Research on a Deep Learning-Based Lane Line Recognition System

Yirui Liu, Aimei Xiao
School of Computer Science and Technology, Shandong University of Technology, Zibo, Shandong, China

Abstract: The identification of lane lines and the detection of road objects are important parts of environmental perception for intelligent driving. The current methods for lane detection are usually influenced by complicated backgrounds, changes in lighting and shadows, as well as the quality of the video, and a single detection function is not enough to fully understand the road environment. To solve these problems, a deep learning-based lane line recognition system has been developed and used in this paper. The system is implemented using Python, OpenCV, PyQt5, PyTorch and YOLOv5, including video pre-processing, lane line detection, road object recognition, video clipping and saving, uploading the video to Android side and measuring distance with binocular LiDAR cameras. The lane detection module identifies the left and right lane lines by using grayscale transformation, Canny edge detection, region of interest masking, Hough line detection and slope filtering. The road object recognition module employs YOLOv5 for vehicle and object detection and performs non-maximum suppression to improve the detection results of the bounding boxes. Experimental results indicate that the object-box selection accuracy is over 95% in three test videos, and the lane line recognition accuracy is also good. The system can stably detect and visualize road-scene information from video inputs, providing a practical reference for lightweight assisted-driving perception systems.

Keywords: Deep Learning; Lane Line Recognition; Yolov5; Opencv; Intelligent Driving; Object Detection

1. Introduction

With the rapid development of intelligent driving and advanced driver assistance systems, road-environment perception has become an important basis for vehicle safety control, path planning, and driving decision-making. Lane lines provide structured road information for lateral vehicle positioning, lane keeping, and lane-departure warning. Road object detection further helps a perception system identify vehicles, pedestrians, and other obstacles, thereby improving the system's understanding of surrounding traffic conditions. Since lane departure is a major cause of traffic accidents, a robust lane line recognition system has clear practical value. Traditional real-time lane detection research has demonstrated that image-based lane extraction can support autonomous vehicle navigation when the road boundary is clear and the computational pipeline is carefully designed [1].

Object detection and lane recognition have also benefited from deep learning. YOLO-based detectors have become widely used in real-time visual perception because they balance speed and accuracy in a single-stage detection framework [2]. Recent work further shows that monocular lane detection based on deep learning has become a major research direction for autonomous driving perception [3]. YOLOv5-based lane segmentation has been explored for real-time autonomous vehicle applications [4], and attention-enhanced models have been proposed to improve lane detection on complex roads [5]. Optimization frameworks that combine lane detection and steering control have also appeared in recent autonomous driving research [6]. In 2025, improved YOLOv5 lane segmentation systems with additional segmentation heads were reported, showing the continuing relevance of YOLOv5-style architectures for lightweight lane perception [7].

At the same time, semantic segmentation networks such as DeepLab and U-Net provide useful ideas for pixel-level road understanding [8,9]. Robust lane detection studies and surveys have discussed continuous driving scenes, lane marking detection, and deep learning-based lane

Copyright @ STEMM Institute Press

detection methods from different viewpoints [10-12]. Vehicle detection studies also support the integration of road object recognition into lane perception systems [13]. End-to-end self-driving research and recent challenging-condition lane classification work indicate that lane perception is gradually moving from isolated line extraction to integrated scene understanding [14,15]. Domestic engineering studies have provided implementation references for lane detection systems and highway road-condition detection [16-18].

Based on the original system design and experimental materials, this paper reorganizes and refines the lane line recognition system into a publishable English paper. The proposed system combines a traditional lane detection algorithm with YOLOv5-based object recognition to form a complete workflow from video acquisition, preprocessing, detection, visualization, and video saving. It also integrates a PyQt5 desktop interface, Android-side video uploading, and Gemini 2 binocular LiDAR camera-based distance measurement, which improves interaction and extends application scenarios. Compared with a purely algorithmic demonstration, the system emphasizes both detection performance and usable software workflow. The main contributions are as follows: an integrated lane detection and object recognition pipeline is built; video uploading, clipping, playback, and result saving functions are implemented; a distance-measurement extension is integrated; and the system performance is analyzed using training curves and test data.

2. System Design and Methods

2.1 Overall System Architecture

The system consists of training datasets, test datasets, video acquisition, video preprocessing, lane line detection, object recognition, video clipping and saving, user interaction, and an extended distance-measurement module. Video sources include user-uploaded video files and data captured by a Gemini 2 binocular LiDAR camera. The desktop interface is implemented with PyQt5, allowing users to select videos, play or pause videos, drag the progress bar, set clipping intervals, choose recognition modes, and save processed results. The Android module supports video selection and uploading from mobile devices. After receiving the uploaded video, the server invokes the recognition program and saves the processed output. The revised English functional architecture of the system is shown in Figure 1.

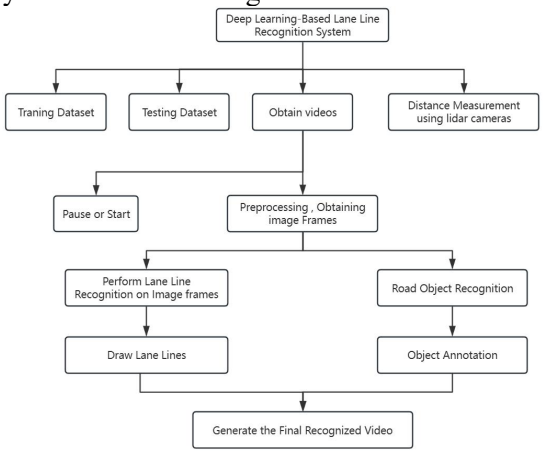

Figure 1. Functional Architecture of the System

This architecture divides the algorithmic processing from the user interaction. The front part deals with the video input, parameter setting and result presentation, whereas the back part carries out frame extraction, model inference, lane detection, object annotation and video recording. The modular structure allows the system to change among various video sources, recognition modes and installation methods. It also facilitates the replacement of the model and the future cooperative processing by multiple cameras. In actual application, this distinction prevents the user interface operations from affecting the frame processing, and simplifies the maintenance of the software when the model or the input device is updated.

2.2 Video Preprocessing and Lane Line Detection

Preprocessing of the video is the initial phase in lane line recognition. It involves the reading of video format, frame extraction, image resizing, conversion to grayscale and denoising. OpenCV is utilized to read the video stream and change the continuous video information into single frames. Each frame is afterwards treated by edge detection and region filtering. These pre-processing stages decrease the effect of the road background, image noise and unnecessary areas, enabling the following algorithm to concentrate on the road region. Moreover, the pre-processing stage enhances the similarity of the input frames, which is significant when the videos are from various sources, resolutions and encoding formats.

 Copyright @ STEMM Institute Press

The LaneDetection module is mainly implemented in the LaneDetection class which includes the LD, calculate_slope and app_canny functions. First, the LD function transforms the input image into grayscale and performs Canny edge detection, then establishes a region of interest mask to highlight the possible lane edges in the road area. The Hough transform is used to detect line segments and the slope of each detected line is calculated. According to the sign and value of the slope, the system distinguishes between the left and right lane lines, removes unreasonable line segments, fits the lane boundaries and superimposes these boundaries onto the original video frame. This method keeps a clear view and reduces the overall visual computation, thus it is suitable for real-time video processing.

The advantage of this design is that the lane detection procedure remains clear. All intermediate steps, such as edge extraction, region selection, line detection and slope filtering, can be checked and modified. It is advantageous for a system which operates on common hardware and processes user-submitted videos. However, the method is still affected by strong shadows, worn lane-lines, severe curves and complex roadside patterns. Therefore, this paper considers the traditional lane detection module as a simple foundation which can be used together with deep learning models in the subsequent versions.

2.3 Road Object Recognition Based on YOLOv5

The road object recognition module identifies objects in road scenes using YOLOv5. A pre-trained model is loaded by PyTorch and used for inference in the run_inference function. After preprocessing, each frame of the video is inputted into the YOLOv5 network, which gives probable bounding boxes, category labels and confidence values. Non-maximum suppression is carried out to eliminate the overlapping boxes and keep those with higher confidence. Subsequently, the bounding boxes and labels are drawn on the frame together with the lane detection results, so that the users can observe both the lane boundaries and the surrounding road objects. An identification example is depicted in Figure 2.

Training and testing are conducted using the PyTorch framework and GPU acceleration is utilized to enhance the training speed. During the training process, data concerning lane lines and road objects are loaded, hyperparameters are set up and various epochs are carried out. The model is assessed by means of box_loss, obj_loss and cls_loss which indicate the errors in bounding-box localization, objectness prediction and classification respectively. In the testing phase, an independent test set is employed to examine the generalization ability. The values of precision, recall, mAP and the change of loss are recorded and presented graphically.

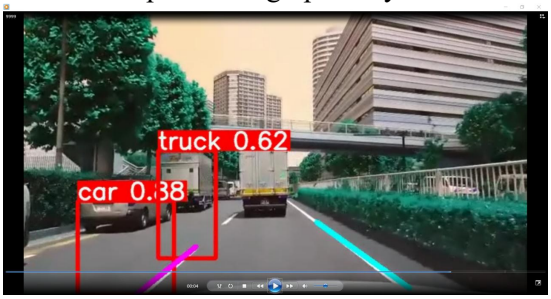

Figure 2. Lane Line and Road Object Recognition Result

The application of YOLOv5 is suitable for the light-weight feature of the system. In comparison with the heavy segmentation-only procedures, YOLOv5 can offer effective object recognition speed without complicating the implementation and deployment. In this system, object recognition is not aimed at replacing the lane detection; instead, it supplements the road-object information with the lane information. This arrangement is beneficial for assisted driving as the lane boundaries indicate the drivable area and the object boxes represent the obstacles in the traffic scene.

2.4 User Interface, Mobile Uploading, and Distance Measurement

The desktop application is developed with PyQt5 and includes a start page, a video selection page and a main processing page. After a video is submitted, the users can operate the playback, adjust the progress bar, set the clip limits and save the altered video. The video clipping is performed by the cut_video, update_range and format_time functions in the VideoAPP class. The cut_video function determines the start and end times chosen by the user and uses FFmpeg for precise clipping. Video saving is accomplished via the save_video, update_frame and process_frame functions. In our system, OpenCV VideoWriter is utilized to record the processed frames into the output file.

Copyright @ STEMM Institute Press

To improve the flexibility, the original system can now upload videos on the Android platform as well. One can select a local video from his/her mobile phone and transmit it to the server. The server will then process the video, carry out the recognition procedure and save the result in a designated folder. In addition, a Gemini 2 binocular LiDAR camera has been installed for distance measurement. Once an object is identified, the system provides the X distance, Y distance and depth information. This change enables the system to recognize road targets and lane lines and offer the relative distance for environmental perception. Figure 3 depicts the Android interface.

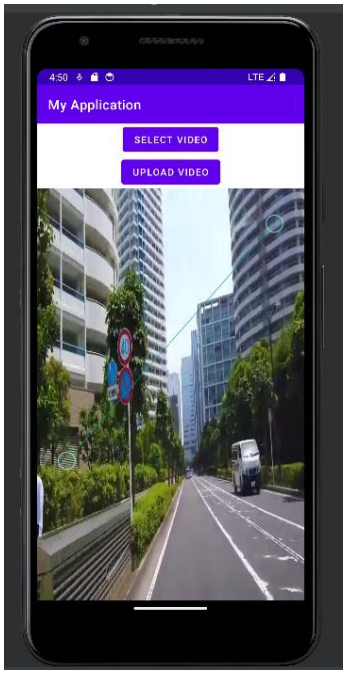

Figure 3. Android-Side Video Selection and Uploading Interface

3. Experimental Results and Analysis

3.1 Analysis of Model Training

During the model training process, the system keeps track of the objectness loss, classification loss and bounding-box loss. The training curves indicate that the initial loss values are relatively large, suggesting that the model is not yet stable in the prediction of target existence, category recognition and bounding-box localization. With the increase of the number of epochs, the three losses tend to decrease generally, implying that the model gradually acquires the visual features of road objects and continually enhances the target localization and classification. The training loss curves are depicted in Figure 4.

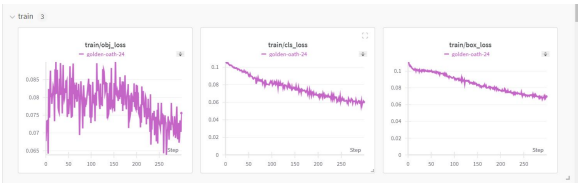

Figure 4. Training Loss Curves

The test loss curves exhibit a similar pattern. During the validation process, obj_loss, cls_loss and box_loss decrease with the increase of training, which suggests that the model enhances its object recognition, category discrimination and object location on separate test data. The overall loss and accuracy curves further show that the model learns quickly in the early training stage and then gradually becomes stable. The precision, recall, and mAP curves rise and tend to converge, suggesting that the trained model can perform road object detection with acceptable stability. The evaluation curves are shown in Figure 5.

Figure 5. Evaluation Metric Curves of the Model

The curves should be interpreted together rather than separately. A decline in box_loss indicates that the predicted location of a detected object becomes closer to the ground-truth location. A decline in obj_loss means that the model becomes more reliable in judging whether an object exists in a candidate region. A decline in cls_loss suggests that the learned visual representation becomes more discriminative for object categories. When these losses decrease while precision, recall, and mAP improve, the model training process can be considered effective for the current dataset and task setting.

3.2 Road Object Detection Results

To verify the road object recognition performance, the original system uses two self-made videos and one online video for testing. The evaluation includes object-box selection accuracy and object-label accuracy. Object-box accuracy describes whether the detected bounding box correctly covers the main target,

 Copyright @ STEMM Institute Press

while object-label accuracy describes whether the predicted category label is consistent with the actual object category. The results are shown in Table 1.

Table 1. Road Object Detection and Recognition Results

Input video stream	Object-box accuracy	Object-label accuracy
Self-made video 1	95%	88%
Self-made video 2	98%	86%
Online video 1	96%	85%

As shown in Table 1, the object-box selection accuracy is no lower than 95% for all three videos, indicating that the YOLOv5 model can accurately localize major objects in road scenes. The object-label accuracy ranges from 85% to 88%, showing that the model can complete basic category recognition, but there is still room for improvement under complex backgrounds, occlusion, and long-distance targets. Overall, the object detection module provides a useful foundation for road-scene perception. Future improvement can be achieved by expanding road-scene datasets, balancing category samples, and applying transfer learning.

3.3 Lane Line Recognition Results

Lane line recognition is evaluated using left-lane accuracy and right-lane accuracy. Under average grayscale processing, the original system compares different filtering and detection Configuration and obtains the results shown in Table 2.

Table 2. Lane Line Detection and Recognition Results

Input video stream	Left-lane accuracy	Right-lane accuracy
Self-made video 1	95%	88%
Self-made video 2	98%	86%
Online video 1	96%	85%

As shown in Table 2, the left-lane recognition accuracy remains above 95%, while the right-lane recognition accuracy ranges from 85% to 88%. The difference between the two sides may be related to camera perspective, illumination, lane-line clarity, and interference from the roadside background. The combination of Canny edge detection and Hough transform has good real-time performance and interpretability, but it may still suffer from missed detections and fitting errors in shadows, occlusion, worn lane markings, or curved roads. Future work may introduce deep segmentation networks or temporal information fusion to improve robustness in complex scenarios.

The experimental results also show that detection stability depends not only on the model but also on the quality of video acquisition. Self-made videos usually have more controllable shooting angles and clearer lane boundaries, while online videos may contain compression artifacts, inconsistent camera positions, and more complex backgrounds. Therefore, the system should be evaluated on a larger set of road videos before practical deployment. For a publishable experimental protocol, future work should report the number of frames, the annotation standard, the confidence threshold, the processing frame rate, and the hardware Configuration used during inference.

3.4 Functional Verification of the System

In terms of functional completeness, the system implements video loading, playback control, progress-bar dragging, video clipping, result saving, Android-side uploading, server-side processing, and binocular camera-based distance measurement. The desktop application can overlay detected lane lines and object boxes on video frames in real time and save the processed video through VideoWriter. The Android module can upload mobile videos to the server, and the server saves processed outputs after recognition. This workflow forms a complete closed loop from user input to algorithmic processing and result output.

From an application perspective, the system combines lane line recognition, road object detection, and object distance measurement, providing intuitive results for assisted-driving perception. Compared with a system that only detects lane lines, the proposed system can simultaneously show road boundaries and front vehicles. Compared with a system that only performs object detection, the lane line results provide additional road-structure constraints. Because the original experimental materials are limited, the current test sample size is still small, and the evaluation mainly uses accuracy statistics. Future evaluation should include night scenes, rainy and foggy weather, backlighting, curves, and worn lane markings, and should further report FPS, average processing time, precision, recall, and mAP.

4. Conclusion

This paper presents a revised English double-

column version of a deep learning-based lane line recognition system. The system is built with YOLOv5, OpenCV, PyTorch, and PyQt5, and implements video preprocessing, lane line detection, road object recognition, video clipping and saving, Android-side uploading, and binocular LiDAR camera-based distance measurement. The lane detection module uses Canny edge detection, region-of-interest masking, Hough transform, and slope filtering to fit left and right lane lines. The object recognition module uses YOLOv5 inference and non-maximum suppression to detect road objects. Experimental results show that the system can stably output lane line and road object recognition results on three test videos, with object-box selection accuracy above 95%.

The system still has limitations. The test sample size is limited, and generalization needs to be verified under more complex road conditions. The common method for detecting lanes is affected by lighting changes, occlusions and deteriorated lane marks. Moreover, the object recognition accuracy can be increased. In the future, we might integrate deep semantic segmentation networks, temporal feature fusion, collaboration between multiple cameras, transfer learning and lightweight network optimization to improve the stability and real-time ability in challenging driving conditions. The modified article also removes the Chinese text from the main text and the figures and adjusts the reference sequence according to the order of the first cited reference.

References

[1] Assidiq A A M, Khalifa O O, Islam M R, et al. Real time lane detection for autonomous vehicles//2008 International Conference on Computer and Communication Engineering. IEEE, 2008: 82-88.

[2] Redmon J, Farhadi A. YOLO9000: better, faster, stronger//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017: 7263-7271.

[3] He X, Guo H, Zhu K, et al. Monocular lane detection based on deep learning: a survey. arXiv:2411.16316, 2024.

[4] Swain S, Tripathy A K. Real-time lane detection for autonomous vehicles using YOLOV5 Segmentation Model. International Journal of Sustainable Engineering, 2024, 17(1): 718-728.

[5] Maddiralla V, Subramanian S. Effective lane detection on complex roads with convolutional attention mechanism in autonomous vehicles Scientific Reports, 2024, 14: 19193.

[6] Yordanov D, Chakraborty A, Hasan M M, Cirstea S. A framework for optimizing deep learning-based lane detection and steering for autonomous driving. Sensors, 2024, 24(24): 8099.

[7] Feilong Q, Khan N A, Jhanjhi N Z, Ashfaq F, Hendrawati T D. Improved YOLOv5 lane line real time segmentation system integrating Seg Head Network. Engineering Proceedings, 2025, 107(1): 49.

[8] Chen L C, Papandreou G, Kokkinos I, et al. Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2017, 40(4): 834-848.

[9] Ronneberger O, Fischer P, Brox T. U-Net: convolutional networks for biomedical image segmentation//Medical Image Computing and Computer-Assisted Intervention. Springer, 2015: 234-241.

[10] Zou Q, Jiang H, Dai Q, et al. Robust lane detection from continuous driving scenes using deep neural networks. IEEE Transactions on Vehicular Technology, 2019, 69(1): 41-54.

[11] Zhang Y, Lu Z, Zhang X, et al. Deep learning in lane marking detection: a survey. IEEE Transactions on Intelligent Transportation Systems, 2021, 23(7): 5976-5992.

[12] Tang J, Li S, Liu P. A review of lane detection methods based on deep learning. Pattern Recognition, 2021, 111: 107623.

[13] Wang Z, Zhan J, Duan C, et al. A review of vehicle detection techniques for intelligent vehicles. IEEE Transactions on Neural Networks and Learning Systems, 2022, 34(8): 3811-3831.

[14] Bojarski M, Del Testa D, Dworakowski D, et al. End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316, 2016.

[15] Rahman Z, Morris B T. LVLane: deep learning for lane detection and classification in challenging conditions//2023 IEEE 26th International Conference on Intelligent Transportation Systems. IEEE, 2023: 3901-3907.

[16] Li S Z. Design and implementation of a lane line detection system based on deep learning.

 Copyright @ STEMM Institute Press

Harbin Institute of Technology, 2016.
[17]Wang J Q. Design and implementation of a lane line detection system based on deep learning. Southwest University, 2022.
[18]Yang H. Design and implementation of a highway road-condition detection system. University of Electronic Science and Technology of China, 2023.

Copyright @ STEMM Institute Press