Automatic Detection and Recognition of Vehicle License Plates using Real-Time Image Processing and Optical Character Recognition

By: Vishnu Vardhan Kashyap Madhav Sai Chetan Reddy

***Abstract*— This paper presents a system for real-time vehicle license plate detection and recognition from live video streams using image processing and Optical Character Recognition (OCR). The proposed method leverages OpenCV for contour detection and EasyOCR for text extraction, mapping the registration state based on the first two characters of the detected license plate. The system identifies, processes, and annotates license plate information in real-time with high accuracy, supporting applications in intelligent transportation and automated monitoring.**

***Keywords*— License plate recognition, Optical Character Recognition, Image processing, OpenCV, EasyOCR, Real-time detection, Contour-based detection**

I. Introduction (*Heading 1*)

The development of Automatic License Plate Recognition (ALPR) technology has transformed traffic monitoring, law enforcement, and automated toll systems. Traditional ALPR systems typically require expensive, proprietary hardware and software configurations, posing challenges for deployment at scale. This paper proposes a lightweight, real-time solution built with OpenCV and EasyOCR libraries in Python. The system is designed to capture video from a live camera feed, process the frames to detect a license plate, and extract state-based vehicle registration information.

II. Methodology

A. *System Overview*

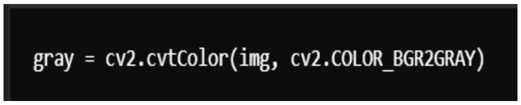
The system captures real-time video from a camera, detects vehicle license plates in each frame, and extracts text from the detected regions. The detected plate number's first two characters are mapped to the corresponding Indian state for display. The methodology relies on OpenCV for image processing and EasyOCR for text recognition.

B. *Image Processing Pipeline*

The image processing pipeline consists of several steps to ensure accurate contour detection and noise reduction.

1) *Grayscale Conversion*

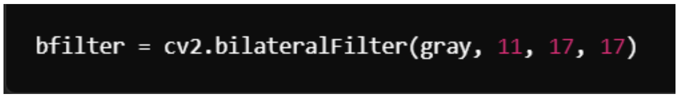
Each frame is first converted to grayscale to reduce computational complexity and improve contour detection accuracy.



```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

2) *Noise Reduction*

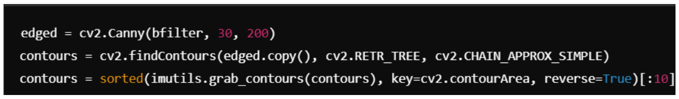
Bilateral filtering is used to reduce noise while preserving edges, which is crucial for detecting license plates under varying lighting conditions



```
bfilter = cv2.bilateralFilter(gray, 11, 17, 17)
```

3) *Edge Detection and Contour Finding*

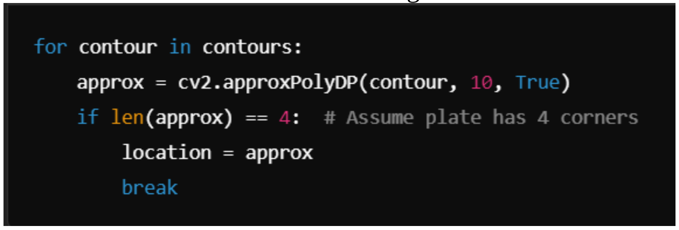
The Canny edge detection algorithm identifies the edges, and contours are located to isolate potential license plate regions. The contours are sorted by area to prioritize larger, more prominent ones that are likely to be license plates.



```
edged = cv2.Canny(bfilter, 30, 200)
contours = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
contours = sorted(imutils.grab_contours(contours), key=cv2.contourArea, reverse=True)[:10]
```

4) *License Plate Localization*

A quadrilateral contour is assumed to be a license plate. If such a contour is found, it is extracted and passed to the OCR module for text recognition.



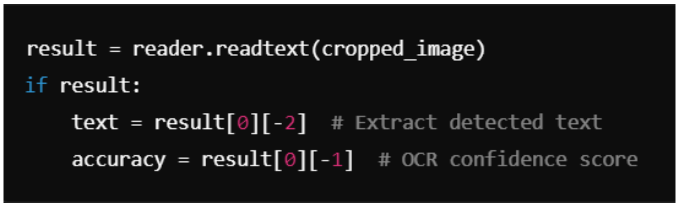
```
for contour in contours:
    approx = cv2.approxPolyDP(contour, 10, True)
    if len(approx) == 4:  # Assume plate has 4 corners
        location = approx
        break
```

C. *Text recognition using EasyOCR*

Once the license plate region is detected, EasyOCR is used to perform text recognition on the extracted region. The detected

Identify applicable funding agency here. If none, delete this text box.

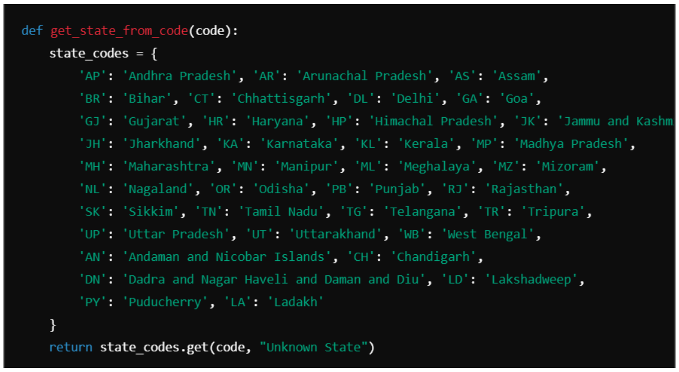
text is then processed to identify the state code based on the first two characters.



```
result = reader.readtext(cropped_image)
if result:
    text = result[0][-2]  # Extract detected text
    accuracy = result[0][-1]  # OCR confidence score
```

D. *Mapping state code to state code*

The get_state_from_code function maps the first two characters of the detected text to the respective Indian state using a dictionary.



```
def get_state_from_code(code):
    state_codes = {
        'AP': 'Andhra Pradesh', 'AR': 'Arunachal Pradesh', 'AS': 'Assam',
        'BR': 'Bihar', 'CT': 'Chhattisgarh', 'DL': 'Delhi', 'GA': 'Goa',
        'GJ': 'Gujarat', 'HR': 'Haryana', 'HP': 'Himachal Pradesh', 'JK': 'Jammu and Kashm
        'JH': 'Jharkhand', 'KA': 'Karnataka', 'KL': 'Kerala', 'MP': 'Madhya Pradesh',
        'MH': 'Maharashtra', 'MN': 'Manipur', 'ML': 'Meghalaya', 'MZ': 'Mizoram',
        'NL': 'Nagaland', 'OR': 'Odisha', 'PB': 'Punjab', 'RJ': 'Rajasthan',
        'SK': 'Sikkim', 'TN': 'Tamil Nadu', 'TG': 'Telangana', 'TR': 'Tripura',
        'UP': 'Uttar Pradesh', 'UT': 'Uttarakhand', 'WB': 'West Bengal',
        'AN': 'Andaman and Nicobar Islands', 'CH': 'Chandigarh',
        'DN': 'Dadra and Nagar Haveli and Daman and Diu', 'LD': 'Lakshadweep',
        'PY': 'Puducherry', 'LA': 'Ladakh'
    }
    return state_codes.get(code, "Unknown State")
```

III. Results and Discussions

The system was tested in a controlled environment with standard lighting conditions, successfully detecting license plates with high OCR confidence scores. Sample output frames displayed license plate text, accuracy percentages, and the state of registration.

A. *Limitations*

1. **Lighting Conditions:** Performance decreased under poor lighting or glare, affecting OCR accuracy.

2. **Non-Standard Fonts**: Plates with stylized fonts or obstructions resulted in lower recognition accuracy.

3. **Environmental Factors**: Environmental variables, such as vehicle speed and motion blur, impacted detection consistency.

B. *Proposed Improvements*

Future improvements could include integrating a machine learning model trained for license plate detection and character segmentation to enhance accuracy under various conditions. Additionally, real-time color or adaptive contrast adjustments could improve readability under challenging lighting conditions

IV. Conclusion

This paper presented a real-time, cost-effective license plate recognition system using open-source libraries. The system successfully extracted license plate numbers and mapped them to registration states in real time. Future work may explore the use of deep learning techniques to improve recognition in unconstrained environments

Acknowledgment *(Heading 5)*

The authors would like to thank OpenCV and EasyOCR communities for their resources and support.

References

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

[1] R. Smith, "An Overview of the Tesseract OCR Engine," in *Proc. Ninth Int. Conf. on Document Analysis and Recognition (ICDAR)*, Parana, Brazil, 2007, pp. 629-633.

[2] G. Bradski, "The OpenCV Library," *Dr. Dobb’s Journal of Software Tools*, 2000.

[3] EasyOCR Documentation, "EasyOCR: A Ready-to-Use Optical Character Recognition with 80+ Supported Languages," Available: https://github.com/JaidedAI/EasyOCR.

[4] X. Zhai, Y. Wu, and Z. Fang, "License Plate Recognition Based on Machine Learning and Segmentation-Free Methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 2, pp. 728–739, 2020.

[5] S. Du, M. Ibrahim, M. Shehata, and W. Badawy, "Automatic License Plate Recognition (ALPR): A State-of-the-Art Review," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 311–325, 2013.

[6] A. Rafique, A. Siddiqui, H. Ashraf, and M. A. Azam, "Real-time License Plate Recognition System Using OpenCV and Tesseract," in *Proc. International Conference on Computer and Information Sciences (ICCOINS)*, Kuala Lumpur, Malaysia, 2016, pp. 367–372.

[7] H. Kim and D. Lee, "Character Segmentation and Recognition for Automatic License Plate Recognition Using a Probabilistic Neural Network," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 1–10, 2017.

[8] S. M. Silva and C. R. Jung, "License Plate Detection and Recognition in Unconstrained Scenarios," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 3, pp. 596–609, 2017