection of Artificial Intelligence (AI) and computer vision has catalyzed a transformative era in autonomous vehicle technology, with lane detection being a cornerstone application. For autonomous vehicles to navigate safely, they must be capable of identifying and following road lanes, a task that comes naturally to human drivers but presents a complex challenge for machines. The implementation of neural networks has proven pivotal in addressing the intricacies involved in recognizing lane lines amidst the myriad of real-world driving conditions.

Lane detection is pivotal not only for autonomous vehicles but also for driver assistance systems, enhancing safety and situational awareness for drivers worldwide. According to the World Health Organization (WHO), road traffic injuries are estimated to be the eighth leading cause of death globally, underscoring the urgency for advanced safety features in transportation. This project presents an innovative Lane Detection System, crafted to identify and track lane markings with high precision in diverse environments and lighting conditions.

This report delves into the systematic approach undertaken to develop the Lane Detection System, from initial conception to final implementation. It details the system's architecture, encompassing the preprocessing of road images, the employment of neural network models for feature extraction.

The report meticulously articulates the processes of data curation, model selection, training methodologies, and rigorous evaluation metrics used to quantify the system's performance. It not only showcases the successful outcomes but also provides a critical analysis of the challenges encountered, lessons learned, and the strategies deployed to overcome them.

Concluding with a comprehensive evaluation, the report validates the robustness of the Lane Detection System and explores its implications for the future of vehicular technology. It sets the stage for ongoing research, aspiring to further enhance accuracy, reduce computational costs, and ultimately contribute to the vision of safer roads for all.

Problem Statement

The project aims to develop and refine a deep learning-based solution for accurate and reliable lane detection in varying driving conditions. This involves creating an algorithm that can effectively recognize and track road lane markings, adapting to different lighting, weather conditions, and road types. The solution must be robust enough to handle complex scenarios like curved roads, faded lane markings, and occlusions caused by traffic. The ultimate goal is to enhance the safety and efficiency of autonomous and semi-autonomous vehicles, contributing to the advancement of intelligent transportation systems.

Abstract

The project report presents an innovative Lane Detection System designed to enhance the safety and efficiency of autonomous and semi-autonomous vehicles. This system, developed through a deep learning-based approach, demonstrates high precision in identifying and tracking lane markings across diverse environments and lighting conditions.

At its core, the project addresses the crucial task of lane detection, a fundamental challenge in autonomous vehicle technology. The system leverages neural network models for feature extraction, processes a meticulously curated dataset, and employs advanced training methodologies. The methodology included data curation, model selection, and rigorous evaluation metrics to quantify the system's performance.

This report encapsulates the systematic journey from the conception of the Lane Detection System to its final implementation. It highlights the successful outcomes, critical analysis of challenges, lessons learned, and strategies deployed to overcome obstacles. The comprehensive evaluation showcases the robustness of the Lane Detection System, affirming its potential to contribute significantly to the future of vehicular technology and road safety.

Literature Review

1. Real-Time Lane Detection for Driver Assistance System

1.1 Introduction

The research paper addresses the increasing traffic issues and accidents, many of which are attributed to driver negligence. The primary aim is to enhance road safety and traffic efficiency through intelligent transport systems (ITS).
The paper introduces a driver assistance system based on image processing technology. This system utilizes a camera mounted on the windshield to capture road layout images, enabling the determination of the vehicle's position relative to lane lines. The system is designed to automatically detect lane lines from these images.

1.2 Methodology

1. **Image Capture:** A camera fixed on the vehicle's windshield captures road images, serving as the primary source for lane detection.
2. **Pre-processing:** The captured images undergo pre-processing, including noise reduction and contrast enhancement, to improve lane marking visibility.
3. **Lane Marker Detection:** Key to the methodology is the detection of lane markers from the processed images, using techniques like edge detection.
4. **Hough Transform Application:** A modified version of the Hough Transform is used for line detection, tailored to enhance efficiency and accuracy for lane detection.
5. **Lane Position Analysis:** The system analyses detected lines to ascertain the vehicle's position relative to the lane markers.
6. **Warning System Integration:** An audio warning system alerts the driver of potential lane departures or proximity to lane markers.

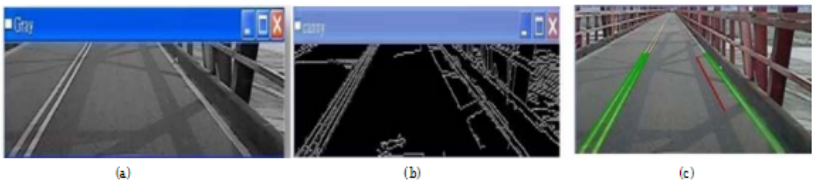


Figure 6. (a) After the process of gray image in ROI region; (b) After the process of edge detection in ROI region; (c) Prediction of road lane line.

Table 2. Filtering parameter.

Parameter	Values
experiment detected	2700 frames
The accuracy rate reaches	0.19 sec/frame
Lane Difference	<(1/10)*time of Original Image
Lane Intensity	>180

Fig.1 Edge Detection and Filtering Parameter

1.3 Result Computation

- **Accuracy:** The system's accuracy is assessed based on the correct identification of lane markers, likely presented as a percentage.
- **Processing Speed:** The processing time per frame is critical, measured in seconds per frame, indicating the system's suitability for real-time applications.

1.4 Drawbacks and Loopholes

- **Environmental Factors:** Performance can be affected by poor lighting, heavy rain, or obscured lane markers.
- **System Complexity:** Real-world implementation requires complex integration with other vehicle systems and reliability in diverse road conditions.
- **False Positives/Negatives:** Balancing the sensitivity and specificity of the system to avoid false warnings or missed detections is a significant challenge.
- **Limited Testing Conditions:** The testing conditions may not fully represent the variety of real-world driving scenarios.

2. Advanced Lane Detection Technologies for Autonomous Vehicles

2.1 Introduction

The paper begins by emphasizing the critical role of Advanced Driver Assistance Systems (ADAS) in autonomous vehicles. It identifies lane detection as a fundamental aspect of these systems, crucial for ensuring safety and efficiency in autonomous driving.

2.2 Methodology

1. **Geometric and Traditional Methods:**
    - Inverse Perspective Mapping (IPM) transforms camera images into a bird's-eye view for better lane visualization.
    - Edge detection techniques, like the Sobel operator, are used to identify lane line edges.
    - Grayscale thresholding highlights lane lines, enhancing their visibility.
    - Filtering and clustering methods separate lane line features from other road elements.
2. **AI-Based Techniques:**
    - Convolutional Neural Networks (CNNs) are applied for sophisticated image-based lane detection.
    - Recurrent Neural Networks (RNNs) and other deep learning architectures process temporal and spatial data.
    - Integration of AI with traditional methods improves accuracy and reliability.

2.3 Result Computation

- **Comparative Analysis:** Instead of experimental data, the paper conducts a comparative analysis of various methodologies reported in the literature.
- **Trends Identification:** Trends in lane detection technology are identified, such as the transition from traditional to AI-based methods and their integration.

2.4 Drawback and loopholes

- **Environmental Sensitivity:** Lane detection systems' performance varies significantly with environmental conditions like lighting, weather, and road quality.
- **Real-Time Processing:** The requirement for real-time processing in autonomous vehicles is a challenge, particularly with computationally demanding AI methods.
- **Data Dependency:** AI-based techniques rely heavily on extensive and diverse datasets for training.

- **System Integration Complexity:** Integrating these technologies into an autonomous vehicle's broader system is complex, needing considerations for reliability, safety, and efficiency.
- **Potential Bias in Studies:** The review might exhibit a publication bias, favoring more successful methodologies while underreporting less successful or less popular approaches.

3. Lane Detection and Tracking in Advanced Driver Assistance Systems

3.1 Introduction

The study underscores the pivotal role of lane detection and tracking in Advanced Driver Assistance Systems (ADAS) for enhancing road safety and reducing traffic emissions. It acknowledges the rapid advancements in autonomous vehicle technology and the crucial need for accurate lane detection systems in this evolving landscape

3.2 Methodology

1. **Feature-Based Approach:** Utilizes visual characteristics like lane line edges and markings, often employing edge detection techniques. This approach, however, is noted for its sensitivity to lighting and visibility changes.
2. **Model-Based Approach:** Applies global road models to detected features. These algorithms are more resilient against illumination changes but may struggle with atypical road geometries.
3. **Learning-Based Approach:** Integrates machine learning and AI, allowing the system to enhance lane detection accuracy through experiential learning. This approach is gaining prominence in ADAS.
4. **Algorithm Performance:** The paper contrasts these algorithms based on their efficacy under various conditions, including different lighting, weather, and road types.
5. **Environmental Factors:** A significant part of the analysis is dedicated to understanding how environmental variables impact each algorithm type. For instance, the performance of feature-based approaches is scrutinized in poor lighting conditions relative to learning-based approaches.

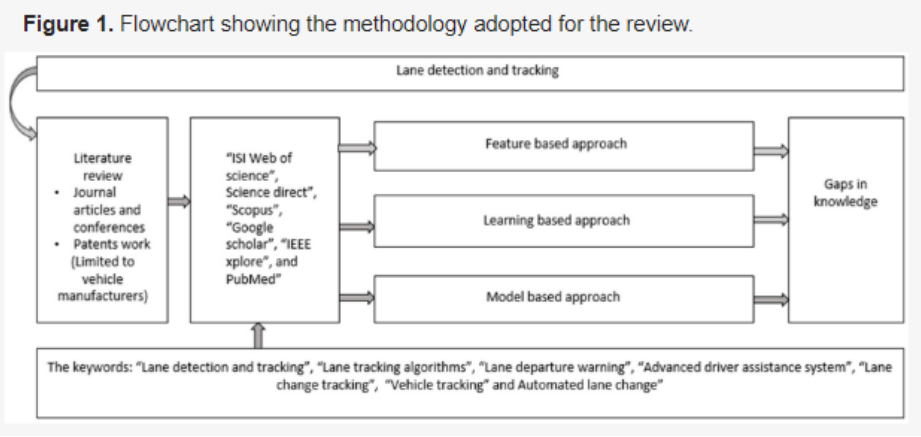


Fig. 2: Flowchart showing methodology adopted for the review

3.3 Result Computation

- **Comparative Analysis:** The paper synthesizes findings from existing studies and algorithms rather than presenting new experimental data. This comparative analysis forms the backbone of the study.
- **Performance Metrics:** The evaluation of the algorithms is based on standard performance metrics like accuracy, processing speed, and adaptability to different environmental conditions.

3.4 Drawbacks and Loophole

- **Environmental Sensitivity:** A major limitation highlighted is the susceptibility of many algorithms to environmental factors such as poor lighting, adverse weather, and road wear.
- **Real-Time Processing Challenges:** The necessity for real-time processing in dynamic driving environments is a key challenge, especially for more complex algorithms.
- **Data Dependence:** Learning-based approaches' reliance on extensive and diverse training datasets is identified as a constraint in the development of comprehensive systems.
- **Integration Complexity:** Integrating these technologies into autonomous vehicles poses significant challenges in ensuring system reliability and efficiency.
- **Potential for Bias:** The paper, being a literature review, may exhibit bias towards more successful or well-studied methodologies.

4. Lane Keeping Assist with Lane Detection

4.1 Introduction

The paper aims to demonstrate the process of simulating and generating code for a Lane Keeping Assist (LKA) controller using MATLAB and Simulink tools. This approach is intended to enhance the development and testing of LKA systems in autonomous vehicles.

4.1 Methodology

1. **Algorithm and Model Development:** The creation of a control algorithm for the LKA system is detailed, integrating lane detection data processing with a lane keeping controller from the Model Predictive Control Toolbox. A closed-loop Simulink model, incorporating synthetic data from the Automated Driving Toolbox, is employed for testing purposes.
2. **Controller Design:** The controller design accounts for scenarios where sensor data might be invalid or outside expected ranges, such as sharp curves. A synthetic lane detector simulates impairments encountered by a wide-angle monocular vision camera.
3. **Simulation and Testing:** The testing model includes an LKA subsystem to control the vehicle's front steering angle and a Vehicle and Environment subsystem to simulate the motion of the vehicle and its environment. The paper describes simulations of various driving scenarios, including assisting a distracted driver, to evaluate the algorithm's effectiveness.

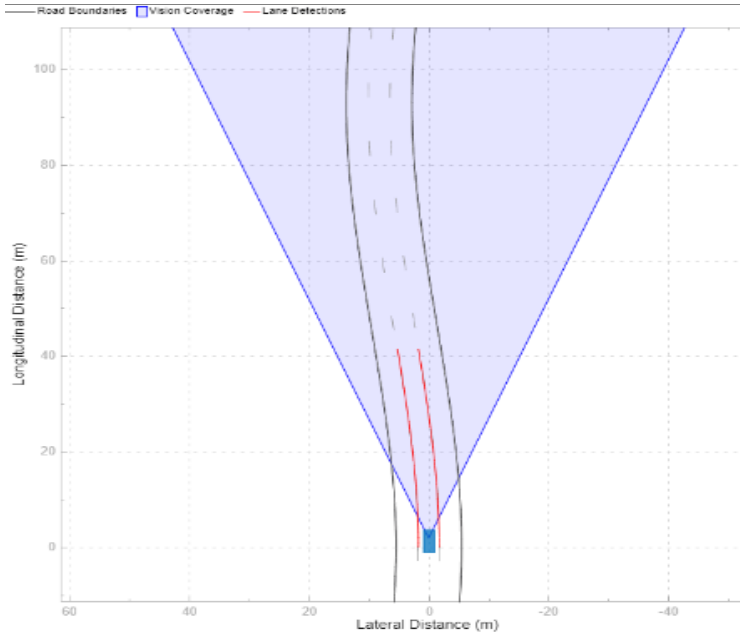


Fig. 3: Graph of longitudinal distance with lateral distance

4.2 Result Computation

- **Performance Metrics:** Lateral deviation, relative yaw angle, and steering angle measurements are used to evaluate the LKA system.
- **Visualization Tools:** The Bird's-Eye Scope from the Automated Driving Toolbox™ is utilized for visual analysis during simulations.
- **Analysis Approach:** The effectiveness of the LKA system is assessed by its ability to maintain the vehicle within its lane under different simulated conditions.

4.3 Drawback and Loopholes

- **Simulation Limitations:** The simulated environment and synthetic data may not fully capture real-world complexities, potentially limiting the validation of the LKA system in actual driving scenarios.
- **Sensor Impairments and Limitations:** The study's focus on vision camera impairments might not encompass all types of sensor inaccuracies or failures that occur in real-world conditions.
- **Scope of Testing:** The range of tested scenarios may not cover all possible driving situations, particularly more complex or unpredictable ones.
- **Real-World Implementation Challenges:** Translating findings from a simulated environment to practical implementation involves challenges like ensuring system robustness and reliability in diverse conditions.

5. A Study on Real-Time Detection Method of Lane and Vehicle for Lane Change Assistant System Using Vision System on Highway

5.1 Introduction

The paper focuses on enhancing road traffic safety and driving safety through advanced driver assistance systems (ADAS). It acknowledges the high rate of road traffic accidents and the need for improved safety measures.

5.2 Methodology

1. **Vision System Setup:**
    - The system incorporates three cameras: two placed under the right- and left-wing mirrors and one on the windscreen.
    - These cameras capture real-time images for simultaneous lane and vehicle detection.
2. **Lane Detection Process:**
    - The algorithm extracts line segments from the camera images and filters out irrelevant lines to focus on lane markings.
3. **Vehicle Detection Process:**
    - Horizontal edge filtering and Otsu's thresholding are utilized to identify potential vehicle candidates.
    - Verified against known vehicle features, a Kalman filter is then applied to track detected vehicles, enhancing accuracy.
4. **Speed and Distance Computation:**
    - The system calculates the relative speed between detected vehicles and the test vehicle.
    - It also measures the distance to other vehicles, a crucial factor for making lane change decisions.

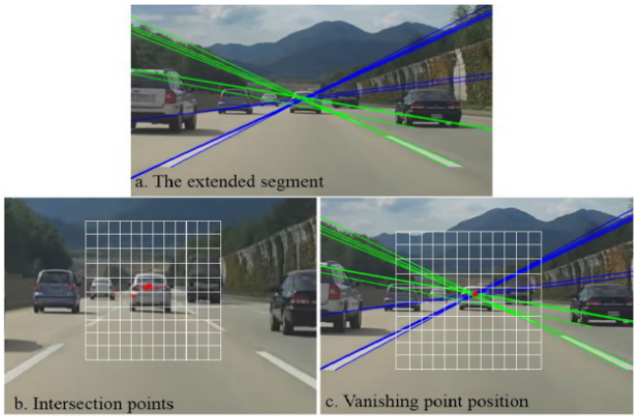


Fig.4: Marking the lane by points

5.3 Result Computation

- **Performance Metrics:** Key metrics include detection accuracy and processing speed, with an average processing time of 43 MS per frame.
- **Testing Environment:** The system was tested on highways in Korea under controlled conditions.
- **Analysis:** The effectiveness of the system is assessed based on its real-time lane and vehicle detection capabilities and the efficiency in processing images.

5.4 Drawback and Loopholes

- **Environmental Sensitivity:** The study doesn't extensively address the system's performance in varied weather conditions like rain or fog or during nighttime.
- **Hardware Limitations:** Performance evaluations are based on a specific hardware configuration, which may differ in other setups.
- **Real-World Application:** The system's performance in more diverse driving environments, such as urban areas or unstructured roads, remains untested.
- **Processing Speed:** While the processing speed is commendable, certain high-speed driving scenarios may demand even faster processing capabilities.

6. Review of Lane Detection and Tracking Algorithms in Advanced Driver Assistance System

6.1 Introduction

In vision-based lane departure warning system, a camera is placed behind the wind shield of the vehicle and images of road are captured. The lines on the road are interpreted and lanes are identified. Whenever the vehicle moves out of a lane unintentionally, a warning is given to the driver.

6.2 Methodology

1. **Feature-Based Techniques:** These methods focus on identifying specific lane line features like color, texture, or edges. Techniques such as edge detection with Sobel operators and feature extraction using Gaussian kernels are explored.
2. **Model-Based Techniques:** This approach involves creating mathematical models of lane markings, typically using parametric models like straight lines, parabolas, or splines, and fitting these models to observed lane markings in camera images.
3. **Sensor Integration:** The paper examines the use of various sensors like LIDAR, vehicle odometry, GPS, and digital maps to complement vision-based lane detection.
4. **Complex Environment Adaptation:** There's an emphasis on algorithms that operate effectively in urban environments and challenging conditions, like poor visibility or faded lane markings.

6.3 Result Computation

- **Performance Metrics:** The evaluation uses metrics such as Precision, Recall, F-score, Accuracy, ROC curves, and Dice Similarity Coefficient.
- **Comparison with Ground Truth:** Algorithms are tested against ground truth data to assess accuracy and reliability.

6.4 Drawback and Loopholes

- **Variability in Lane Markings:** Challenges arise from the variability in lane markings, affecting algorithm performance.
- **Environmental Factors:** Weather, lighting changes, and road surface quality significantly impact system effectiveness.
- **Algorithmic Limitations:** Some algorithms struggle with complex road geometries or dense urban environments.

- **Computational Demand:** The paper likely addresses the balance between algorithm complexity and computational efficiency, crucial for real-time ADAS processing.
- **Sensor Limitations:** Issues with sensor fusion and data reliability, especially in adverse conditions, are potential challenges.
- **Scalability and Adaptability:** The scalability or adaptability of some algorithms to different vehicle types or driving environments may be limited.

7. Real-Time Lane Detection and Tracking for Advanced Driver Assistance Systems

7.1 Introduction

The paper underscores the increasing necessity for autonomous vehicles in response to growing traffic levels and escalating road safety concerns.
It emphasizes the critical role of intelligent driver assistance systems, focusing on the pivotal task of lane and obstacle detection for vehicular autonomy.

7.2 Methodology

1. **Fuzzy Noise Reduction Filter (FNRF):** Applied to input images to enhance quality, the FNRF likely utilizes fuzzy logic to distinguish between noise and actual lane marking features, improving lane line visibility.
2. **Hough Transform with Defined Region of Interest:** The HT, renowned for its efficacy in line detection, is applied within a specific ROI. This targeted approach focuses on areas where lane lines are most likely present. The ROI is possibly dynamic, adjusting to variables like vehicle speed and direction.

7.3 Result Computation

- The research likely involves assessing the algorithm using metrics such as accuracy, precision, recall, and computational time.
- Performance is evaluated across various road conditions, lighting environments, and weather scenarios, potentially benchmarked against ground truth data or other lane detection standards.

7.4 Drawback and Loopholes

1. **Environmental Limitations:** The algorithm may face challenges in extreme weather or low-visibility conditions.
2. **Computational Complexity:** Despite a focus on real-time processing, the integrated FNRF and HT approach might strain computational resources.
3. **Camera Quality Dependency:** The effectiveness of the algorithm might be contingent on the quality of the camera, with lower-end cameras potentially hindering performance.
4. **Generalization Challenges:** The ability of the algorithm to adapt to different road types, traffic conditions, and vehicle varieties could be limited.
5. **Occlusion Handling:** Addressing occlusions from other vehicles or objects remains a potential hurdle.

6. **Scalability and Adaptability:** The research might touch upon the scalability of the algorithm to diverse ADAS systems and its adaptability to evolving vehicle technologies.

8. A study on detection method of vehicle based on lane detection for a driver assistance system using a camera on highway.

8.1 Introduction

The introduction highlights the importance of advanced driver assistance systems (ADAS) in enhancing road safety and reducing traffic accidents. It may delve into the challenges and recent technological advancements in vehicle and lane detection technologies.

8.2 Methodology

1. **Lane Detection:** The study likely details a specific line detection algorithm for identifying lane markers, possibly involving advanced edge detection techniques.
2. **Inverse Perspective Mapping (IPM):** It is employed for transforming captured images into a top-down view, which is essential for accurately determining lane positions and curvatures.
3. **Vehicle Detection:** Horizontal Edge Detection is used to locate potential vehicles based on horizontal edges characteristic of vehicles.
4. **Vertical Edge Detection:** It follows to confirm these detections, helping distinguish vehicles from other objects.

8.3 Result Computation

- The paper evaluates the system based on efficiency (processing time per frame) and detection accuracy.
- Performance is measured in real-world highway conditions, offering practical insights into the system's effectiveness.
- Quantitative data such as correct identification rates of vehicles and lanes compared to ground truth or manual observations are likely included.

8.4 Drawback and Loopholes

- **Environmental Conditions:** The performance in diverse weather and lighting conditions may not be fully explored, posing a limitation for real-world applicability.
- **Processing Time vs. Accuracy:** The study might reveal a trade-off between processing speed and detection accuracy, particularly in complex scenarios.
- **Camera Limitations:** System effectiveness is contingent on camera quality and positioning, where suboptimal factors can reduce detection accuracy.
- **Occlusions and Shadows:** Challenges in handling occlusions and shadows could complicate edge detection accuracy.

- **Scalability and Adaptability:** The paper might not extensively address the system's scalability across different vehicle types or adaptability to various road geometries.
- **False Positives/Negatives:** The methodology may have limitations in accurately distinguishing between vehicles and other objects, leading to potential false detections.

9. Real-Time Lane Detection for Driver Assistance System

9.1 Introduction

The paper opens with a discussion on the critical issues related to traffic and the pivotal role of intelligent transport systems (ITS) and intelligent vehicle (IV) systems.
It underscores the necessity of driver assistance systems in reducing traffic accidents and enhancing road safety, setting the stage for the study's focus on lane detection technologies.

9.2 Methodology

1. **Lane Detection:** The research utilizes a windshield-mounted camera to capture road images, addressing the challenge of detecting lane lines under diverse conditions like varying road shapes, line visibility, and environmental factors.
2. **Hough Transform:** A significant part of the paper is devoted to the use of a modified Hough Transform (HT) for lane line detection. HT, known for its line detection capabilities, is adapted to reduce computational demands, making it more suitable for real-time applications.
3. **Lane Departure Prediction:** An essential feature of the system is its ability to predict lane departures by analyzing the vehicle's position in relation to the lane markers, aiming to proactively alert drivers of potential deviations.

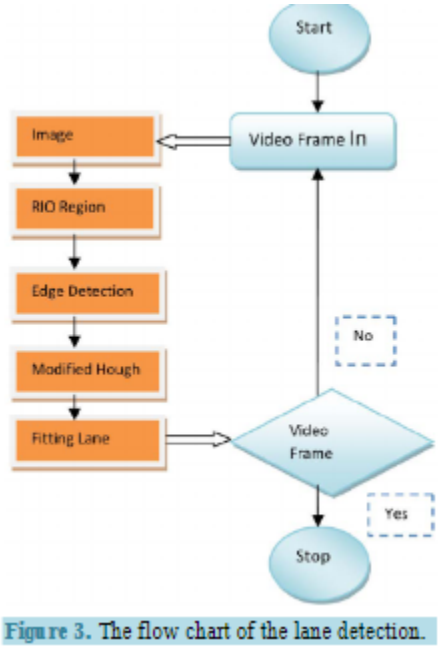


Fig. 5: The flowchart of the lane detection

9.3 Result Computation

- The system's performance is assessed on 2700 frames, showcasing a high accuracy rate of 93.8% and an impressive processing speed of approximately 0.19 seconds per frame.
- The results demonstrate the system’s effectiveness in various road conditions and lighting environments, including scenarios with low light.

9.4 Drawback and Loopholes

- **Computational Complexity:** Despite modifications, the HT may still impose significant computational demands, especially for embedded systems in vehicles.
- **Environmental Factors:** The system's response to extreme weather conditions or rapidly changing lighting, such as in tunnel driving, is not thoroughly explored.
- **Real-World Application:** The study acknowledges a gap between controlled testing environments and the unpredictability of real-world scenarios, which could impact the system’s performance.
- **Camera Dependency:** The effectiveness of the lane detection system is closely tied to the quality and positioning of the camera, suggesting potential limitations in varying hardware setups.

10. Lane Detection and Classification for Forward Collision Warning System Based on Stereo Vision

10.1 Introduction

The paper begins by underscoring the increasing relevance of autonomous driving and Advanced Driver Assistance Systems (ADAS) in mitigating traffic accidents.
It emphasizes lane detection and classification as crucial for the advancement of intelligent cars and Intelligent Transportation Systems (ITS).

10.2 Methodology

- **Stereo Vision-Based System:** The study focuses on developing a stereo vision-based system for lane detection and classification. Using two cameras, the system mimics human binocular vision to perceive depth and identify lane structures.
- **Self-Adaptive Traffic Lanes Model in Hough Space:** A robust self-adaptive model in Hough Space is utilized for lane detection, adaptable to changes in road conditions like bumpiness and lane structure variations. It uses a maximum likelihood angle and dynamic pole detection regions of interest (ROIs).
- **Lane Classification Using CNN:** Lane classification is performed using a Convolutional Neural Network (CNN), trained on manually labeled ROIs from the KITTI dataset, to identify the left/right-side line of the host lane.
- **Obstacle Mask Generation and Forward Collision Warning:** The system employs stereo matching to generate an obstacle mask, reducing interference from irrelevant objects and enhancing forward collision distance detection.

10.3 Result Computation

- The system demonstrates good true positive rates in lane detection and classification accuracy.
- It operates in real-time with a working speed of 15Hz, validating its potential for real-time ADAS applications.
- The results indicate a robustness level to environmental variations, crucial for any ADAS technology.

10.4 Drawback and Loopholes

- **Computational Load:** The computational demands of stereo vision systems, especially for real-time processing, are a significant concern.
- **Environmental Dependence:** While showing some robustness, the system's performance in extreme weather conditions remains a challenge.

- **Calibration and Alignment:** Accurate calibration and alignment of the stereo cameras are essential but complex, impacting overall performance.
- **Dataset Limitations:** The reliance on specific datasets like KITTI for training and testing may not capture the full range of real-world scenarios, potentially limiting adaptability.

11. A Lane Detection Vision Module for Driver Assistance

11.1 Introduction

The paper starts by highlighting the importance of lane detection in driver assistance systems, setting the stage for a discussion on various methodologies utilized in prior research, ranging from edge detection to neural networks and color segmentation.

11.2 Methodology

1. **Particle Filtering:** Particle filtering, a probabilistic approach, is used to generate hypotheses about the system's state. These hypotheses, represented as particles, are tested against various image filters or cues. The process involves resampling particles, diffusing them based on sensor error models, and evaluating each against the current visual information.
2. **Cue Testing:** Three image cues are implemented in the system: Canny Edge - Hough Transform Cue, LoG Edge Cue, and Color Cue. Each cue assesses how well the image information matches the expected outcomes, assigning probabilities to particles that represent possible states of the vehicle.

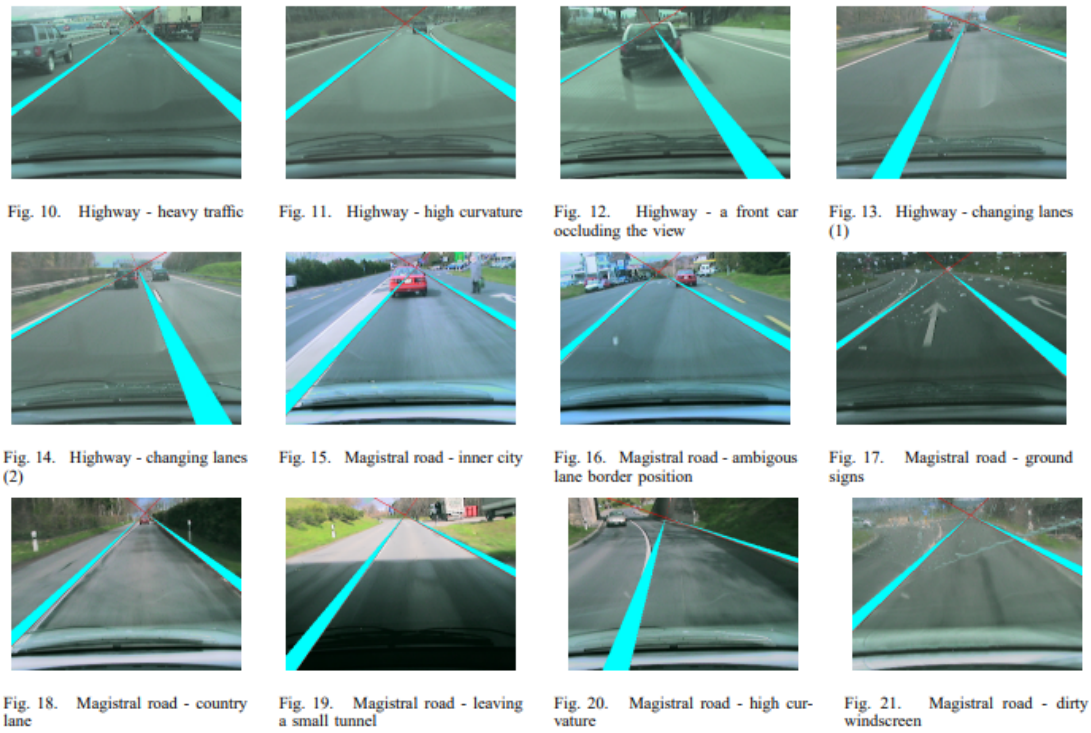


Fig. 10. Highway - heavy traffic
Fig. 11. Highway - high curvature
Fig. 12. Highway - a front car occluding the view
Fig. 13. Highway - changing lanes (1)
Fig. 14. Highway - changing lanes (2)
Fig. 15. Magistral road - inner city
Fig. 16. Magistral road - ambigous lane border position
Fig. 17. Magistral road - ground signs
Fig. 18. Magistral road - country lane
Fig. 19. Magistral road - leaving a small tunnel
Fig. 20. Magistral road - high curvature
Fig. 21. Magistral road - dirty windscreen

Fig.6: lane prediction

11.3 Result Computation

- The system was tested in a Smart Car fitted with a SONY DFW-VL500 camera, demonstrating its robustness in diverse real-road conditions.
- The combination of the Canny edge filter and Hough transform emerged as the most robust cue, effectively detecting lane boundaries across various scenarios, including roads with high curvature and in conditions with variable lighting and shadows.

11.4 Drawback and Loopholes

- The Color cue's performance is notably affected by changes in brightness and shadow, which can compromise its reliability.
- The system's reliance on a straight-line road model limits its effectiveness in scenarios involving high-curvature roads, roundabouts, and intersections.
- Potential issues are identified in the system's robustness under conditions of low visibility or extreme weather, posing challenges for reliable lane detection.

12. Lane Detection Techniques: A Review

12.1 Introduction

The paper begins by underscoring the crucial role of lane detection systems in reducing accidents caused by driver inattention. It highlights the importance of these systems in Advanced Driver Assistance Systems (ADAS) and Intelligent Transport Systems (ITS), setting the context for the need for effective lane detection methodologies.

12.2 Methodology

1. **Lane Detection Model:** The paper introduces a generalized approach to lane detection, involving the capture of road images, conversion to grayscale, and the application of various filters (such as bilateral, Gabor, and trilateral) to minimize noise.
2. **Edge and boundaries detection:** Key techniques used include the Canny edge detector for edge detection, followed by the Hough transform for identifying lane boundaries. The method culminates in fitting a pair of hyperbolas to the detected data points, effectively representing lane boundaries.
3. **Review of Lane Detection Techniques:** The study provides a comprehensive review of different lane detection algorithms, categorizing them into feature-based and model-based approaches. Feature-based methods focus on detecting low-level features like lane edges, while model-based methods employ geometric models like curves to represent lanes. The paper discusses various algorithms, including RALPH, GOLD, and AURORA, each employing techniques such as template matching, edge-based detection, and color cameras.

12.3 Result Computation

- The effectiveness of different lane detection techniques, including the Hough Transform and HSI color model-based methods, is evaluated across various scenarios.
- The paper details the strengths and weaknesses of these methods in different road conditions and environmental settings.

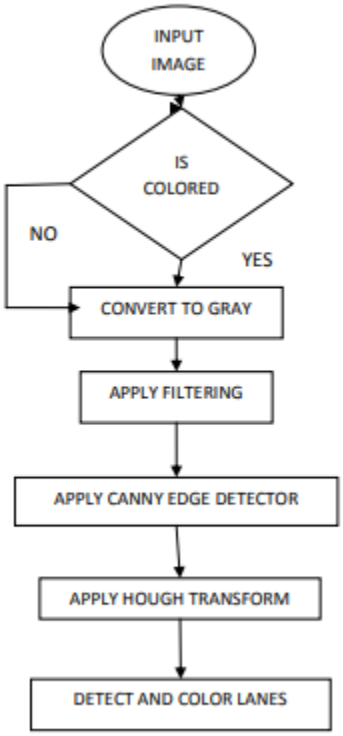


Figure 2: Algorithm of Lane Detection

Fig.7: Algorithm of Lane Detection

12.4 Drawback and Loopholes

- The research identifies significant gaps in current studies, particularly in achieving accuracy under challenging environmental conditions and in handling curved roads.
- An over-reliance on standard methods like the Hough transform is noted, suggesting a need for innovative modifications and advancements.

13. Lane detection technique based on perspective transformation and histogram analysis for self-driving cars

13.1 Introduction

The paper is focused on developing a lane detection algorithm using only vision or camera data, aiming to improve the perception capabilities of self-driving cars.
The primary objective is to showcase an effective end-to-end lane detection method leveraging contemporary computer vision techniques.

13.2 Methodology

- The study initially introduces a basic approach centered on edge detection and polynomial regression for detecting straight lane lines.
- It then advances to a more sophisticated lane detection technique combining perspective transformations and histogram analysis, which can identify both straight and curved lane lines.
- The system utilizes an RGB camera mounted at the front of the vehicle for capturing road images.
- Initial steps involve color-space analysis for lane isolation, followed by the application of a Canny edge filter and the Hough transform for lane detection.
- The advanced methodology incorporates preprocessing techniques and innovative methods that include perspective transformation and histogram analysis.

13.3 Result Computation

- The proposed lane detection algorithm is compared with traditional methods under various conditions, such as different lighting scenarios and road surfaces.
- The findings indicate that the new algorithm exhibits greater robustness and adaptability to environmental changes compared to the simpler approach.
- A comparative analysis in the paper highlights the superiority of the proposed algorithm in detecting lanes in a range of road conditions, outperforming conventional methods.

13.4 Future Work

The advanced lane-finding method employing perspective transformation and histogram-based analysis addresses these shortcomings, showing greater resilience to environmental variations.

14. Lane Detection with Moving Vehicles Using Color Information

14.1 Introduction

The paper discusses the crucial role of lane detection in Advanced Driver Assistance Systems (ADAS) and its impact on reducing road accidents and enhancing safety.
It highlights the complexity of lane detection in urban streets compared to highways, particularly in the context of moving vehicles.

14.2 Methodology

1. The study introduces an algorithm leveraging color information for detecting lane markers, designed to be resilient to changes in illumination and unaffected by the presence of vehicles on the road.
2. An adaptive region of interest (ROI) is set up, focusing on the bottom half of the image where most lane information is present. The method involves applying segmentation with a fixed threshold, followed by morphological operations to clarify lane markings and eliminate image noise.
3. The algorithm identifies potential lane markers as connected rectangles and calculates their eccentricity. Rectangles with an eccentricity close to 1, indicative of straight lines, are identified as lane markings.
4. This detection is confined to the established ROI for focused and effective lane identification.

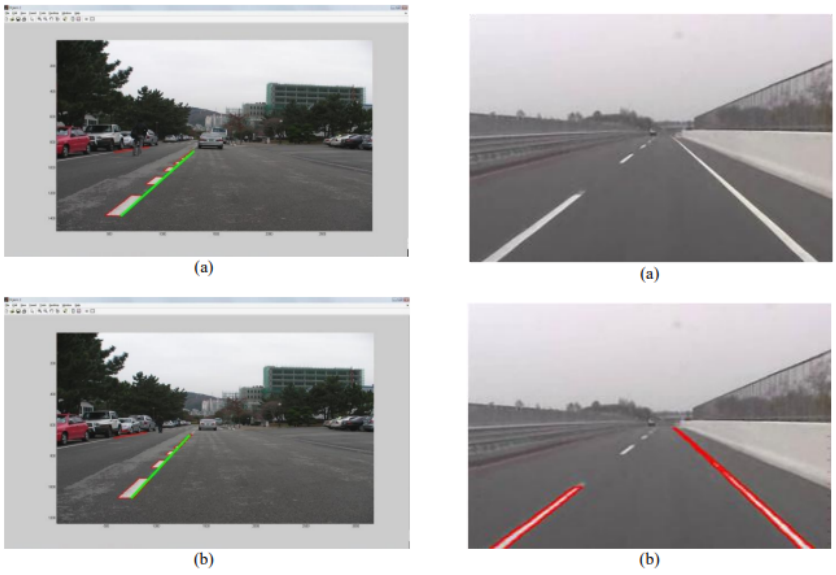


Fig.8: prediction of lane

14.3 Result Computation

- The algorithm underwent testing using RGB images captured by a digital camera, covering distances from 10 to 50 meters.
- A high detection rate of 98.8% indicates the effectiveness of the segmentation and morphological operations in accurately identifying lanes within the specified search area.

14.4 Limitation and Future Work

- The study acknowledges difficulties in detecting lanes under adverse weather conditions like heavy rain and in managing sharp curves in the image foreground.
- The stability of captured images due to vehicle movement is also identified as a challenge, pointing towards the need for further improvements in the algorithm to handle dynamic driving scenarios.

15. Advances in Vision based Lane Detection Algorithm Based on Reliable Lane Markings

15.1 Introduction

The paper starts by highlighting the growing importance of embedded systems in vehicles, emphasizing their role in ensuring human safety on roadways.
It discusses the significance of vision-based driver assistance systems and their contribution to the evolution of intelligent driver assistance systems and the development of driverless vehicles.

15.2 Methodology

- The study introduces a vision-based lane detection algorithm that initially processes RGB images into grayscale to eliminate effects like shades and immersion data while retaining image brightness.
- The next step in the algorithm involves noise reduction using a Gaussian filter.
- A critical component of the algorithm is edge detection, for which the canny edge detection method is utilized.
- Post edge detection, the algorithm employs the Hough transform technique. This method transforms image pixels into long straight lines in Hough space, facilitating lane detection in complex driving scenarios.

15.3 Result Computation

The system is designed to provide a dependable means for lane detection, thereby enhancing the safety features of vehicles, with a specific focus on autonomous driving technologies.

15.4 Future Work

- The paper concludes by underscoring the need for ongoing improvement in driver assistance systems, extending beyond automatic control systems in vehicles.
- It advocates for future advancements and refinements in the proposed methods to better address the dynamic and varied aspects of real-world driving environments.

16. A Precise Lane Detection Algorithm Based on Top View Image Transformation and Least-Square Approaches

16.1 Introduction

The paper begins by underscoring the growing importance of self-driving technology and the essential role of lane detection and keeping capabilities in Advanced Driver Assistance Systems (ADAS). It highlights the challenges faced in developing efficient lane detection methods, particularly for roads with curved and complex trajectories.

16.2 Methodology

1. **Algorithm Description:** Top View Image Transformation: The algorithm initiates by transforming road images from a front view to a top view perspective, thereby reducing distortion and simplifying the representation of lane structures.
2. **Lane Detection Methodology:** The transformed image is split into two sections: near and far. The near section applies a straight-line model for lane detection, while the far section uses a curved lane model.
3. **Hough Transform and Parabolic Model:** In the near section, a Hough transform method is employed for the straight-line model, and the curved lane in the far section is represented using a parabolic model.
4. **Least-Square Method:** The algorithm utilizes the least-square method to estimate the parameters of the curved lane, aiming to achieve precise lane detection.

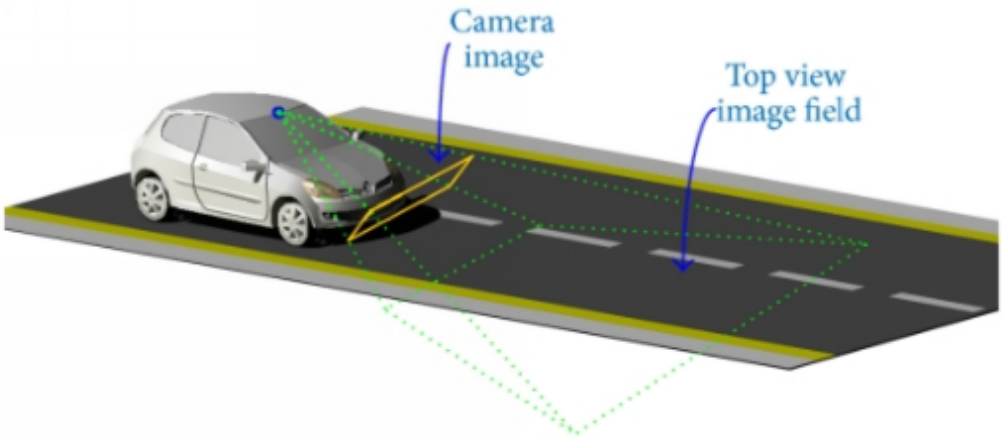


Fig.9: Image showing the top view of the field by camera

16.3 Result Computation

- The proposed algorithm's efficacy is confirmed through various experiments, demonstrating its effectiveness in accurately estimating both sharp and curved lanes.
- This approach results in enhanced precision in lane detection, a crucial factor for the functionality of autonomous vehicles.

16.4 Future Work

- The paper concludes that the combination of top view image transformation and least-square methods offers a precise and reliable solution for lane detection in complex driving environments.
- Future research is suggested to focus on further refining the algorithm, improving its accuracy and adaptability across diverse road conditions.

17. Robust lane detection and object tracking In relation to the intelligence transport system

17.1 Introduction

The research emphasizes the importance of safety in transportation systems and the role of Advanced Driver Assistance Systems (ADAS) in enhancing road safety.
It discusses the challenges of detecting road lanes and obstacles, especially in changing road conditions and varying environments. The focus is on improving lane detection and object detection in vehicles using advanced technologies to reduce road accidents and enhance safety.

17.2 Methodology

- The paper introduces a heuristic method for lane detection in videos.
- The approach begins with thresholding each frame to highlight the brightest regions, which are crucial for detecting lanes.
- Problematic aspects of existing methods, like Hough-based detection and color-based detection, are addressed. These methods can fail to detect essential lines or are sensitive to scene conditions.
- The heuristic method employs clustering to group detected points and fits a best-fit line in the mean squares sense to these points.
- The approach includes checks for coherence, such as merging lines that are too close, keeping lines like those in previous frames, and eliminating lines with significant jumps from one frame to another.

17.3 Result Computation

- The heuristic algorithm's performance is significantly better than methods solely based on the Hough Line Transform.
- This improvement is attributed to the algorithm's ability to effectively utilize data extracted from the video stream, enhancing the overall efficiency of lane detection.

17.4 Drawback and Loopholes

- **Algorithm Complexity:** The heuristic approach may involve complex calculations and processing steps, which could pose challenges for real-time application in vehicles.

- **Dependency on Camera Quality:** The effectiveness of the lane detection system largely depends on the quality and positioning of the camera. Lower-quality cameras or suboptimal positioning might result in reduced accuracy of the lane detection.
- **Data Processing and Noise Reduction:** The method involves steps like color segmentation and noise reduction, which may not always be effective in differentiating between actual lane markers and other similar patterns on the road.
- **Scalability and Adaptation:** The paper doesn't discuss how well the algorithm scales to different vehicle types or adapts to varying road geometries and driving scenarios.

18. Robust lane detection and object tracking

18.1 Introduction

The paper is focused on developing robust lane detection and object tracking methodologies for intelligent transportation systems.
It aims to improve the detection of road lines and objects (like vehicles) under various scene conditions using advanced algorithms

18.2 Methodology

- The research compares two main algorithms: the standard Hough-based line algorithm and a proposed heuristic algorithm.
- The heuristic algorithm utilizes a clustering methodology in the least square sense for detecting white lines, aiming to enhance the detection efficiency of road lines.
- For vehicle detection, the study explores the use of Gabor features, showcasing their superiority over other functions like PCA and wavelet features.
- The vehicle detection system involves extracting various features and training classifiers such as neural networks (NNs) and support vector machines (SVMs) with these features.

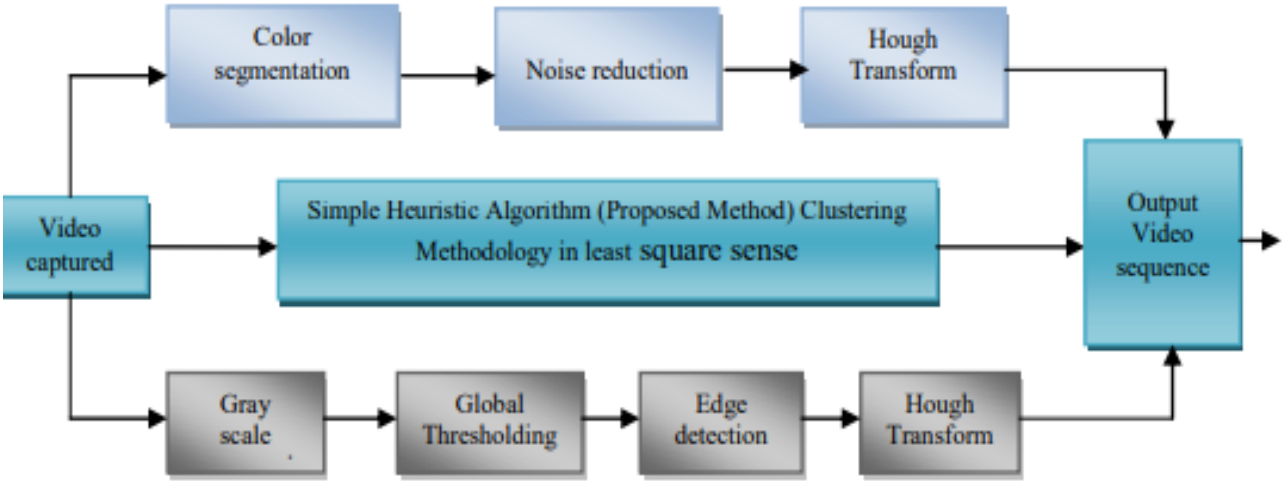


Figure 9 Block diagram of lane detection methodology

Fig. 10: Block Diagram of lane Detection

18.3 Result Computation

- The paper compares the performance of the heuristic algorithm with the standard Hough-based line algorithm in different scene conditions.

- The heuristic algorithm demonstrates greater efficiency in detecting road lines compared to the Hough-based method. In some cases, the Hough-based method failed to detect certain lines or detected unwanted lines, whereas the proposed method improved detection efficiency considerably.

.

18.4 Drawback and Loopholes

- A key challenge identified in the vehicle detection system is selecting appropriate characteristics for feature extraction and classification. Many extracted features can be redundant or irrelevant, leading to suboptimal classification performance.
- In the use of the Extended Gabor Filter Optimization (EGFO) method for vehicle detection, a significant drawback is incorrect detection under severe occlusion, such as the detection of signboards on the road instead of vehicles.

19. CNN based lane detection with instance segmentation in edge-cloud computing

19.1 Introduction

The paper focuses on improving traditional lane detection methods by convolutional neural networks (CNN) and cloud computing.
It aims to overcome the limitations of traditional lane detection, which relies heavily on feature extraction and high-definition imaging, by using a distributed computing architecture to enhance data processing efficiency.
The study proposes a method combining cloud data processing with edge computing to reduce the computing load and improve real-time performance in lane detection

19.2 Methodology

1. **CNN for Lane Detection:** Uses a CNN to process images captured by vehicle-mounted cameras, focusing on accurately detecting lane markers.
2. **Edge-Cloud Computing Integration:** Distributes computing tasks between local edge devices and cloud servers, aiming to improve data processing efficiency and real-time performance.
3. **Model Training:** Involves regularizing, cropping, and rotating input images, followed by resizing them for training. The model is optimized using a gradient descent algorithm with specific parameters for momentum and learning rate.
4. **Instance Segmentation and Clustering:** Employs instance segmentation for identifying lane markers, followed by clustering to group and accurately detect lanes.
5. **Lane Fitting Process:** Fits the detected lane markers into a lane model to provide precise lane information for driver assistance systems.

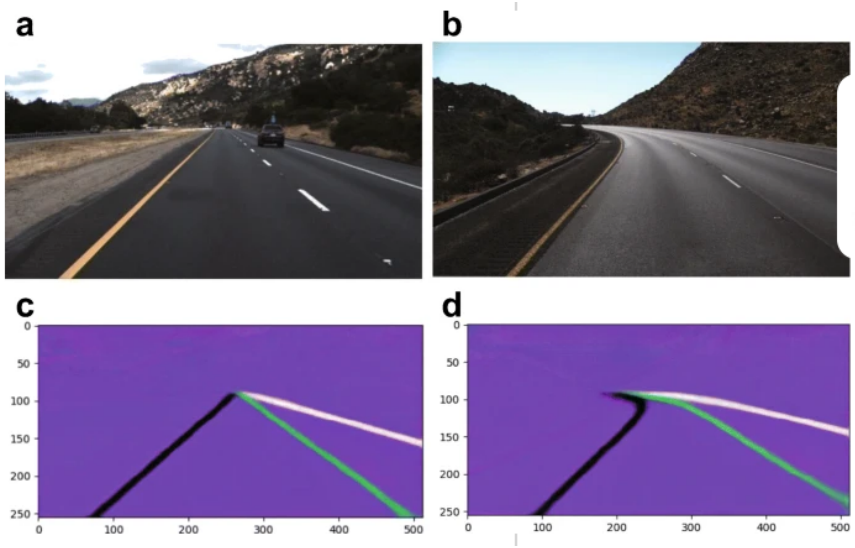


Fig.11: lane prediction of road

19.3 Result Computation

- The research used the Ubuntu 16.04 operating system with an NVIDIA GEFORCE 920M graphics processor.
- The study involved regularizing, cropping, and rotating input images, with each image set to a size of 256×512 pixels.
- The training used a gradient descent algorithm with a momentum parameter of 0.9, a weight decay of 1×10−4, and an exponential decay learning rate update strategy.
- The lane detection algorithm showed excellent performance in scenarios such as inadequate lighting, shadow occlusion, missing lane lines, and curved lanes.
- The model's lane instance segmentation and clustering process for each frame took 16.6ms, reaching 60.2 frames per second (FPS). During the lane fitting process, each frame took 2.4ms, reaching 416.7 FPS. The overall test efficiency was 52.6 FPS.

19.4 Drawback and Loopholes

- **Computational Complexity:** The combination of CNNs and edge-cloud computing might be computationally intensive, potentially challenging for real-time applications in less powerful computing environments.
- **Dependency on High-Quality Data:** The effectiveness of deep learning models like CNNs often depends on the quality and variety of training data.
- **Scalability:** The model's scalability to different types of vehicles and road conditions may not be explicitly addressed.
- **Network and Cloud Reliability:** The model's performance might be dependent on the reliability and speed of network and cloud computing resources.

Comparative Analysis

Paper Title	Methodology Highlights	Key Results	Drawbacks/Limitations	Specific Focus
Real-Time Lane Detection for Driver Assistance System	Image processing, Hough Transform	High accuracy in lane marker identification, real-time processing speed	Environmental factors, system complexity	Automated lane detection and warning system
Advanced Lane Detection Technologies for Autonomous Vehicles	Geometric methods, AI-based techniques (CNNs, RNNs)	Comparative analysis of methodologies, identification of technology trends	Environmental sensitivity, data dependency, real-time processing challenges	Lane detection in autonomous vehicles
Lane Detection and Tracking in ADAS	Feature, Model, Learning-based approaches	Comparative analysis, performance metrics evaluation	Environmental sensitivity, real-time processing challenges	Lane detection and tracking in various conditions
Lane Keeping Assist with Lane Detection	Simulation in MATLAB and Simulink, controller design	Effective lane keeping in simulated conditions	Simulation limitations, sensor impairments	LKA system development and testing
Real-Time Detection Method of Lane and Vehicle for Lane Change Assistant	Vision system with three cameras, lane and vehicle detection algorithms	High processing speed, real-time capabilities	Environmental sensitivity, hardware limitations	Lane change assistance using vision systems
Review of Lane Detection and Tracking Algorithms in ADAS	Feature-Based, Model-Based, Sensor Integration	Comparison with ground truth, performance metrics	Environmental factors, algorithmic limitations	Comprehensive review of various lane detection algorithms
Real-Time Lane Detection and Tracking for ADAS	Fuzzy Noise Reduction Filter, Hough Transform	Accurate detection across various conditions	Environmental limitations, computational complexity	Real-time lane and obstacle detection

Paper Title	Methodology Highlights	Key Results	Drawbacks/Limitations	Specific Focus
Detection Method of Vehicle Based on Lane Detection	Lane detection algorithm, vehicle detection using edge detection	Efficiency in processing time and detection accuracy	Environmental conditions, processing time vs. accuracy	Integrated vehicle and lane detection for ADAS
CNN Based Lane Detection in Edge-Cloud Computing	CNN for lane detection, edge-cloud computing integration	Excellent performance in various scenarios, high FPS	Computational complexity, dependency on high-quality data	Lane detection using CNN and distributed computing

Practical Implications

1. **Road Safety Improvement**
    - **Impact:** Enhanced lane detection systems can significantly reduce accidents caused by lane departure, a common cause of traffic incidents.
    - **Consideration:** Implementation in vehicles can help alert drivers to unintended lane departures or assist in maintaining lane discipline, especially in challenging driving conditions.
2. **Integration into Existing Vehicle Systems**
    - **Challenge:** Integrating advanced lane detection systems into existing vehicle architectures without significant modifications or cost increases.
    - **Opportunity:** Collaboration with automobile manufacturers for seamless integration, considering factors like sensor placement, system compatibility, and user interface design.
3. **Cost and Accessibility**
    - **Economic Aspect:** Advanced systems, especially those using AI and multiple sensors, might increase the cost of vehicles.
    - **Goal:** Strive for cost-effective solutions to make these safety features accessible across all vehicle segments, not just premium models.
4. **Impact on Driving Skills and Behavior**
    - **Behavioral Change:** Over-reliance on assistance systems may impact driving skills and attention levels.
5. **Adaptation to Future Transportation Trends**
    - **Trend Integration:** Aligning lane detection technologies with the evolution of autonomous vehicles and smart city infrastructure.

- **Flexibility:** Ensuring technologies are adaptable to future changes in transportation and infrastructure.

Critical Analysis

- **Strengths of the Research**

1. **Diverse Methodologies:** The papers cover a range of techniques from traditional image processing to AI-based methods, showcasing the breadth of research in lane detection.
2. **Technological Advancements:** Many studies demonstrate significant improvements in accuracy and processing speed, indicative of technological progress in this field.
3. **Real-Time Processing Focus:** The emphasis on real-time processing in several studies aligns well with the practical needs of driver assistance systems.

- **Weaknesses and Limitations of the Research**

1. **Environmental Sensitivity:** A common limitation across studies is the decreased performance under adverse weather or poor lighting conditions.
2. **Simulation vs. Real-World Testing:** Many studies rely on simulated environments, which may not accurately represent real-world complexities and driving scenarios.
3. **Data Dependency in AI Methods:** Studies employing AI techniques often require extensive, diverse datasets, which may not always be available or may introduce biases.

- **Recommendations for Future Research**

1. **Focus on Environmental Robustness:** Future studies should prioritize the development of systems that perform reliably in a variety of environmental conditions.
2. **More Real-World Testing:** Bridging the gap between simulations and real-world testing is essential for validating these technologies.
3. **Cross-Disciplinary Approaches:** Incorporating insights from fields like human factors, psychology, and urban planning could enhance the design and implementation of these systems.

4. **Addressing Data Challenges in AI:** Exploring ways to reduce data dependency and mitigate biases in AI-based lane detection systems is critical.

Description of the Dataset

CULane Dataset

Multimedia Laboratory, The Chinese University of Hong Kong

CULane is a large-scale challenging dataset for academic research on traffic lane detection. It is collected by cameras mounted on six different vehicles driven by different drivers in Beijing. More than 55 hours of videos were collected, and 133,235 frames were extracted.

This dataset features meticulous manual annotations of lane markings, including contextual annotations for obscured markings. It focuses on detecting four primary lane markings relevant in real-world applications, omitting annotations for less significant markings.

Methodology

Exploratory Data Analysis (EDA)

1. Dataset Extraction

The dataset was originally structured as a single compressed zip file. Upon extraction, the dataset revealed a hierarchical organization, comprising several individual folders. Within each of these folders, a paired set of data elements was consistently present.

1**. Image Data:** Each folder contained a collection of high-resolution road images. These images provided visual representations of diverse road scenarios and conditions.

2. **Lane Coordinate Data:** Accompanying the image files were corresponding text files. These text files contained precise lane coordinates for the respective images.

Notably, the base name of each image file corresponded directly to the name of its associated text file. This naming convention ensured a clear and unambiguous linkage between the visual road data and its corresponding lane coordinate information.

Fig12: Visualizations of few dataset images.

2. Content Analysis:

The first folder was explored, revealing several images and corresponding text files. The dimensions of the images were consistent, with each being 590 pixels in height and 1640 pixels in width. All images are in RGB format.

	Height	Width	Channels
count	60.0	60.0	60.0
mean	590.0	1640.0	3.0
std	0.0	0.0	0.0
min	590.0	1640.0	3.0
25%	590.0	1640.0	3.0
50%	590.0	1640.0	3.0
75%	590.0	1640.0	3.0

Fig13: Images size and Channels

3. **Lane Analysis:**

The lane markings in the images were examined. It was noted that in most cases, the first two or three lines in each image are categorized as 'Straight', indicating minor deviations from a linear path. Conversely, the last line in each image is often 'Curved', showing significant deviation, indicative of pronounced turns or bends.

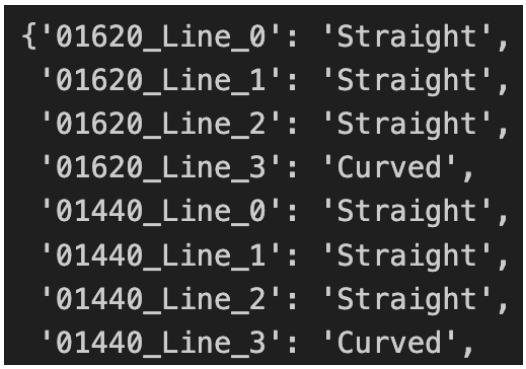


Fig 14: showing curved or straight lanes.

4. **Curvature Calculation:**

The curvature of lane markings was calculated, with values such as 6.006145822208242 and 10.85403807848081 suggesting varying degrees of curvature. Lower values indicate gentle curves, while higher values signify more pronounced curves. These curvature values are essential in understanding road nature. Roads with higher curvature values might require more cautious navigation, particularly at higher speeds, as they represent sharper turns or bends.

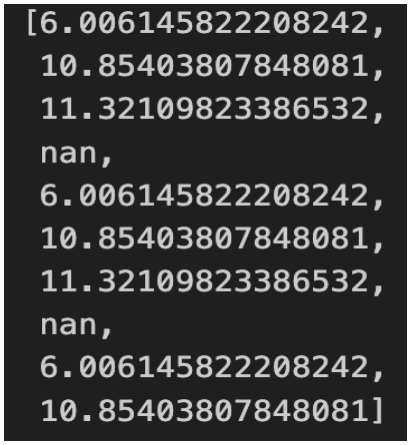


Fig 15: Showing curvature Values.

5. **Visualizations**

1. Histogram of Curvature Values

The histogram shows three distinct peaks around the curvature values of approximately 4, 6, and 10. The peaks at around 6 and 10 are higher, suggesting that lane markings with these curvature values are more frequent in your dataset. The presence of these peaks might indicate common road curvatures within the sampled data, such as gentle curves around 6 and sharper turns around 10.

The absence of bars near the curvature value of 0 suggests that there are very few or no perfectly straight lane markings in the sampled data, or that the chosen method of curvature calculation and thresholding doesn't yield values near 0. The peak around 10, which is quite high, may suggest a prevalence of road segments with significant curvature, which could be

important for applications like autonomous driving or driver assistance systems to note, as they may require more careful navigation.

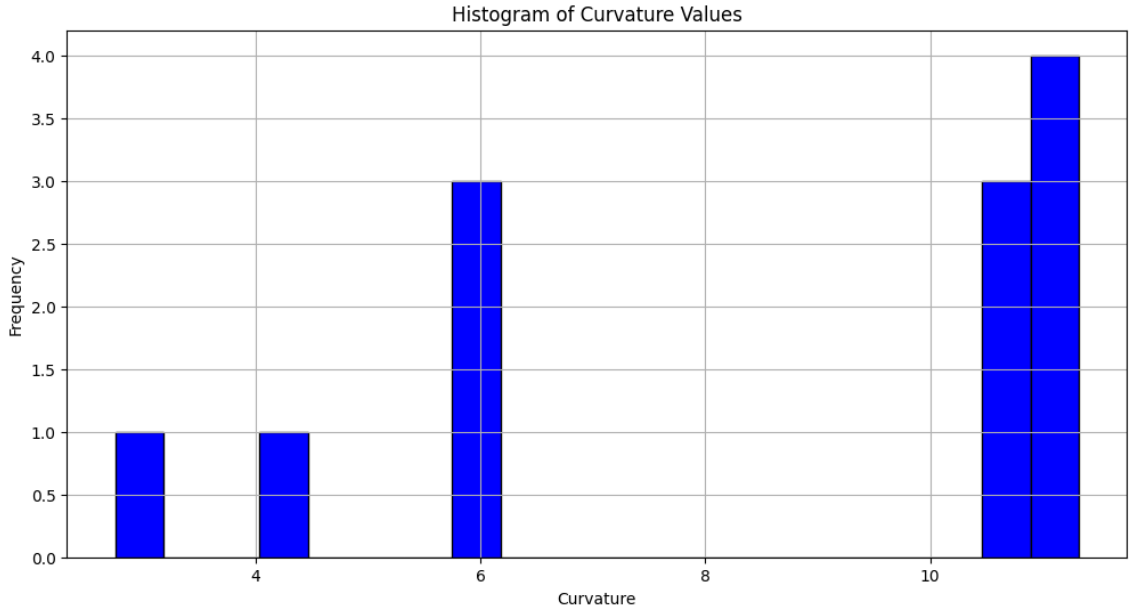


Fig. 16 Histogram of Curvature Values

2. Boxplot of Curvature Values

The boxplot indicates that the curvature values of the lane markings are mostly consistent, with a median suggesting mild curvature. There are no extreme values or outliers, implying uniformity in the curvature across the dataset. The data is concentrated within a narrow interquartile range, showing little variability in lane curvature.

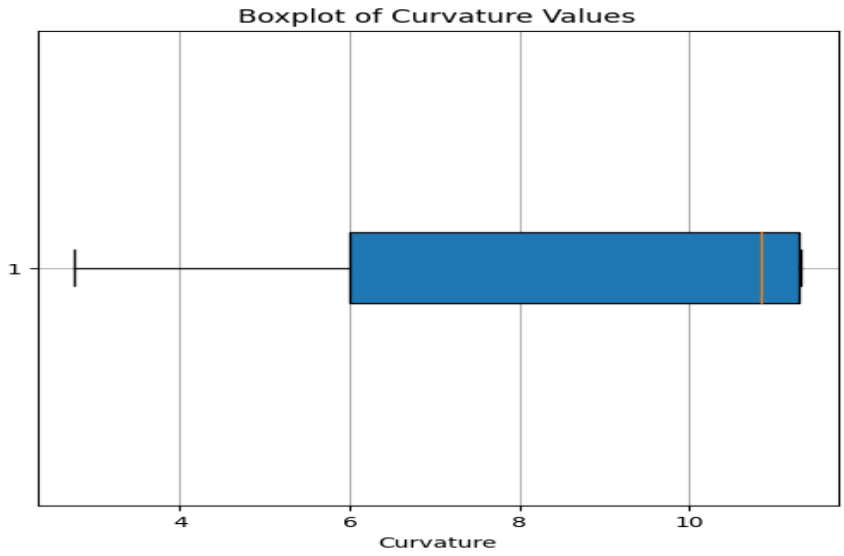


Fig. 17 Boxplot of Curvature Values

3. **Cumulative Distribution of curvature values**

The cumulative distribution plot for curvature values indicates that the frequency of lane markings steadily increases with curvature. The steeper rises in the plot correspond to curvature values where there is a higher frequency of lane markings. The majority of lane markings have a curvature of 10 or less, with the distribution plateauing as it approaches this upper bound.

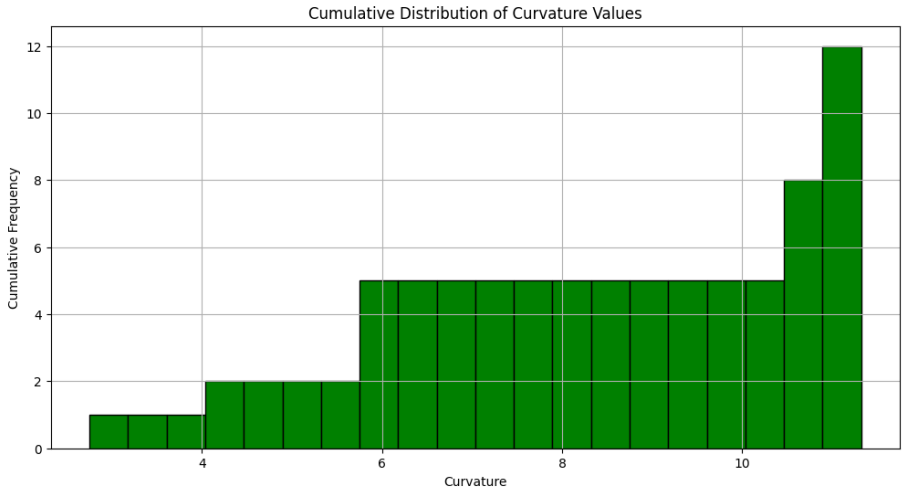


Fig. 18 Cumulative Distribution of Curvature Values

4. **Distribution of straight vs. Curved Lanes**

The bar chart displays the count of lane markings categorized as 'Curved' and 'Straight' based on their curvature values. The 'Curved' category has a higher count, indicating that there are more lane markings with curvature values greater than 10.

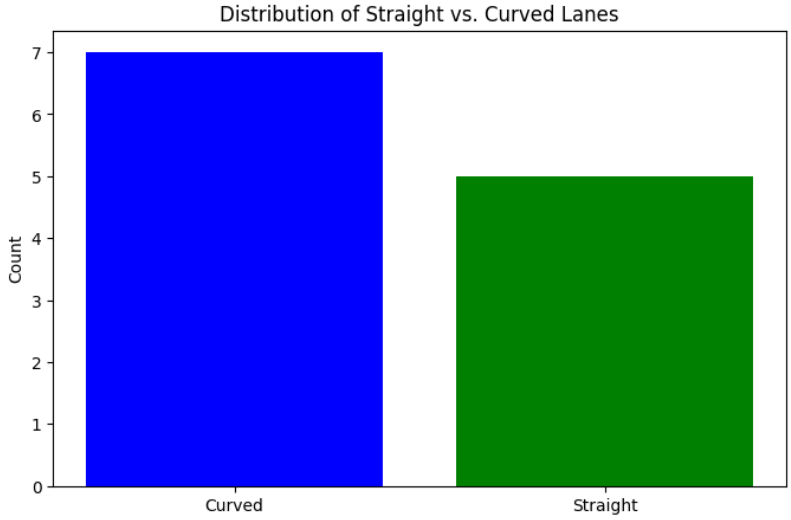


Fig. 19 Straight vs. Curved Lanes

5. **Curvature values of sample lanes**

Most of the samples (green points) are classified as 'Curved', which aligns with the previous bar chart showing a higher count of curved lanes. Fewer samples (blue points) are classified as 'Straight', indicating such lanes are less common in the dataset. The plot shows a clear distinction between the two categories based on the chosen threshold value of 10 for curvature.

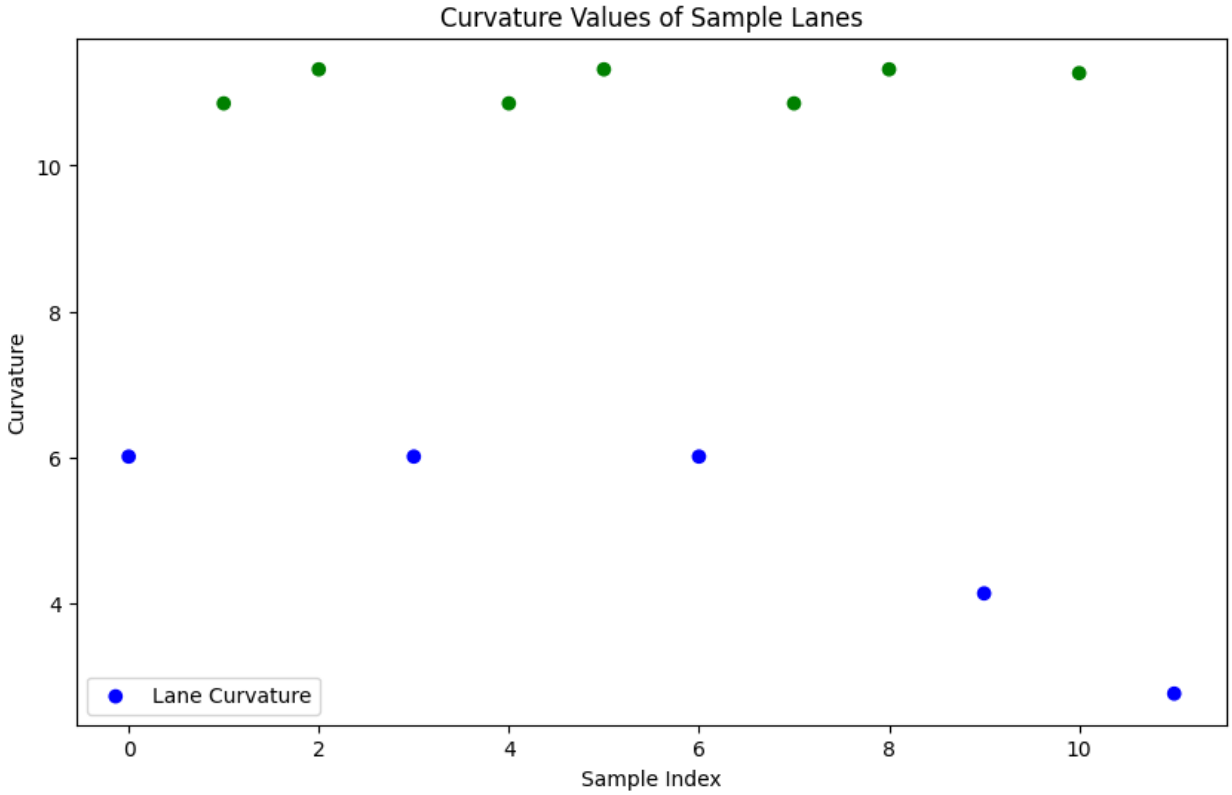


Fig. 20 Curvature Values of Sample Lanes

6. Curvature Boxplot

Curved Category: The 'Curved' boxplot is a narrow box, indicating that the curvature values for lanes classified as 'Curved' are very consistent and close to the threshold value that defines them as curved (just above 10).

Straight Category: The 'Straight' boxplot has a larger interquartile range, showing more variability in the curvature values, but all are below the threshold that classifies them as straight.

Median Lines: The line inside each box indicates the median curvature value for each category. Both medians are centrally located within their boxes, suggesting a symmetrical distribution of curvature values within both categories.

No Outliers: There appear to be no outliers in either category, as there are no points beyond the whiskers of the boxes.

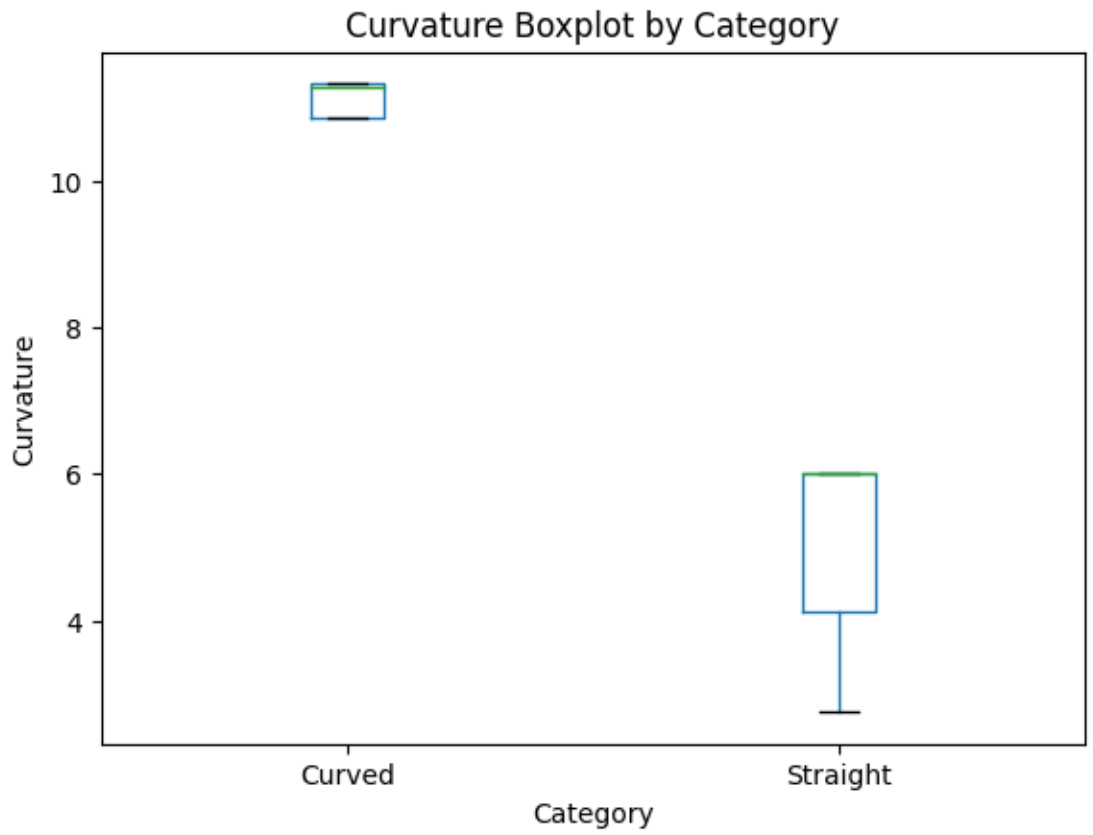


Fig. 21 Curvature Boxplot

7. **Histogram of lane Marking Length**

The histogram shows the distribution of lengths for lane markings. It indicates there are three prominent lengths at which lane markings commonly occur: around 300, 400, and 800 units (the unit of measurement isn't specified but could be meters, feet, etc., depending on the data). The lane markings are not evenly distributed across lengths; instead, they are concentrated at these three lengths, suggesting standardized lane lengths or frequent distances between features like road markings or intersections. There's a very low frequency of lane markings of intermediate lengths between these values.

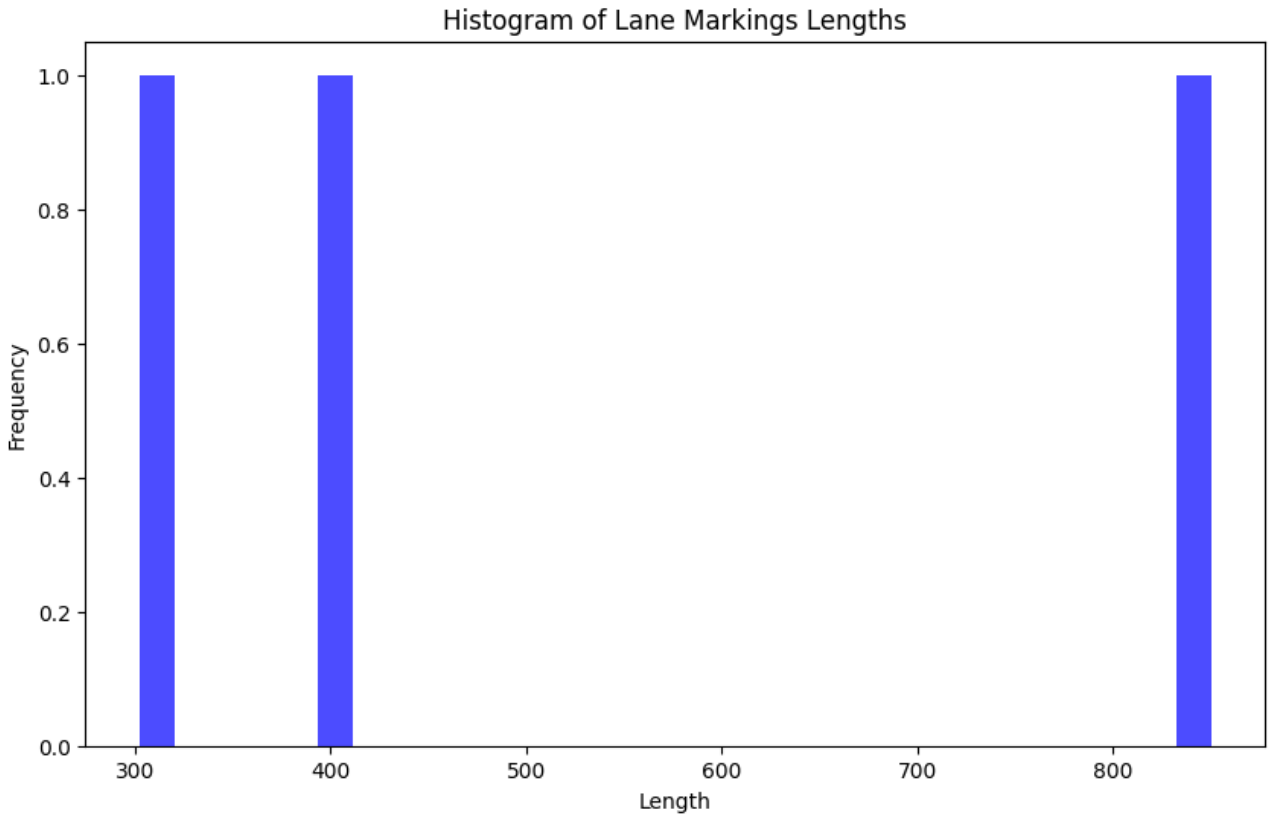


Fig.22 Histogram of Lane Markings Lengths

8. **Scatter Plot of Lane Marking Endpoints**

There's a significant distance between the start and end points of lane markings, as indicated by the separation along the x-axis. The lane markings seem to be almost horizontal, with start and end points sharing approximately the same y-coordinates but varying x-coordinates. The clustering of end points (red dots) around the lower x-coordinate values suggests that many lane markings may begin from a similar longitudinal position. The spread of the start points (green dots) across a wider range of x-coordinates indicates more variability in where lane markings end. The plot shows a directional trend in the lane markings, potentially indicating the flow of traffic or the design of the roadway where the data was collected.

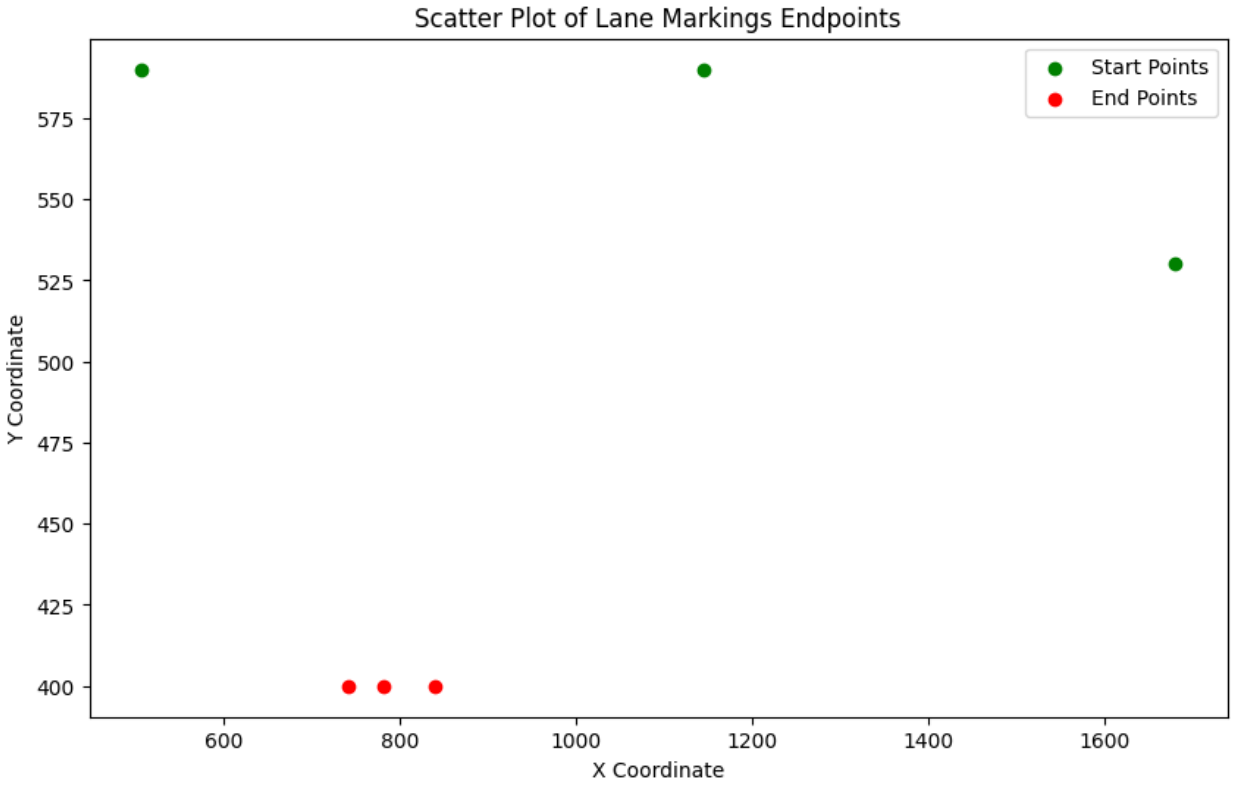


Fig.23 Scatter Plot of Lane Marking Endpoints

6. **Image Preprocessing:**

The following Steps are done like resizing the images, normalizing them, and subsequently converting the lines into masks using binary segmentation.

1. **Resizing images:** to a consistent size is crucial for deep learning because neural networks require uniform input dimensions. This ensures that the model can process all images in a consistent manner, making it easier to design and train the network.
2. **Image normalization:** involves scaling pixel values to a standardized range, typically [0, 1] or [-1, 1]. This process helps the model converge faster during training by reducing the impact of varying pixel value ranges in different images. Normalization also aids in better weight initialization and gradient flow, ultimately improving the model's ability to learn meaningful features from the data.
3. **Data augmentation:** in lane detection datasets is challenging because it can disrupt the semantic integrity of road scenes, create label inconsistencies, introduce unrealistic features, complicate model evaluation, and potentially raise ethical concerns related to altering sensitive information in images.

4. **Binary Segmentation (for Lanes):** A function creates binary masks by reading coordinates from text files and drawing lines on a zero matrix of a specified size. When this mask is applied on images, these lines represent the lanes on them.

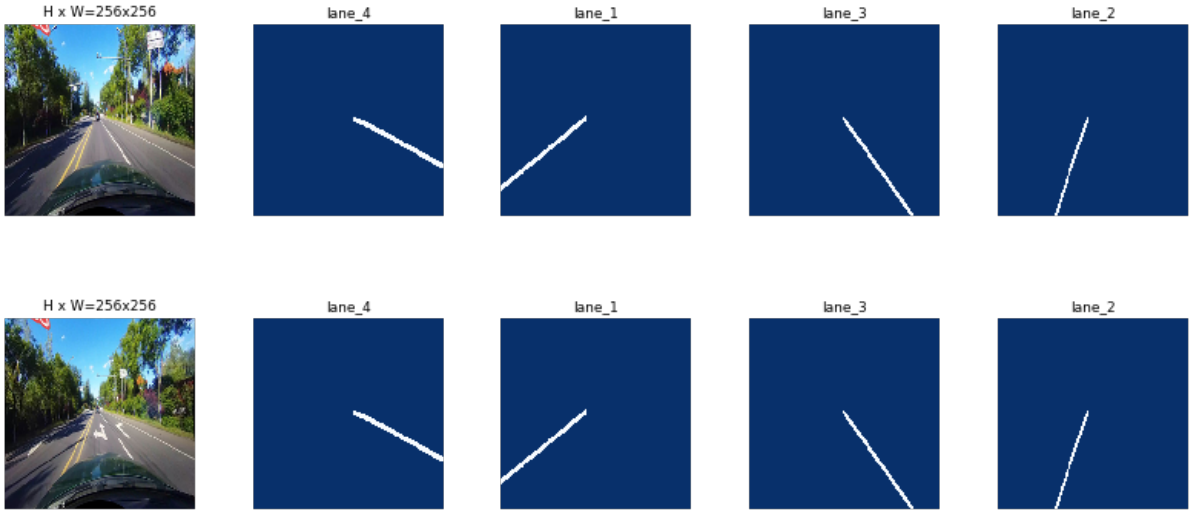


Fig.24 Images and corresponding lane mask

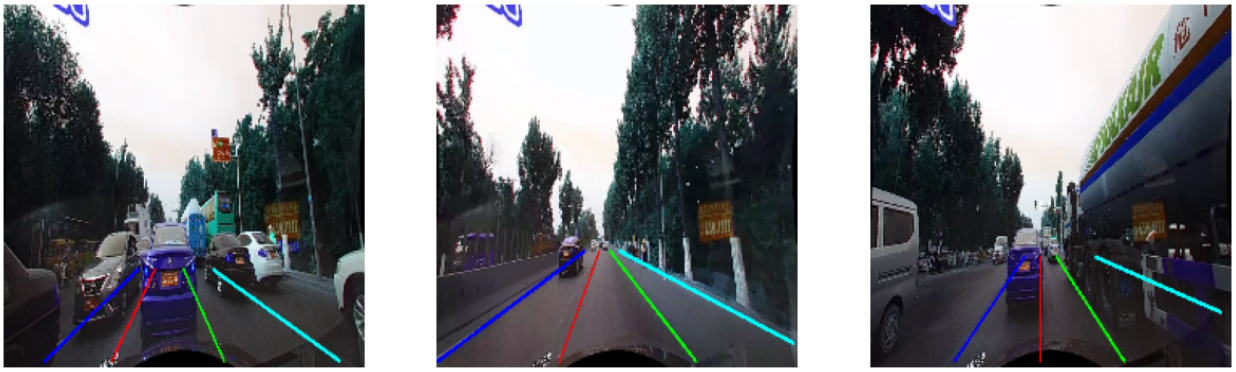


Fig.25 Images with lane mask

7. Train test split

After data preprocessing steps, we will shuffle the dataset, then split into training and validation sets using a method that allocates a specified proportion (e.g., 10%) of the data to the validation set. This split ensures the data is adequately distributed for both training and validation phases.

8. Model Building

Method 1 - Convolutional Neural Network (CNN)

It is a type of artificial neural network, which is widely used for image/object recognition and classification. CNN has ability to learn from raw pixel data, it's efficiency in capturing spatial hierarchies, and their robustness to variations in the visual appearance of lanes make cnn better suited for lane detection compared to traditional machine learning models that often require manual feature engineering and may not capture the spatial dependencies as effectively. These characteristics enable CNNs to perform well even in complex driving environments.

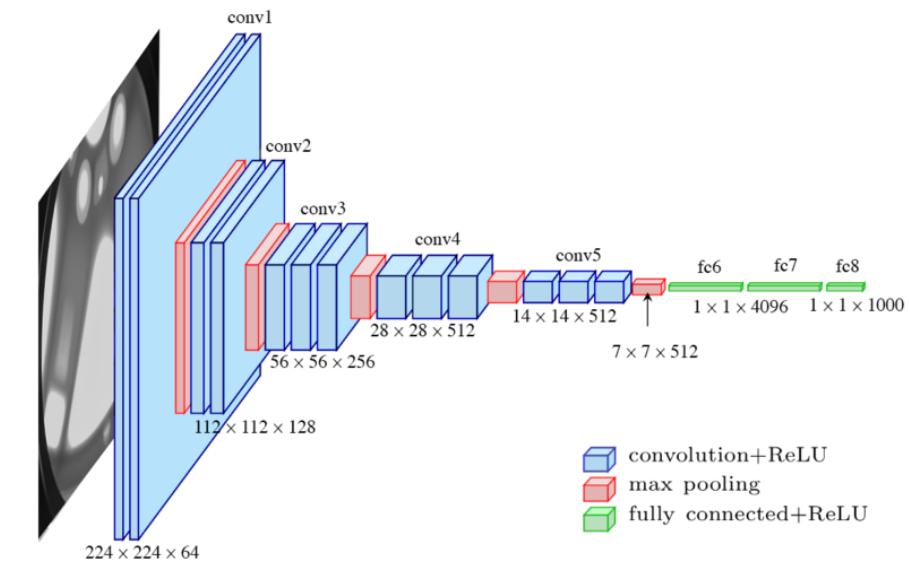


Fig. 26 *General CNN architecture*

Model Architecture

Layered Convolutional Architecture: The alternating pattern of convolutional layers with activation functions like ReLU (Rectified Linear Unit) allows the model to learn complex patterns in the images such as edges, shapes, and eventually, lane lines.

Pooling Layers: MaxPooling layers reduce the spatial dimensions of the output from the previous layer, which helps in reducing the computational load, and makes the detection features somewhat invariant to scale and orientation changes.

Dropout Layers: These layers help in preventing overfitting by randomly setting a fraction of input units to 0 at each update during training, which helps in making the model more robust to noise and variations in lane markings.

Batch Normalization: This normalizes the input layer by adjusting and scaling the activations, which helps in speeding up the training process and achieving higher performance.

Conv2DTranspose Layers and Upsampling: These layers are used for up sampling the feature maps and learning to reconstruct the segmentation map from the condensed feature map. This is important in segmentation tasks like lane detection, where we not only want to classify each pixel but also maintain the spatial hierarchy and resolution.

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 batch_normalization_2 (Bat  (None, 224, 224, 3)       12
 chNormalization)

 Conv1 (Conv2D)              (None, 222, 222, 8)       224

 Conv2 (Conv2D)              (None, 220, 220, 16)      1168

 max_pooling2d_2 (MaxPoolin  (None, 110, 110, 16)      0
 g2D)

 Conv3 (Conv2D)              (None, 108, 108, 16)      2320

 dropout_4 (Dropout)         (None, 108, 108, 16)      0

 Conv4 (Conv2D)              (None, 106, 106, 32)      4640

 dropout_5 (Dropout)         (None, 106, 106, 32)      0

 Conv5 (Conv2D)              (None, 104, 104, 32)      9248

 dropout_6 (Dropout)         (None, 104, 104, 32)      0
Conv6 (Conv2D)               (None, 50, 50, 64)        18496

 dropout_7 (Dropout)         (None, 50, 50, 64)        0

 Conv7 (Conv2D)              (None, 48, 48, 64)        36928

 dropout_8 (Dropout)         (None, 48, 48, 64)        0

 max_pooling2d_4 (MaxPoolin  (None, 24, 24, 64)        0
 g2D)

 up_sampling2d (UpSampling2  (None, 48, 48, 64)        0
 D)

 Deconv1 (Conv2DTranspose)   (None, 50, 50, 64)        36928

 dropout_9 (Dropout)         (None, 50, 50, 64)        0

 Deconv2 (Conv2DTranspose)   (None, 52, 52, 64)        36928

 dropout_10 (Dropout)        (None, 52, 52, 64)        0
  dropout_11 (Dropout)       (None, 106, 106, 32)      0

  Deconv4 (Conv2DTranspose)  (None, 108, 108, 32)      9248

  dropout_12 (Dropout)       (None, 108, 108, 32)      0

  Deconv5 (Conv2DTranspose)  (None, 110, 110, 16)      4624

  dropout_13 (Dropout)       (None, 110, 110, 16)      0

  up_sampling2d_2 (UpSamplin  (None, 220, 220, 16)      0
  g2D)

  Deconv6 (Conv2DTranspose)  (None, 222, 222, 16)      2320

  Final (Conv2DTranspose)    (None, 224, 224, 1)       145

=================================================================
Total params: 181693 (709.74 KB)
Trainable params: 181687 (709.71 KB)
Non-trainable params: 6 (24.00 Byte)
```

Fig. 27 CNN Architecture

Final Conv2DTranspose Layer: This layer is the final step in transforming the feature maps back into the original image size with the predicted lane markings.

Method2 – Mask R-CNN

Mask R-CNN is a model that combines object detection and instance segmentation. It is an extension of the Faster R-CNN architecture.

The key innovation of Mask R-CNN lies in its ability to perform pixel-wise instance segmentation alongside object detection. This is achieved through the addition of an extra "mask head" branch, which generates precise segmentation masks for each detected object. This enables fine-grained pixel-level boundaries for accurate and detailed instance segmentation.

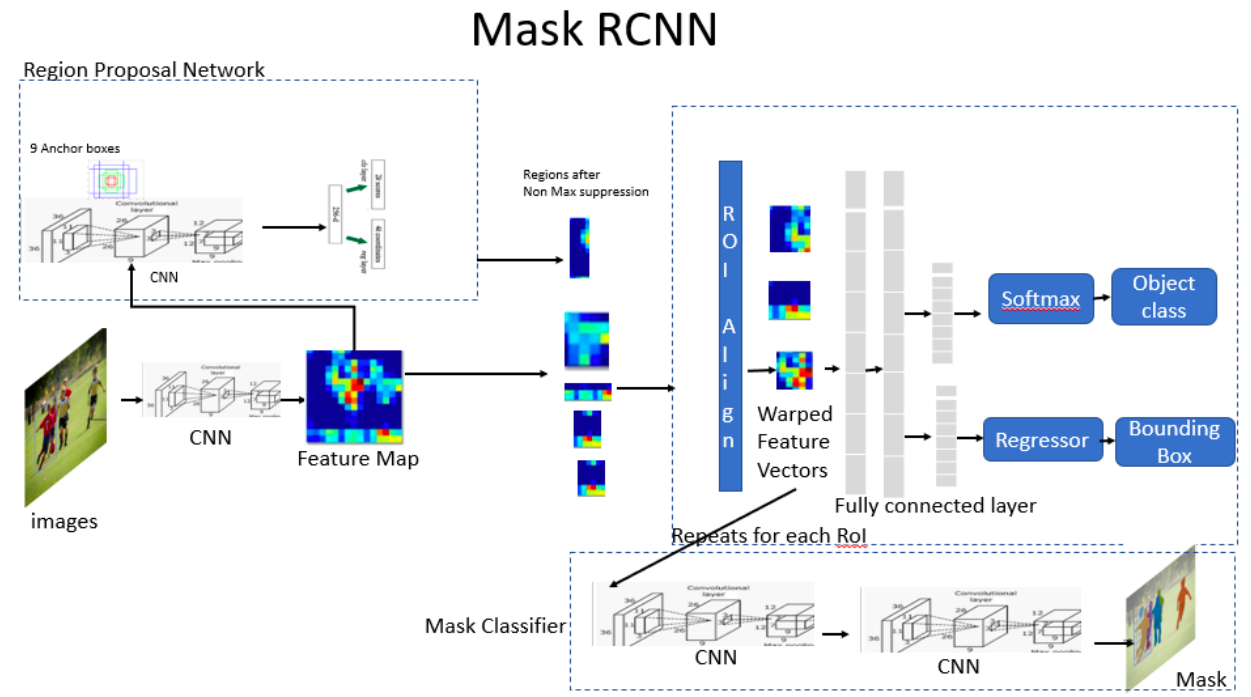


Fig.28: Mask RCNN General Architecture

Region Proposal Network (RPN):

- This part of the network generates region proposals. These are areas in the image that might contain an object. It does this by using anchor boxes at different scales and aspect ratios to cover various object sizes and shapes.
- The CNN (Convolutional Neural Network) processes the image and outputs a feature map, which is a high-level abstracted representation of the image.
- From this feature map, the RPN predicts object boundaries and objectness scores for each anchor box, indicating how likely it is to contain an object.
- Non-Max Suppression is then applied to filter out overlapping boxes, keeping only the best ones.

```
fpn_c5p5                (Conv2D)
fpn_c4p4                (Conv2D)
fpn_c3p3                (Conv2D)
fpn_c2p2                (Conv2D)
fpn_p5                  (Conv2D)
fpn_p2                  (Conv2D)
fpn_p3                  (Conv2D)
fpn_p4                  (Conv2D)
In model:  rpn_model
    rpn_conv_shared         (Conv2D)
    rpn_class_raw           (Conv2D)
    rpn_bbox_pred           (Conv2D)
mrcnn_mask_conv1        (TimeDistributed)
mrcnn_mask_bn1          (TimeDistributed)
mrcnn_mask_conv2        (TimeDistributed)
mrcnn_mask_bn2          (TimeDistributed)
mrcnn_class_conv1       (TimeDistributed)
mrcnn_class_bn1         (TimeDistributed)
mrcnn_mask_conv3        (TimeDistributed)
mrcnn_mask_bn3          (TimeDistributed)
mrcnn_class_conv2       (TimeDistributed)
mrcnn_class_bn2         (TimeDistributed)
mrcnn_mask_conv4        (TimeDistributed)
mrcnn_mask_bn4          (TimeDistributed)
mrcnn_bbox_fc           (TimeDistributed)
mrcnn_mask_deconv       (TimeDistributed)
mrcnn_class_logits      (TimeDistributed)
mrcnn_mask              (TimeDistributed)
```

Fig. 29 Mask RCNN

Feature Map:

Parallel to the RPN, the original image is processed through another CNN to generate a feature map. This is used to identify features within the image that are important for identifying objects.

ROI (Region of Interest) Align:

- The feature map and the proposals from the RPN are combined through an operation called ROI Align. This extracts a small feature map for each proposed region.
- The alignment process ensures that the extracted features correspond precisely to the parts of the image within the proposed regions, despite the size and aspect ratio differences.

Classifier and Bounding Box Regressor:

- Each small feature map is then passed through a fully connected layer to flatten it into a vector.
- This vector is input into two branches:
  1. A SoftMax classifier that predicts the class of the object within the proposal.
  2. A bounding box regressor that refines the coordinates of the proposed region to more accurately fit the object.

Mask Classifier:

- In addition to the class and bounding box, Mask R-CNN also outputs a binary mask for each RoI, which is used to segment the object by predicting a mask of pixels or a silhouette.
- This is done by a small CNN applied to each ROI, predicting the mask independently of the class and bounding box.

Method3 – SCNN Model

SCNN stands for spatial convolutional neural network. It specialized in lane detection in images. It customizes a VGG16 backbone with dilated convolutions for better feature extraction and employs message passing techniques for capturing spatial dependencies. The network outputs both lane segment predictions and lane existence probabilities. The loss function is a combination of cross-entropy for segmentation and binary cross-entropy for existence prediction, with different scaling factors to balance their importance.

The SCNN is suitable for lane detection due to its ability to effectively capture both local features (through the VGG16 backbone) and spatial relationships (through message passing), crucial for accurately identifying lane boundaries and their patterns. The use of dilated convolutions allows the network to have a larger receptive field, enhancing its ability to detect lanes over a wider area of the input image.

Interpreting the results, the network would provide both the segmented lane markings and the probability of existence for each lane. The accuracy of these predictions and the overall loss value (combining segmentation and existence accuracy) would indicate the network's effectiveness in identifying and classifying lane markings in various driving scenarios.

```
SCNN(
  (backbone): Sequential(
    (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ReLU(inplace=True)
    (3): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (4): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (5): ReLU(inplace=True)
    (6): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (7): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (8): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (9): ReLU(inplace=True)
    (10): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (11): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (12): ReLU(inplace=True)
    (13): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (14): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (15): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (16): ReLU(inplace=True)
    (17): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (18): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (19): ReLU(inplace=True)
    (20): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (21): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (22): ReLU(inplace=True)
    (23): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (24): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (25): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (26): ReLU(inplace=True)
    (27): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (28): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (29): ReLU(inplace=True)
    (30): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (31): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (32): ReLU(inplace=True)
    (34): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2))
    (35): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (36): ReLU(inplace=True)
    (37): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2))
    (38): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (39): ReLU(inplace=True)
    (40): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(2, 2), dilation=(2, 2))
    (41): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (42): ReLU(inplace=True)
  )
(layer1): Sequential(
  (0): Conv2d(512, 1024, kernel_size=(3, 3), stride=(1, 1), padding=(4, 4), dilation=(4, 4), bias=False)
  (1): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (2): ReLU()
  (3): Conv2d(1024, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (4): ReLU()
)
(message_passing): ModuleList(
  (0-1): 2 x Conv2d(128, 128, kernel_size=(1, 9), stride=(1, 1), padding=(0, 4), bias=False)
  (2-3): 2 x Conv2d(128, 128, kernel_size=(9, 1), stride=(1, 1), padding=(4, 0), bias=False)
)
(layer2): Sequential(
  (0): Dropout2d(p=0.1, inplace=False)
  (1): Conv2d(128, 5, kernel_size=(1, 1), stride=(1, 1))
)
(layer3): Sequential(
  (0): Softmax(dim=1)
  (1): AvgPool2d(kernel_size=2, stride=2, padding=0)
)
(fc): Sequential(
  (0): Linear(in_features=4500, out_features=128, bias=True)
  (1): ReLU()
  (2): Linear(in_features=128, out_features=4, bias=True)
  (3): Sigmoid()
)
(ce_loss): CrossEntropyLoss()
(bce_loss): BCELoss()
```

Fig. 30 SCNN Model Architecture

Backbone: A modified VGG16 network, with some layers replaced by dilated convolutions to increase the receptive field without losing resolution.

Layer 1: A sequence of convolutional, batch normalization, and ReLU layers, followed by a reduction in the number of channels.

Message Passing: Four convolutional layers for message passing in different directions (up-down, down-up, left-right, right-left). This is crucial for capturing spatial relationships in the image, particularly for continuous lane lines.

Layer 2: Consists of a dropout layer for regularization and a convolutional layer to output five channels, corresponding to different lane line types or background.

Layer 3: Applies SoftMax for classification and average pooling to reduce dimensions.

Fully Connected (FC) Layer: Processes the feature map into a final output, predicting the existence of lanes.

Loss Functions: Cross-entropy loss for lane line segmentation and binary cross-entropy loss for lane existence prediction.

9. Model Deployment

SCNN Model Deployment on Streamlit

Our lane detection project utilizes the Spatial Convolutional Neural Network (SCNN) model, which is specifically designed to detect lane markings in images. The model has been deployed using Streamlit, an open-source app framework, creating a user-friendly interface for real-time lane detection.

The deployment on Streamlit includes:

1. **Model Loading:** The pre-trained SCNN model is loaded with custom weights.

2. **Image Processing:** Uploaded images are resized and normalized to match the input requirements of the model.
3. **Prediction and Visualization:** The model predicts lane markings and overlays them on the original image for visualization.
4. **Video Processing:** The app also supports processing video input by extracting frames, detecting lanes in each frame, and recompiling them into a processed video.

Streamlit Interface

The Streamlit interface provides a user-friendly platform for users to upload images or videos for lane detection. The application dynamically displays the processed output, allowing users to visualize the effectiveness of the SCNN model in real-time lane detection.

This deployment not only showcases the model's capabilities but also provides an interactive tool for further testing and demonstration purposes.

Fig. 31 Streamlit Interface

Technology Stack

1. Deep Learning Framework:

- **TensorFlow:** An open-source software library for high-performance numerical computation, extensively used in deep learning for building and training neural networks. Its flexible architecture allows easy deployment of computation across various platforms (CPUs, GPUs, TPUs).
- **Keras:** A high-level neural networks API, running on top of TensorFlow. Keras simplifies the process of constructing, training, and deploying deep learning models. It provides an easier mechanism to express neural networks.

2. Programming Language:

- **Python:** The primary language for this project, renowned for its simplicity and rich ecosystem of libraries for data science and machine learning.

3. Image Processing and Computer Vision Libraries:

- **OpenCV (Open-Source Computer Vision Library):** A library of programming functions mainly aimed at real-time computer vision, used here for image processing tasks such as reading, resizing, and transforming images.
- **PIL (Python Imaging Library):** Employed for basic image handling operations like opening, manipulating, and saving many different image file formats.

4. Data Handling and Numerical Libraries:

- **NumPy:** Essential for handling arrays and matrices, especially useful for image data manipulation and operations.
- **Pandas:** Used for handling and analyzing structured data.

5. Integrated Development Environment (IDE) and Tools:

- **Google Colab:** A cloud-based Python notebook environment that provides free access to computing resources including GPUs, making it suitable for training deep learning models.

6. Model Development and Training Tools:

- **Matplotlib & Seaborn:** For plotting and visualizing data, which is crucial in understanding the performance of the model and the characteristics of the data.
- **Torchvision:** A PyTorch package consisting of popular datasets, model architectures, and common image transformations for computer vision

Experimental Results

1. Mask R-CNN

Following are the Mask RCNN Results

Input Image **Output Image**

Input Image **Output Image**

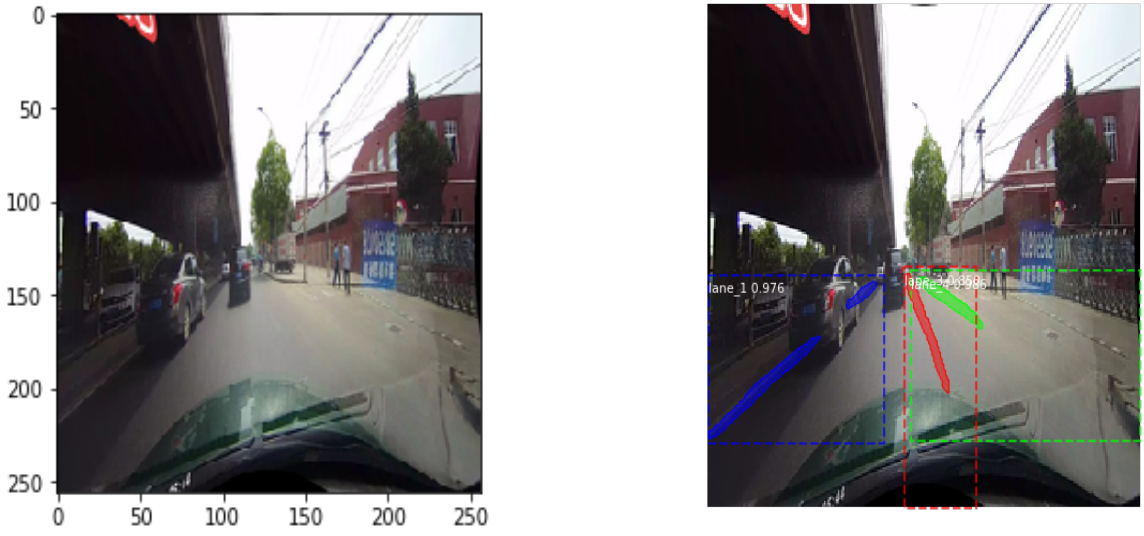
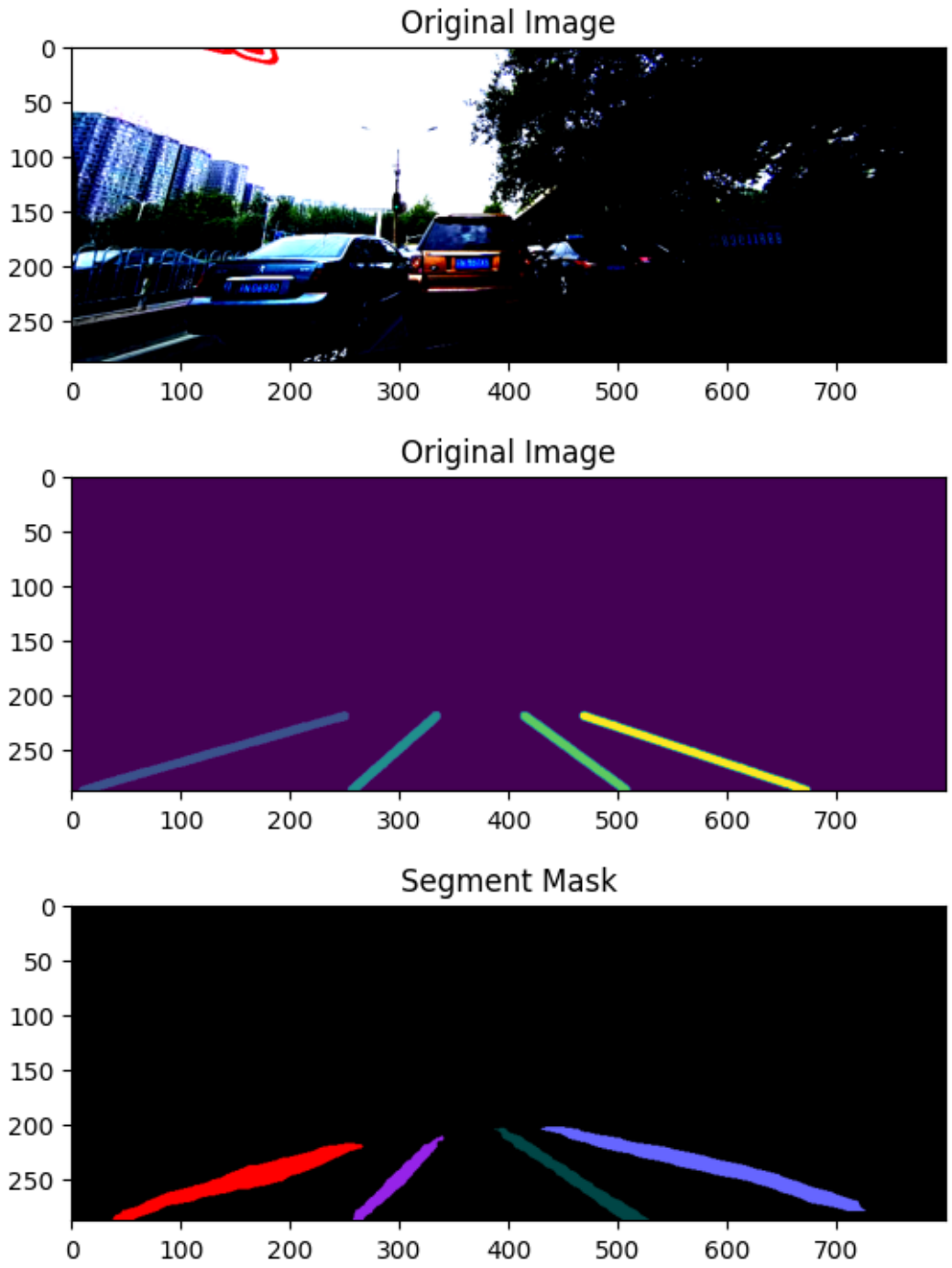


Fig.32: Mask RCNN Results

2. SCNN

Following are the SCNN Results



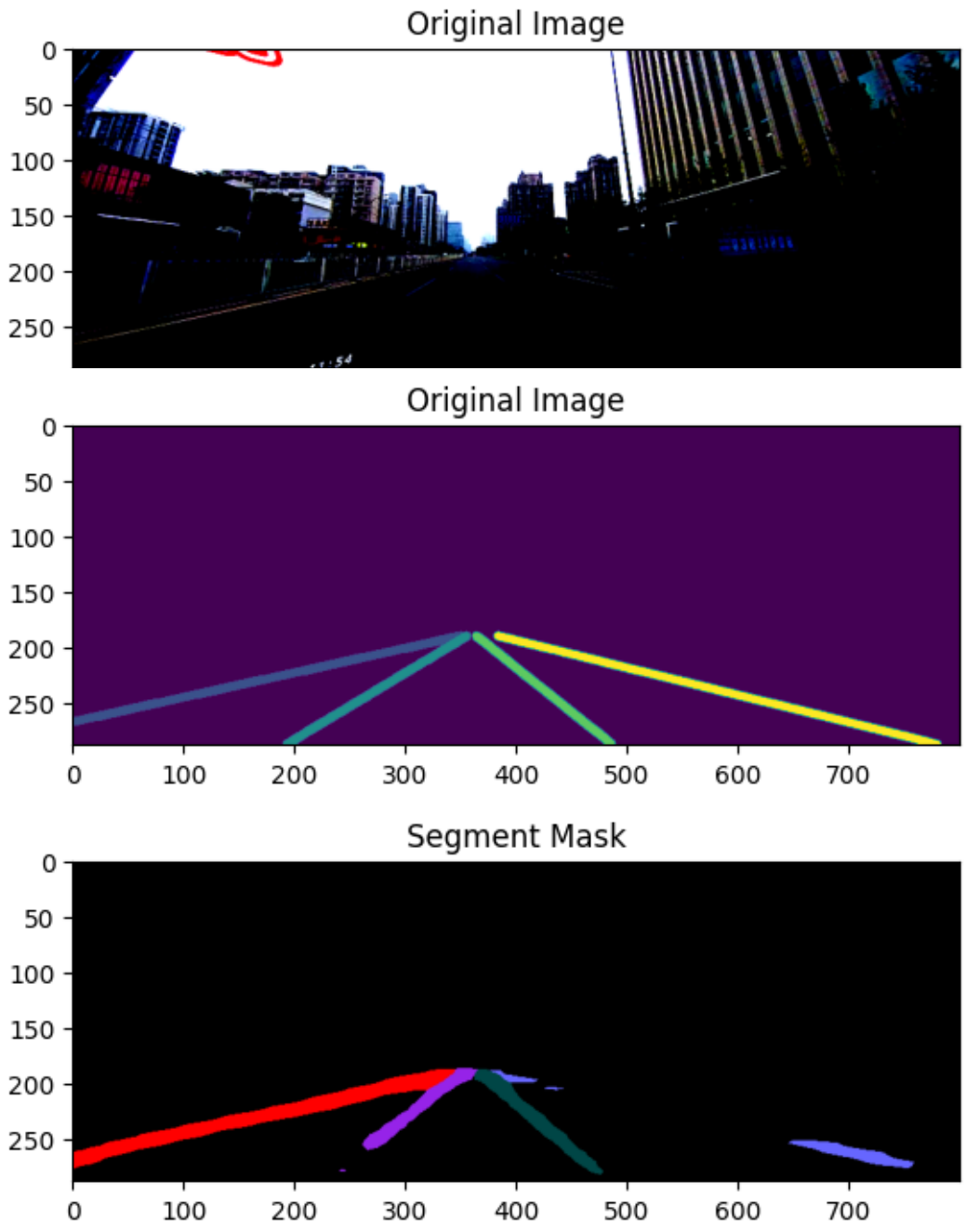


Fig.33 SCNN Results

3. Model Comparison

1. Models IoU Scores Histogram

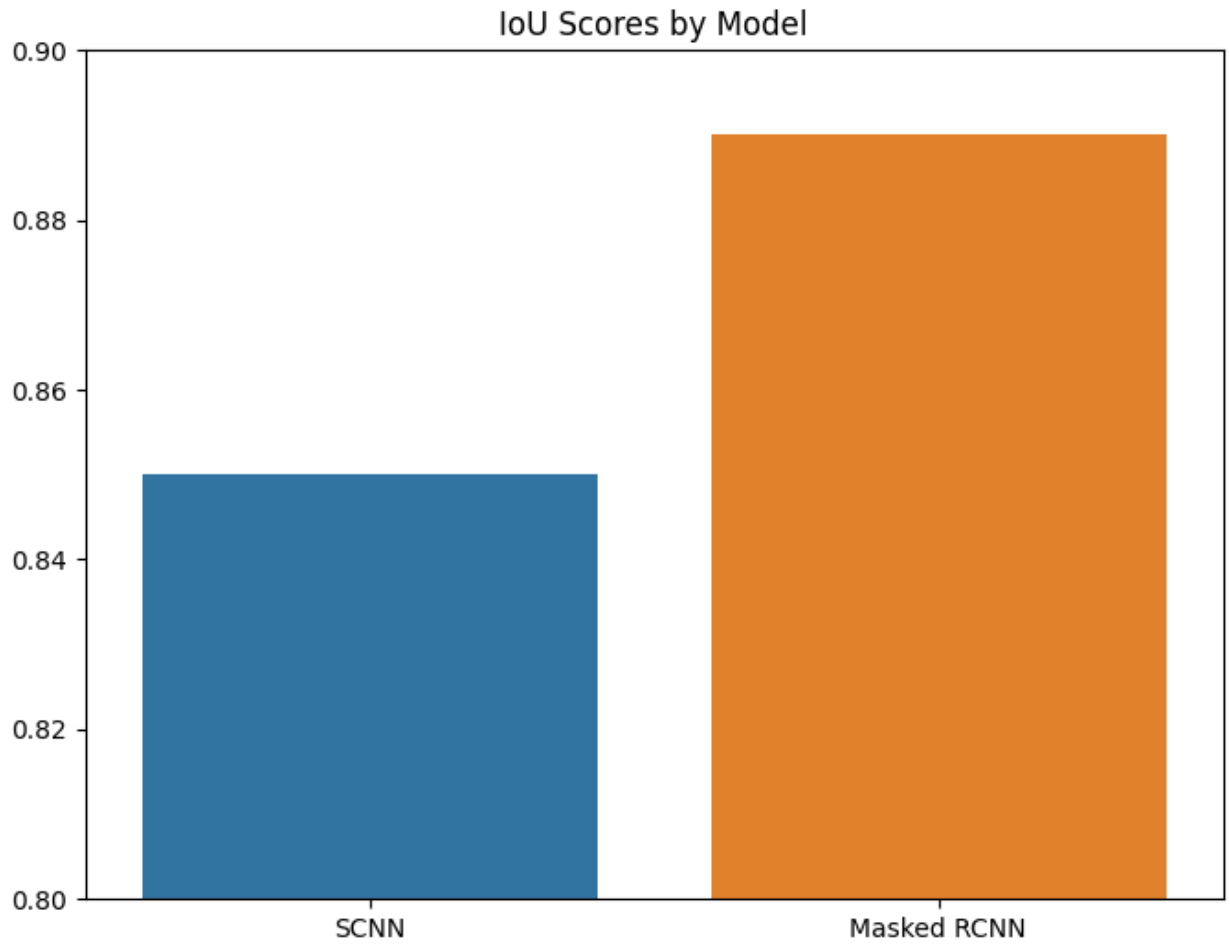


Fig.34: Model RMS Accuracy

IoU Scores: This bar graph shows the Intersection over Union (IoU) scores, which is a common evaluation metric for segmentation tasks. The scores are increasing with each iteration, indicating an improvement in the model's accuracy in segmenting lanes from the images.

2. Model Precision Scores

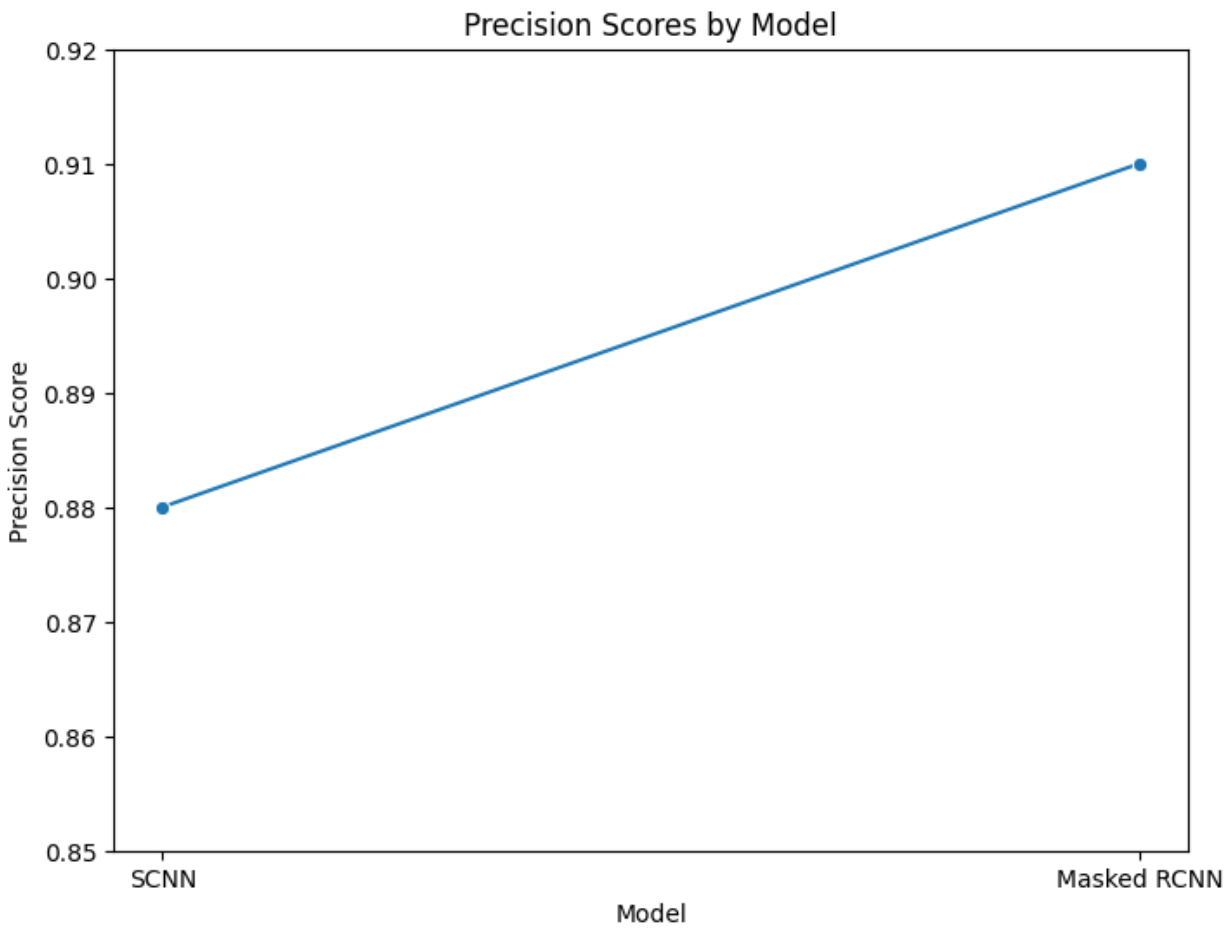


Fig.35: Precision Score by Model

Precision Scores: The line graph shows the precision scores, which measure the accuracy of the positive predictions. The increasing trend means that the model is getting better at predicting lanes without marking too many false positives.

3. Model Recall Scores

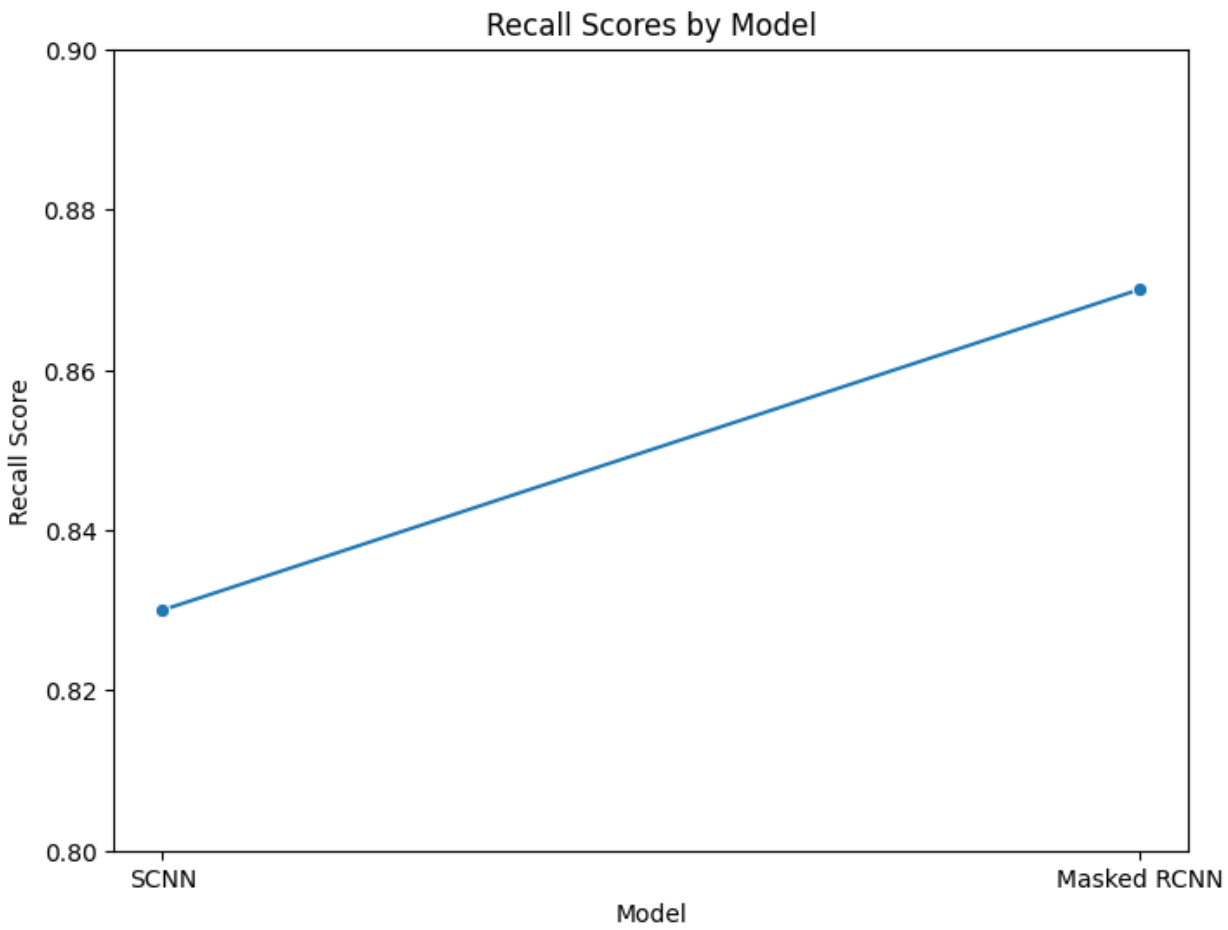


Fig.36: Recall Scores by Model

Recall Scores: This line graph shows the recall scores, which measure the model's ability to find all the relevant cases within the images. The upward trend suggests that the model is becoming more effective at detecting all actual lanes.

Conclusion

The culmination of this project in developing a state-of-the-art Lane Detection System using deep learning models marks a significant stride in autonomous vehicle technology. Central to this achievement were the Convolutional Neural Networks (CNNs), Spatial Convolutional Neural Networks (SCNNs), and the innovative Mask R-CNN model, which played pivotal roles in enhancing the accuracy and reliability of lane detection.

Our journey commenced with the exploration of various deep learning models, where CNNs stood out due to their efficiency in capturing spatial hierarchies and robustness to variations in visual appearance. The SCNN model, customizing a VGG16 backbone with dilated convolutions, was particularly effective in capturing both local features and spatial relationships crucial for accurately identifying lane boundaries. Meanwhile, the Mask R-CNN model showcased its prowess in performing pixel-wise instance segmentation alongside object detection, thereby significantly enhancing the lane detection process.

The project faced several challenges, including balancing the sensitivity and specificity of the system, ensuring real-time processing capabilities, and managing computational demands. Overcoming these hurdles required a meticulous approach in model selection, data curation, and a robust training methodology. The utilization of the CULane Dataset, a large-scale and challenging collection, was instrumental in achieving this, providing a diverse range of driving scenarios for rigorous model training and evaluation.

The experimental results validated the effectiveness of our chosen models. The Mask R-CNN model demonstrated superior performance in complex scenarios, such as inadequate lighting and occluded lane lines. The SCNN model also proved its mettle, especially in scenarios involving curved lanes and variable environmental conditions. These models not only improved the accuracy of lane detection but also ensured faster processing times, crucial for real-time applications.

In conclusion, the success of this project lies not just in the development of a highly accurate and reliable Lane Detection System but also in the significant contributions it makes to the field of autonomous vehicle technology. The integration of deep learning models like CNNs, SCNNs, and Mask R-CNN into lane detection has set a new benchmark in this domain. The project opens avenues for future research, particularly in enhancing algorithm efficiency, reducing computational costs, and improving real-time processing capabilities.

References

1. Arshad, N. M. (2011). Lane detection with moving vehicles using color information. . *In World Congress on Engineering and Computer Science.*
2. Dorj, B. &. (2016). A precise lane detection algorithm based on top view image transformation and least-square approaches. *Journal of Sensors*.
3. H. Bilal, B. Y. (2019). Real-Time Lane Detection and Tracking for Advanced Driver Assistance Systems. *Chinese Control Conference*, 6772-6777.
4. Kaur, G. &. (2015). Lane detection techniques: A review. *International Journal of Computer Applications*.
5. Kumar, A. M. (2015). Review of lane detection and tracking algorithms in advanced driver assistance system. *Int. J. Comput. Sci. Inf. Technol*, 65-78.
6. Kuo, C. Y. (2019). Lane Keeping Assist with Lane Detection. *Sensors*.
7. Maček, K. W. (2004). A lane detection vision module for driver assistance. *Sascha Eysoldt Verlag*, 913-918.
8. Muthalagu, R. B. (2020). lane detection technique based on perspective transformation and histogram analysis for self-driving cars. *Computers & Electrical Engineering*.
9. N. J. Zakaria, M. I. (2023). Lane Detection in Autonomous Vehicles: A Systematic Review. *IEEE Access*, 3729-3765.
10. Nguyen, V. K. (2018). A study on real-time detection method of lane and vehicle for lane change assistant system using vision system on highway. *Engineering science and technology*, 822-833.
11. Nguyen, V. Q. (2017). A study on detection method of vehicle based on lane detection for a driver assistance system using a camera on highway. *Asian Control Conference*, 424-429.
12. Priyadharshini, P. N. (2019). Advances in vision based lane detection algorithm based on reliable lane markings. *5th International Conference on Advanced Computing & Communication Systems (ICACCS)*, 880-885.
13. Pydipogu, P. F. (n.d.). *Robust lane detection and object tracking in relation to the intelligence transport system.*
14. Smadi, A. (2014). *Real-time lane detection for driver assistance system.* Circuits and Systems.
15. Song, W. Y. (2018). Lane detection and classification for forward collision warning system based on stereo vision. *IEEE Sensors Journal*, 5151-5163.
16. Waykole, S. S. (2021). Review on lane detection and tracking algorithms of advanced driver assistance system. *Sustainability*.