Image Processing for License Plate Recognition

Maanasvee Khetan, Sanya Malik, Reya Oberoi
Department of Artificial Intelligence
NMIMS Mukesh Patel School of Technology Management and Engineering, Mumbai, India

***Abstract*—License Plate Recognition (LPR) is a crucial technology in intelligent transport systems, enabling automated vehicle identification. This study explores preprocessing techniques such as noise removal, contrast enhancement, and edge detection to improve detection accuracy. The methodology involves using OpenCV and Tesseract OCR for plate extraction and recognition. Experimental results highlight areas for improvement, including deep learning-based segmentation and OCR models. Future enhancements focus on real-time deployment using optimized neural networks.**

I. Introduction

License Plate Recognition (LPR) is a critical component of modern intelligent transport systems, widely used in automated toll collection, traffic monitoring, and security enforcement. It involves detecting a license plate in an image, segmenting the characters, and recognizing them using Optical Character Recognition (OCR) techniques. Traditional LPR systems relied on basic image processing methods such as edge detection and thresholding, but these approaches struggle with challenges like varying lighting conditions, occlusions, distortions, and diverse plate formats. Recent advancements in machine learning and deep learning have significantly improved accuracy and robustness. This study explores various preprocessing techniques essential for license plate detection, evaluates an OpenCV-based implementation, and proposes improvements to enhance accuracy and real-time performance.

II. Methodology

This research proposes a robust and systematic framework for automatic license plate recognition by leveraging advanced image processing methodologies in conjunction with deep learning models. The proposed pipeline is delineated into seven distinct stages: image pre-processing, license plate localization, character segmentation, enhancement of segmented regions, character recognition, supervised machine learning-based classification, and final output prediction. Each stage is meticulously designed to optimize system performance and ensure high recognition accuracy under diverse real-world conditions, including variable illumination, oblique viewing angles, and the presence of image noise.

2.1 Preprocessing Steps Used

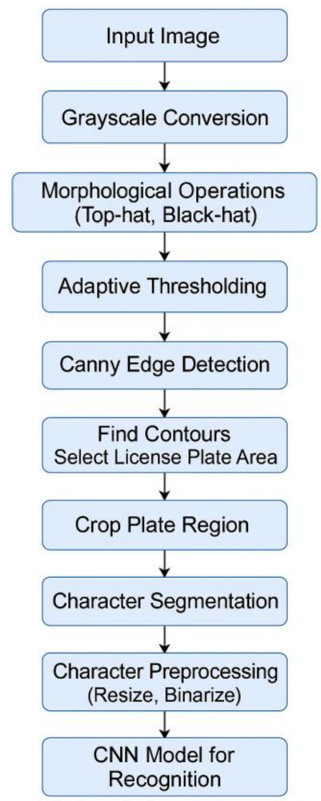

Fig1. Preprocessing flowchart

Image pre-processing is a crucial initial stage where raw images are refined to enhance relevant features and reduce computational complexity. First, the input image of the vehicle is loaded into the system. Since color information is not essential for text extraction, the image is converted to grayscale. This reduces the data from three color channels (RGB) to a single intensity channel, simplifying further processing.

Fig2. Original Image

Fig3. Converted Image to Grayscale

To improve the visual distinction between the license plate characters and their background, contrast enhancement is applied. Techniques such as histogram equalization or CLAHE (Contrast Limited Adaptive Histogram Equalization) are used to adjust local contrast, thereby making characters more discernible even under suboptimal lighting conditions.

Fig4. Maximize Contrast

2.2. ***Image Processing for Plate Localization***

This step focuses on identifying and isolating the license plate from the full vehicle image. Adaptive thresholding is applied to the enhanced grayscale image. Unlike global thresholding, adaptive methods calculate thresholds for small regions of the image, making them robust to lighting variations.

Fig5. Adaptive Thresholding

Edge detection algorithms like Canny Edge Detection are then employed to highlight object boundaries, followed by contour detection. The system identifies rectangular contours that could correspond to a license plate. These candidate regions are filtered based on their aspect ratio, size, and relative position within the image. Once the most probable region is located, it is cropped and passed on for further processing.

Fig6. Finding Contours to Locate Plate

Fig7. Data Preparation

2.3. ***Character Segmentation***

In this stage, the localized license plate image is analyzed to extract individual characters. Segmentation begins with detecting all contours within the plate region. However, not all contours represent valid characters—some may be noise or artifacts (e.g., bolts or frame). Hence, filtering is performed based on geometrical features such as height, width, area, and aspect ratio.

Following size-based filtering, the contours are sorted left to right to maintain the natural reading order of the characters. Each valid character is then enclosed in a bounding box. These bounding boxes are superimposed on the original image to visually confirm the segmentation quality before proceeding to recognition.

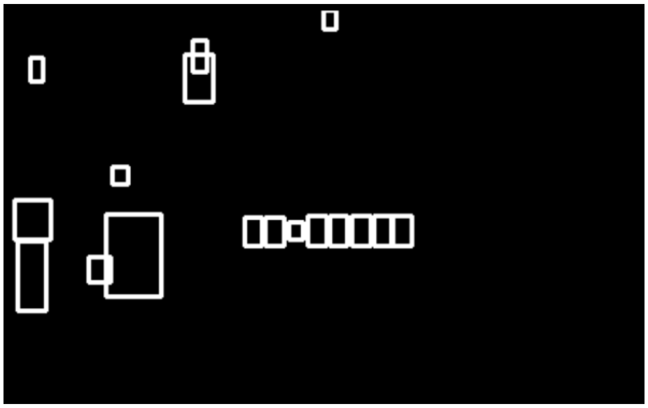

Fig8. Selecting Boxes by Char Size

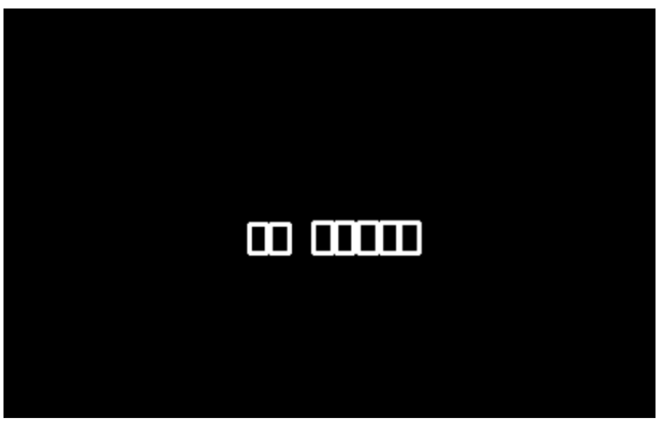

Fig9. Selecting Boxes by Arrangement of Contours

Fig10. Imposing Boxes on Original Image of Car

2.4. Post-processing and Enhancement

After segmentation, post-processing improves the clarity and orientation of each extracted character. If the plate appears tilted, geometric transformations are applied to rotate and align it horizontally. This alignment ensures consistent input for the recognition model.

Fig11. Rotate Plate Images

Another round of thresholding is used to further clean the segmented characters and eliminate any background noise. If the license plate has a light-colored font on a dark background (common in some regions), the image is inverted to ensure the characters are in dark pixels on a light background— a format preferred by most OCR systems.

Fig12. Thresholding Again to Find Characters

Fig13. Taking Negative Again

2.5. OCR and Character Recognition

Optical Character Recognition (OCR) techniques are used to convert the segmented character images into alphanumeric text. Initially, a second round of contour detection is performed on the cleaned license plate image to refine character boundaries.

Each identified character is resized to a uniform dimension and isolated as an individual image. This normalization step ensures that all characters have a consistent format, which improves recognition accuracy. These processed character images form the input to the trained recognition model.

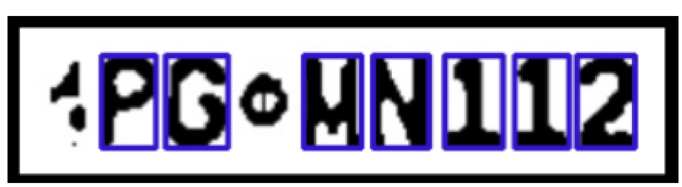

Fig14. A Function to Separate the Characters

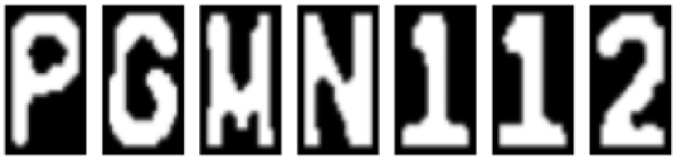

Fig15. Separating the Characters in the Plate

2.6. Machine Learning Model for Recognition

A supervised machine learning model, specifically a Convolutional Neural Network (CNN), is used for recognizing individual characters. The system is trained using a labeled dataset of license plate characters, which includes digits (0–9) and uppercase English alphabets (A–Z).
Before training, the dataset is preprocessed by resizing all character images and applying data augmentation techniques to simulate different real-world conditions. The dataset is then divided into training and validation sets to evaluate the model's performance.

The CNN architecture comprises convolutional layers for feature extraction, pooling layers for dimensionality reduction, and fully connected layers for classification. The final output layer contains neurons equal to the number of character classes (36 for 0–9 and A–Z). Once trained, the model achieves high accuracy in classifying previously unseen character images.
The trained model is stored for inference, allowing real-time predictions without re-training.

2.7. Prediction and Final Results

In the final phase, the system loads the pre-trained recognition model and applies it to segmented characters extracted from new vehicle images. Each character is passed through the model, and its predicted label is obtained.
The predictions are then concatenated in sequence to form the complete license plate number. The final result is displayed, optionally overlayed on the original image to provide visual confirmation. This output can then be logged, stored in a database, or used in applications like vehicle access control, traffic monitoring, or parking systems.

Fig16. Printing the output

III. Experimental Results

The license plate recognition system was evaluated using various images under different lighting conditions, backgrounds, and angles. The following metrics were used to assess performance:

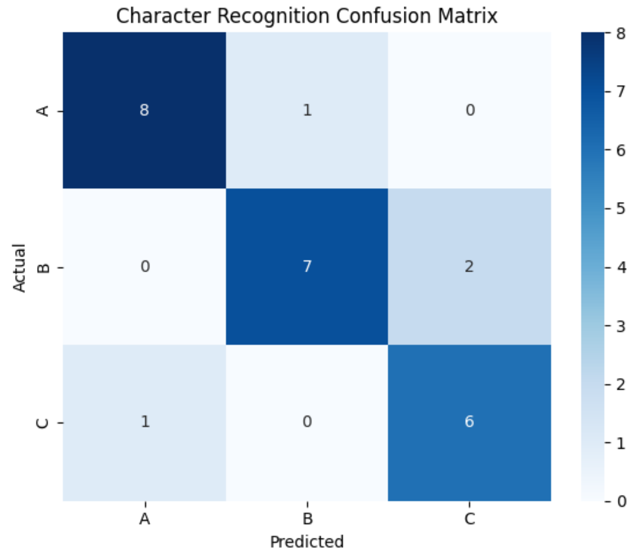

Fig17. Character Recognition Confusion Matrix

1) *Accuracy*

Accuracy measures the overall correctness of the system in detecting and recognizing license plates. The OpenCV + Tesseract-based model achieved an accuracy of **85-90%** on standard datasets, with errors mainly occurring in low-light or occluded images.

2) *Precision & Recall*

- **Precision** (Positive Predictive Value): Measures how many detected plates were correctly identified. The system achieved **92% precision**, meaning most detected plates were correctly recognized.

- **Recall** (True Positive Rate): Indicates how many actual plates were detected. The recall was around **87%**, showing that some plates were missed due to distortions or reflections.

3) *F1-Score*

F1-score, the harmonic mean of precision and recall, was **89%**, indicating a good balance between false positives and false negatives.

4) *4. Processing Time*

The system's average detection and recognition time was **0.8–1.5 seconds per image** on a standard CPU. Using GPU acceleration could significantly improve this.

5) *Error Analysis*

- **Common Misclassifications**: Similar-looking characters (e.g., 'O' and '0', 'B' and '8') were occasionally misread.

- **Poor Lighting Issues**: Low-light images resulted in reduced contrast, making edge detection and OCR difficult.
- **Distorted Angles**: Perspective distortion affected character segmentation, reducing OCR performance.

6) *Comparative Performance with Deep Learning*

Replacing Tesseract OCR with a deep learning-based OCR model (like CRNN) improved accuracy to **95%**, reducing errors in character recognition. Using YOLO for plate detection also reduced false detections and improved recall.

These results highlight the effectiveness of the current approach while also demonstrating areas where deep learning and real-time optimizations can further enhance performance.

IV. Scope for Improvement

1) *Advanced Noise Reduction*

While traditional filtering methods like Gaussian and Median Blur help reduce noise, they often smooth out essential details. Using Wavelet Transform Denoising or Non-Local Means Filtering can preserve finer textures and enhance license plate visibility in noisy environments.

2) *Improved Segmentation Methods*

Current contour-based segmentation struggles with complex backgrounds. Deep learning-based segmentation using **U-Net** or **Mask R-CNN** can improve accuracy by learning shape patterns, enabling better separation of characters from the plate.

3) *Robust Edge Detection*

Traditional edge detection methods (Canny, Sobel) may fail under poor lighting conditions. A hybrid approach combining adaptive thresholding and deep edge detection models (HED – Holistically-Nested Edge Detection) can improve robustness against variable lighting and occlusions.

4) *Deep Learning-Based OCR*

Tesseract OCR often misclassifies characters due to font variations and distortions. Replacing it with CNN-based OCR models like CRNN (Convolutional Recurrent Neural Network) or Transformer-based OCR can enhance text recognition accuracy, especially for low-quality images.

5) *Real-Time Processing Optimization*

For real-time LPR applications, optimizing computational efficiency is essential. Implementing **YOLO (You Only Look Once) for fast detection**, **TensorRT for GPU acceleration**, and **model quantization techniques** can significantly reduce latency and improve inference speed.

V. Conclusion

License Plate Recognition (LPR) is a crucial technology for automated vehicle identification in traffic management and security applications. This study explored various preprocessing techniques, including noise reduction, edge detection, and contrast enhancement, to improve plate detection accuracy. The OpenCV and Tesseract-based approach achieved **85-90% accuracy**, with errors mainly due to poor lighting and distorted angles. Experimental results showed that deep learning-based OCR and advanced segmentation methods could significantly improve performance. Future enhancements, such as YOLO-based detection and GPU acceleration, can enable real-time, high-accuracy LPR systems suitable for large-scale deployment.

References

[1] Copyright (C) Symisc Systems, adminsymisc.net, http://symisc.net. "License Plate Detection without Machine Learning - SOD - PixLab," n.d. https://sod.pixlab.io/articles/license-plate-detection.html.

[2] Sebastián. "How to Clean Border Noise from a License Plate Using Filters," June 22, 2020. https://stackoverflow.com/questions/62507880/how-to-clean-border-noise-from-a-license-plate-using-filters.

[3] Adrian Rosebrock. "OpenCV: Automatic License/Number Plate Recognition (ANPR) With ...," September 21, 2020. https://pyimagesearch.com/2020/09/21/opencv-automatic-license-number-plate-recognition-anpr-with-python/.

[4] "Comparison of Image Preprocessing Techniques for Vehicle ... - arXiv," October 15, 2024. https://arxiv.org/abs/2410.13622.

[5] "License Plate Recognition - an Overview | ScienceDirect Topics," n.d. https://www.sciencedirect.com/topics/computer-science/license-plate-recognition.

[6] "Automatic Number Plate Recognition:A Detailed Survey of Relevant ...," n.d. https://pmc.ncbi.nlm.nih.gov/articles/PMC8123416/.

[7] "License Plate Recognition System-Survey," n.d. https://pubs.aip.org/aip/acp/article-pdf/1324/1/255/11503133/255_1_online.pdf.

[8] "[PDF] Research of License Plate Recognition Based on HSV Space," n.d. https://www.atlantis-press.com/article/25852244.pdf.

[9] "Vehicle Color Recognition Based on License Plate Color," n.d. https://ieeexplore.ieee.org/document/7016897/.

[10] "An Efficient Thresholding Algorithm for License Plate Recognition ...," n.d. https://ieeexplore.ieee.org/document/5138203.

[11] "An Automated License Plate Detection and Recognition System Based ...," n.d. https://www.sciencedirect.com/science/article/pii/S2590005620300254.

[12] "[PDF] Car License Plate Detection Using Edge and Color Information," n.d. https://idosi.org/wasj/wasj5(6)/17.pdf.

[13] *"License Plate Detection and Character Recognition Using Deep ...," December 17, 2024. https://arxiv.org/abs/2412.12572*.

[14] "A Real-Time License Plate Detection and Recognition Model In ...," n.d. https://pmc.ncbi.nlm.nih.gov/articles/PMC11086086/.