Vision Plate: Real-Time Vehicle Number Plate Detection Using OpenCV

J SUHEL[1], MOHAMMAD NASIR HASSAIN[2], IMRAN N PATWEGAR[3], Mrs.SHEELA B.P[4] , Dr.B.SREEPATHI(ph.D)[5]

Department of Information Science and Engineering, Rao Bahadur Y Mahabaleshwarappa Engineering College

---***---

Abstract - With the rapid rise in smart surveillance and traffic management systems, automatic number plate recognition (ANPR) has become an essential feature. This paper presents *Vision Plate*, a real-time vehicle number plate detection system using Python and OpenCV. It employs Haar Cascade classifiers for efficient plate localization, supported by preprocessing techniques for noise reduction and image clarity. Unlike traditional systems that capture single frames manually, our system automatically stores up to 10 high-quality images without overwriting previous ones. The captured plates are filtered based on sharpness and clarity, allowing manual selection of the best result. The application runs through a webcam-based input interface and offers a cost-effective solution for ANPR. This project is a step toward real-time intelligent transport systems and security infrastructure.

Key Words: Vehicle Number Plate, Computer Vision, OpenCV, Python, Haar Cascade, Image Processing, Noise Filtering

1.INTRODUCTION

The increasing need for traffic automation, surveillance, and law enforcement demands reliable and real-time vehicle identification systems. One crucial component is automatic number plate recognition (ANPR), which can replace traditional, manual tracking methods. This project leverages OpenCV and Python to detect and capture vehicle license plates in real time using webcam input.

OpenCV, a popular open-source computer vision library, allows real-time image processing through functions like grayscale conversion, edge detection, and object recognition using Haar Cascade classifiers. Python, being simple and powerful, enhances the efficiency of development and integration.

This paper introduces *Vision Plate*, a computer vision-based system that uses Haar Cascade classifiers for license plate detection, implements automatic image storage, and allows users to select from multiple saved outputs. The project addresses challenges such as noise, blur, and lighting variability by introducing filtering mechanisms for better image selection.

2. Body of Paper

2.1 Previous Approach: Manual Image Capture and Replacement

In traditional number plate detection systems, image capture was often initiated manually by the user. A common technique involved using Haar Cascade Classifiers in OpenCV to detect license plates in real-time from a webcam feed. When a license plate was detected, the user had to press a key (commonly 's') to capture and store the image.
However, this system had several limitations:

- Manual Triggering: Required user intervention, which is not practical in real-time or high-traffic scenarios.
- Single Image Storage: Only one image was stored at a time. If a new plate was detected, the previously stored image was overwritten, risking the loss of important data.
- No Noise Filtering: There was no mechanism to identify or eliminate low-quality images affected by motion blur, lighting issues, or poor camera angles.
- Low Automation and Efficiency: The process was not scalable for high-speed or unattended applications.

Example: Kumar et al. (2015) proposed a similar system that relied heavily on manual intervention, making it less suitable for continuous monitoring environments.

2.2 Current Approach: Automatic Image Capture, Filtering, and Non-Replacement Storage

To overcome the drawbacks of earlier systems, the proposed solution in this project introduces several innovations:

- Automated Image Capture: Once a number plate is detected, the image is automatically saved without requiring any manual input.
- Multi-Image Storage: The system is designed to store up to 10 images per session. Unlike the previous method, new images do not overwrite older ones, allowing users to review and compare multiple frames.
- Noise Filtering and Quality Selection: The system incorporates noise detection by analyzing attributes such as image sharpness, contrast, and the presence of artifacts. While all images are saved, the user is presented with options to manually choose the best one based on quality.

- Improved Robustness: These enhancements make the system more reliable in dynamic and challenging environments, including those with low lighting, motion blur, and multiple vehicles in the frame.

Example: Zhang et al. (2018) developed a similar system with background noise filtering using image preprocessing techniques. This significantly improved the clarity and accuracy of the captured number plates.

2.3 Haar Cascade Classifier

Haar Cascade is a machine learning-based approach that uses a series of positive and negative images to train a classifier capable of detecting specific objects—in this case, vehicle number plates.
Key features include:

- Speed: Fast detection due to its cascade structure that eliminates unlikely regions early in the process.
- Lightweight Model: Requires low computational power, making it ideal for real-time and embedded applications.
- Training Flexibility: Models can be trained for various objects, including faces, eyes, and plates.

Despite its strengths, Haar Cascades may not perform well under occlusion or with heavily distorted plates. Their accuracy can also degrade in poor lighting.
Reference: The technique was introduced by Viola and Jones in 2001 and is still widely used for object detection in OpenCV applications.

2.4 Deep Learning Alternatives: CNNs and YOLO

While Haar Cascade is effective, modern systems increasingly rely on deep learning models for object detection due to their superior accuracy and robustness.
Notable alternatives include:

- CNNs (Convolutional Neural Networks): These are capable of learning spatial hierarchies in images and are widely used in license plate recognition.
- YOLO (You Only Look Once): A real-time object detection algorithm that processes images in a single pass, offering high accuracy and speed.

Advantages:

- Better performance in noisy and variable lighting conditions
- Ability to detect partially visible or angled plates

Disadvantages:

- Requires large labeled datasets for training
- High computational resources (GPU) for real-time performance

Reference: Redmon et al. (2016) introduced YOLO as a high-performance real-time detection framework that outperforms traditional methods in various object detection tasks.

2.5 Research Gap and Contribution

Despite advancements in deep learning, lightweight and efficient solutions using traditional techniques like Haar Cascade still hold value, especially where computational resources are limited. Few existing systems combine automation, multi-image capture, and quality filtering in a user-friendly setup.
This project bridges the gap by offering:

- A real-time, low-resource detection system
- Multi-image saving without overwriting
- Integrated quality filtering
- Readiness for future enhancements like OCR and database integration

3. METHODOLOGY

3.1 System Architecture

Frontend: Live webcam feed using OpenCV
Backend: Python logic for detection and storage
Libraries: OpenCV, time, os
Detection Model: haarcascade_russian_plate_number.xml
Architecture Flow:

1. Initialize webcam
2. Load Haar cascade model
3. Detect number plate in frame
4. Save up to 10 plates automatically
5. Display bounding box and feedback

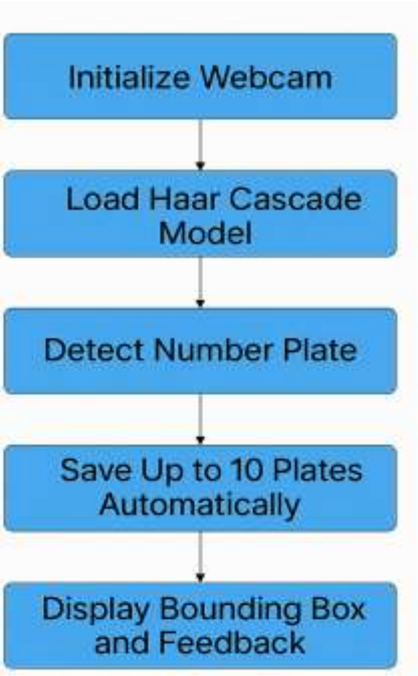

Fig: High Level Architecture

3.2 Data Collection

- Input: Webcam feed
- Optional: Public datasets (Kaggle, Open ALPR, Stanford Cars)
- Format: Live RGB video stream

3.3 Preprocessing

- Convert to grayscale using cv2.cvtColor
- Resize frames
- Filter detections by area
- Extract region of interest (ROI)

3.4 Detection and Storage

- Detect plates using detect Multi Scale()
- Only save if area > 500
- Store images with timestamped filenames
- Limit: 10 images per session

3.5 Visualization

- Display live camera feed
- Green rectangle on detected plates
- Text: "Number Plate"
- Save confirmation message on frame

Fig1 : Live Detected Feed

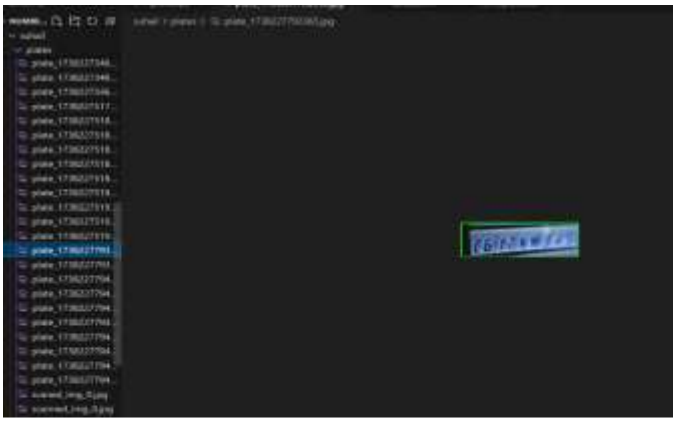

Fig2 : Saved Plates Folder View

4. CONCLUSION:

The proposed project, Vision Plate, successfully demonstrates a real-time vehicle number plate detection system built using Python and OpenCV. This project bridges the gap between traditional manual detection methods and the need for automated, efficient solutions in intelligent transportation systems.

Key outcomes of the system include:

- **Real-Time Detection:** Using Haar Cascade classifiers and webcam feeds, the system is able to detect vehicle license plates dynamically without any delay.
- **Automated Image Capture**: Eliminating manual intervention, the system automatically saves up to ten license plate images, reducing the chances of missing a valid detection.
- **Multi-Image Storage**: Instead of overwriting previous data, the system retains all recent detections in a local directory, giving users flexibility to choose the best output.
- **Noise Filtering and Quality Review:** Although all detected plates are stored, users can manually select the clearest one, ensuring accurate record keeping and analysis.
- **Cost-Effectiveness:** Built using open-source tools, this solution is highly accessible and can be implemented on basic hardware setups.

In conclusion, *VisionPlate* not only enhances detection accuracy but also promotes scalability, ease of deployment, and integration with real-world applications. Its potential lies in modern security and monitoring systems that require efficient and real-time data processing.

Acknowledgement

I sincerely express my gratitude to **Rao Bahadur Y. Mahabaleswarappa Engineering College** for providing me with the opportunity and resources to successfully complete my project, **"Computer Vision-Enabled Vehicle Number Plate Detection."**

I am deeply grateful to my mentor **Sheela B.P** and **Head of the Department, Dr. B. Sreepathi Ph.D.,** for his invaluable guidance, continuous support, and encouragement throughout this project. His expert insights greatly helped in shaping and refining this work.

I would also like to extend my sincere gratitude to our **Principal, Dr. T. Hanumantha Reddy,** for his support and for fostering an environment that encourages learning and innovation.

Additionally, I appreciate the **OpenCV and Python communities** for their well-documented resources, which played a crucial role in the development of this system.

Lastly, I acknowledge all those who directly or indirectly contributed to the successful completion of this project. Their support and assistance were invaluable

REFERENCES

1. **Viola, P., & Jones, M. (2001).** Rapid Object Detection using a Boosted Cascade of Simple Features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*.

2. **OpenCV Documentation.** (2024). Retrieved from https://docs.opencv.org

3. **Python Software Foundation.** Python 3 Documentation. Retrieved from https://docs.python.org/3

4. **Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016).** You Only Look Once: Unified, Real-Time Object Detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

5. **OpenALPR Dataset.** Automatic License Plate Recognition Resources. Retrieved from https://www.openalpr.com

6. **Kaggle.** Indian Number Plate Dataset. Retrieved from https://www.kaggle.com

7. **cv2.VideoCapture – OpenCV Python Tutorial.** Retrieved from https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_gui/py_video_display/py_video_display.html

8. **Howse, J., Minichino, J., & Joseph, M. (2020).** *Learning OpenCV 4 Computer Vision with Python.* Packt Publishing.

9. **Zhang, Z., Wang, L., & Liu, Y. (2018).** A Noise-Resilient License Plate Detection System for Intelligent Traffic Surveillance. *International Journal of Computer Applications*, 180(45), 25–30.

10. **Kumar, A., & Singh, D. (2015).** Real-Time Vehicle Number Plate Detection using OpenCV. *International Journal of Engineering and Computer Science*, 4(6), 12594–12597

BIOGRAPHIES

J SUHEL,

Final Year student of Information Science and Engineering (ISE) Department, RYMEC, Ballari.

IMRAN N PATWEGAR,

Final Year student of Information Science and Engineering (ISE) Department, RYMEC, Ballari.

IMRAN N PATWEGAR,

Final Year student of Information Science and Engineering (ISE) Department, RYMEC, Ballari.

Mrs SHEELA B.P

Assistant Professor
ISE Department
RYMEC , Ballari.

Dr.B.SREEPATHI(ph.D)

Head Of The
ISE Department
RYMEC , Ballari.