Dissertation on
Automatic Vehicle Number Plate Detection System

Thesis submitted towards partial fulfilment
of the requirements for the degree of

Master in Multimedia Development

Submitted by
Subhrajyoti Chakraborty

EXAMINATION ROLL NO.: M4MMD24003B
UNIVERSITY REGISTRATION NO.: 163784 of 2022-2023

Under the guidance of
Prof. Dr. Matangini Chattopadhyay
School of Education Technology
Jadavpur University

Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata-700032
India

2024

Master in Multimedia Development
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF RECOMMENDATION

This is to certify that the thesis entitled **"**Automatic Vehicle Number Plate Detection System**"** is a bonafide work carried out by Subhrajyoti Chakraborty under our supervision and guidance for partial fulfillment of the requirements for the degree of Master in Multimedia Development in School of Education Technology , during the academic session 2023-2024.

SUPERVISOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DIRECTOR
School of Education Technology
Jadavpur University,
Kolkata-700 032

DEAN - FISLM
Jadavpur University,
Kolkata-700 032

Master in Multimedia Development
Course affiliated to
Faculty of Engineering and Technology
Jadavpur University
Kolkata, India

CERTIFICATE OF APPROVAL **

This foregoing thesis is hereby approved as a credible study of an engineering subject carried out and presented in a manner satisfactory to warranty its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not endorse or approve any statement made or opinion expressed or conclusion drawn therein but approve the thesis only for purpose for which it has been submitted.

Committee of final examination for evaluation of thesis

** Only in case the thesis is approved.

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this thesis contains literature survey and original research work by the undersigned candidate, as part of his **Master in Multimedia Development** studies.

All information in this document has been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by this rule and conduct, I have fully cited and referenced all materials and results that are not original to this work.

NAME: Subhrajyoti Chakraborty

EXAMINATION ROLL NUMBER: M4MMD24003B

THESIS TITLE: Automatic Vehicle Number Plate Detection System

SIGNATURE: DATE:

Acknowledgement

Firstly, I would like to express my sincere gratitude to my supervisor Prof. Dr. Matangini Chattopadhyay for her continuous support in my thesis work and related research. Her guidance and valuable suggestions always helped me at time of research and writing this thesis. I could not have imagined having a better advisor and mentor for my Master Degree study. I am very much thankful to her for the motivation and support given during the entire duration of research work.

Besides this, I would like to thank Dr. Saswati Mukherjee, Mr. Joydeep Mukherjee and Ms. Angsumitra Ghosh for their continuous encouragement and support during my entire period of study in the School of Education Technology. I would like to express my gratitude to the entire staff, Lab Assistant. I am very grateful to all my classmates of Master in Multimedia Development and M. Tech IT (Courseware Engineering).

I would like to thank my parents for always supporting me in every ups and downs during my entire period of work. I am also thankful to my friends and well wishers who always have faith in me.

Subhrajyoti Chakraborty
Examination Roll No: M4MMD24003B
Univ. Registration No. 163784 of 2022 2023
Master in Multimedia Development
School of Education Technology
Jadavpur University,
Kolkata – 700032

Dedicated to,
My Parents

Contents

List of Figures

List of Tables

List of Abbreviations

OCR- Optical Character Recognition

ANPR- Automatic Number Plate Recognition

CCTV- Closed Circuit Television

IDE- Integrated Development Environment

JPEG- Joint Photographic Experts Group

PNG- Portable Network Graphic

RGB- Red, Green, and Blue

GUI- Graphical User Interface

LPD- License Plate Detection

AHE- Adaptive Histogram Equalization

CLAHE- Contrast Limited Adaptive Histogram Equalization

NLP- Natural Language Processing

HSV- Hue, Saturation, and Value

ROI- region of interest

CV- Computer Vision

YOLO- You Only Look Once

R-CNN- Region-based Convolutional Neural Network

Executive Summary

Optical Character Recognition (OCR) for vehicle number plate recognition is a technology that translates images of license plates into machine-readable text. With India's growing population, the number of unlicensed vehicles on the road has sharply increased, adding to the already existing traffic issues and even boosting crime rates. This situation has created an urgent need for a system that can efficiently handle traffic violations and ensure only authorized vehicles can entry in the secured areas. Varies on Manual processes of parking management are costly and time consuming. A better solution is need of the hour that can automatically capture and process license plate information.

This project provides an automatic system for detecting and recognizing text on vehicle number plates using OpenCV for image processing and Tesseract OCR for text extraction. The system begins by setting up the working directory and loading images in standard formats. Each image undergoes preprocessing, including grayscale conversion, noise reduction using a Bilateral Filter and edge detection through the Canny method. To identify potential number plates, contours with rectangular shapes and appropriate aspect ratios are selected. Detected regions are processed with Tesseract OCR to extract text, which is displayed alongside confidence scores. Visualization steps allow for comparison of the original, grayscale, and processed images, with highlighted contours and cropped plate regions. Each image in the directory is processed through this workflow, and results are presented with both intermediate visuals and final extracted texts, along with accuracy scores for each image's OCR output. This method provides an efficient approach to automate license plate detection and recognition with transparency in intermediate results and confidence metrics.

There is the possibility that this system could be utilised in application related to track vehicles for compliance and criminal activity. Also, automation reduces the manual workload and processing time, offering real-time insights.

CHAPTER 1

1.0 Introduction

1.1 Overview

Automatic Number Plate Recognition (ANPR) steps in as a smart solution. This system works by capturing an image of the license plate and extracting the characters using advanced image processing techniques. The process involves several key steps, including resizing the image, converting it to grayscale, enhancing and restoring it, and finally detecting and recognizing objects. This research work also uses methods like Canny edge detection and bilateral filtering to refine the image before utilizing the Tesseract engine to read a license plate.

As urbanization continues to rise, especially in countries like India and China, vehicle ownership has skyrocketed, placing enormous pressure on parking spaces and traffic systems. For instance, between 2001 and 2019, the number of private vehicles in India quadrupled. Current parking systems, which often depend on manual entry, are slow, inefficient, and can compromise security. Managing vehicle entry manually, especially in busy areas, takes too much time and leaves room for error.

An automatic parking system powered by ANPR could solve these problems. It removes the need for manual work by automatically detecting and recording license plates via CCTV footage. This ensures faster, smoother vehicle check-ins and check-outs while maintaining top-level security. Moreover, it can instantly verify whether a vehicle is authorized to enter, offering a much quicker and safer alternative to traditional methods.

1.2 Problem Statement

Automatic Vehicle Number Plate Detection System

1.3 Objectives

The objectives of my research work are as follows –

- To study Optical Character Recognition system such as EasyOCR, pytesseract

- To study research papers on text extraction from vehicle number plate
- Learn python programming for implementation.
- To get familiarity with Jupyter Notebook (IDE of python programming)

1.4 Assumptions and Scopes

1.4.1 Assumptions

- Developers must have good network connection with laptops or PC
- Developers must have good knowledge of Python
- Image forma such as JPEG, JPG and PNG have been considered

1.4.2 Scopes

- To design and develop automatic number plate detection system
- To learn about OCR, Python programming

1.5 Concept and Problem Analysis

This research work presents how numbers is extracted from vehicle license number plate, Using Python (Jupyter IDE) we have extracted numbers from various car images. Also we have used Pytesseract to fetch numbers into text.

The main benefit of OCR for vehicle number plate recognition is automation in vehicle identification. By automatically capturing and recognizing license plates from images or video feeds, OCR eliminates the need for manual data entry, which saves time, reduces human error, and improves efficiency in multiple fields. This automated recognition provides several key advantages:

- Enhanced Security and Law Enforcement

- Improved Traffic Flow and Management
- Efficient Toll Collection
- Streamlined Parking Systems

1.6 Organization of the Thesis

- Chapter 1 – This chapter contains the introduction of the thesis which includes overview, problem statement, objectives, assumptions, scopes, concept and problem analysis
- Chapter 2 – It includes all the literature surveys done to carry out the research work.
- Chapter 3 – It includes proposed approach of the methods that we have use to extract text.
- Chapter 4 – This chapter contains the implementation and result.
- Chapter 5 – This chapter describes the conclusion and scope of future works.
- References contain list of all the research papers considered in this work.
- Appendix – It contains listing of codes for the number plate detection system.

CHAPTER 2

2. Literature Survey

The Vehicle Number Plate Reader System is a highly effective tool for automobile identification. A variety of research papers are reviewed to gather valuable insights into relevant applications. This section discusses the methodologies, approaches, and solutions proposed by different studies over the past decades, specifically related to OCR and license plate detection.

Milan, Samantaray, et al. [1] have proposed an automatic vehicle license plate recognition, This system uses a structured approach that involves training, validating, and repeating the learning process on data collected from license plates. The primary objective is to accurately detect and recognize the alphanumeric characters on a license plate, facilitated by various stages of image processing and OCR. In this system, the image should be taken from an optimal distance (around 2 to 3 feet) to ensure image clarity. Here, extraction is done by processing of images and detection of license plate by findcontours() then by character recognition.

U, Salimah, et al. [2] have suggested a systematic approach for license plate recognition that relies on preprocessing, segmentation, and normalization for optimized OCR performance. In the preprocessing stage, the image is first enhanced to isolate the license plate area from the background, involving operations like cropping, grey scaling, and binary thresholding. Gray scaling reduces the colour complexity, and binarization further simplifies the image by converting it to black and white, making it easier to distinguish characters from the background. Also testing on a sample of 100 license plates showed that the system achieved a 75% success rate in plate recognition, accurately identifying 75 out of 100 plates. In terms of character recognition, the system demonstrated a high level of accuracy out of 722 total characters across these plates, 703 were correctly recognized, resulting in a character recognition accuracy rate of 97.36%. This indicates the system's effectiveness in accurately identifying individual characters, even if complete plate recognition was slightly lower.

M.L.S.N.S, Lakshmi, et al. [3] proposed a vehicle license plate recognition system, emphasizing image preprocessing, localization, and OCR techniques. In this system an infrared camera captures the image from a few meters away, ensuring clarity and high detail. To reduce processing complexity, the captured image is scaled to an optimal aspect ratio before being converted from RGB to grayscale, which simplifies the data by removing colour information, making it easier to process the structural components of the license plate. Then,

localization to isolate the relevant areas of the image, retaining only the portions likely to contain the license plate while discarding irrelevant background elements. Following localization, the image is converted into a binary format, enabling better separation between plate characters and the background. Edge detection techniques are applied to identify the boundaries within the binary image, helping to highlight the structural elements of the plate. A key component of this process is component analysis, which assesses connected areas (contours) within the image to identify potential locations for the license plate. This technique helps to maximize the accuracy of plate detection by analyzing the spatial distribution of possible plate candidates.

Shivani, Bansal, et al. [4] reviewed OCR the most widely used technique for extracting text from images particularly for recognizing vehicle license plates. OCR plays a vital role in vehicle identification systems, enabling seamless integration with document management systems to verify and match license plates against recorded data. OCR can be used to identify stolen vehicles and can be used to reduce crime over the roads/ highways. Various algorithms and methods for OCR have been proposed, each improving the accuracy and efficiency of character recognition under different conditions, such as varying lighting, angles, and plate distortions.

Prakhar, Sisodia, Syed Wajahat Abbas, Rizvi [5] have suggested that text extraction can be done not only on images but also from Real-Time OCR applications like (video surveillance, video-license plate recognition, and live document scanning), also extraction of Handwriting Recognition. They also stated that researchers are working on multimodal OCR systems which combine image recognition with other modalities such as speech recognition and natural language processing (NLP).

Mohammed, AS, Shariff, et al. [6] have proposed Number plate extraction for vehicles gathering all contours and detecting top 30 contours followed by inputing image in the form of RGB then conversion of RGB to grey scale image. Conversion of bilateral filter to remove background noise while preserving the edges. They found the text obtained as similar as licensed plate, Over 100 images they got 88 corrected number plates resulting 88% overall accuracy. They apply Tesseract to read the output cropped image for better result and accuracy.

Manpreet, Kaur, et al. [7] have used MATLAB instead of Python. They have proposed a Region-props algorithm to measure the properties of image regions. Region-props (BW, properties) return measurements for the set of properties specified by properties for each connected component in the binary image. They achieve vehicle number plate detection including even/odd identification and

state detection where a series of steps are implemented. First, a video file containing the number plate is created and opened within a graphical user interface (GUI). The process begins by converting the input image from RGB to grayscale, where the pixel values are transformed based on their Red, Green, and Blue (RGB) components. This grayscale conversion simplifies the image for further processing. Next, noise such as salt and pepper artifacts is removed while preserving the sharpness of the image using filtering techniques. Once the image is cleaned, the number plate is localized and OCR is performed. This involves segmenting the image, extracting features, and recognizing the characters, which range from 0-9 and A-Z. After OCR, the recognized number plate information is saved into a text file (.txt), which includes details such as the number, state, and whether the number is even or odd. Finally, the text file can be opened to display the results showing the number plate's digital image along with the state and even/odd information.

Shambharkar, Yash, et al. [8] suggested that deep learning techniques offer significant advantages for Automatic Number Plate Recognition (ANPR) in India especially, when dealing with anomalies. The research presents a comprehensive end-to-end ANPR pipeline, highlighting challenges such as multiple license plate lines, non-uniform padding, varying plate shapes, fonts, and font sizes. The authors propose a License Plate Detection (LPD) model tailored for Indian conditions including an alternative method for CR networks that suits the local environment. The LPD model achieved an accuracy of 96.23%, with a detection threshold of 0.5, and performed exceptionally well on smaller license plates found on cars and buses, maintaining a 98% accuracy rate and a loss of less than 10% at a 92% learning rate. The proposed network exhibited an average accuracy of 94.9%, successfully handling multi-line plates and predicting 93.7% of the characters with over 90% confidence.

Nithin K., Shine, Gariman, Bhutani, Tamatapu, Sai Keerthana, and G., Rohith [9] used a pixel-intensity histogram which is a key tool for understanding the distribution of an image’s intensity across its pixel, highlighting how many pixels correspond to each intensity value. They used Histogram Equalization as a pre-processing technique in OCR that enhances the contrast of images to improve OCR accuracy. This technique spreads the most frequent intensity levels across the tonal range, effectively expanding the image's tonal spectrum. They achieved binarization using the Otsu method. Also, they apply Adaptive Histogram Equalization (AHE) which creates multiple histograms for different regions of an image, improving local contrast and edge sharpness and a further enhancement. Contrast Limited Adaptive Histogram Equalization (CLAHE), limits contrast more than AHE. CLAHE addresses the issue of noise amplification by limiting contrast on smaller regions (tiles) of the image before computing the transformation function. This technique is particularly effective

in localizing grayscale features and improving image readability for OCR. CLAHE can also be applied to colour images, particularly in the HSV colour space, where only the luminance channel is equalized, preserving colour accuracy while enhancing contrast.

Nita M., Thakare, et al. [10] used Median Filter instead of Bilateral Filter. They also used Threshold algorithm (to separate the object from a background image and is converted in binary form). Gray level threshold is a simple process. The value of threshold (T) is selected and compared with the pixel of the image. It also transforms the input image (K) into an output binary image (F) which is being segmented. In global threshold, the histogram of the image is partitioned using a single threshold value. Followed by Morphological Image Processing which use dilation and adding pixels to the boundary of the object to increase the thickness of the edges. Using Shrinking operation thinning the image to eliminate irrelevant parts.

CHAPTER 3

3.0 Proposed Approach

3.1 Methodology

The proposed Automatic Vehicle Number Plate Detection System using Python and OpenCV consists of the following key steps:

- Image Preprocessing
- Grayscale Conversion
- Noise Reduction with Bilateral Filtering
- Edge Detection with Canny Edge Detector
- Contour Detection and Analysis
- License Plate Extraction and Cropping

3.1.1 Image Preprocessing

The input image is first loaded using the OpenCV library's cv2.imread() function, which reads the image data into a format suitable for processing. Image loading forms the basis for subsequent stages, providing a visual representation for pixel-level operations. Given the varying resolutions and qualities of captured images, resizing is applied to standardize the image dimensions to a width of 500 pixels. This standardization, implemented using the imutils.resize() function, improves processing speed and reduces the impact of scale on subsequent detection processes. Also required to install pytesseract to convert image into string. This step reduces computation time and optimizes the system for processing images with varying resolutions.

3.1.2 Grayscale Conversion

For computational efficiency and to simplify the data structure, the image is converted from an RGB (Red-Green-Blue) color format to grayscale using cv2.cvtColor(image, cv2.COLOR_BGR2GRAY). Grayscale conversion reduces the image to a single intensity channel, thereby lowering the

computational requirements. This step is essential, as most license plate detection techniques rely on intensity-based features rather than color information. By using grayscale, we maintain the structural details necessary for edge and contour analysis while eliminating redundant color data.

3.1.7 Noise Reduction with Bilateral Filtering

Noise in an image can lead to incorrect contour detection and poor edge representation, especially in images with complex backgrounds or varying lighting conditions. To address this, a bilateral filter is applied using cv2.bilateralFilter(gray, 11, 17, 17). Unlike standard blurring, the bilateral filter preserves edges by considering both spatial proximity and intensity similarity. This quality is crucial in license plate detection, as it enhances the clarity of the plate edges while removing irrelevant noise from the background, yielding a cleaner image for edge detection.
(11, 17, 17) describe as 11: Diameter of each pixel neighbourhood used during filtering. A larger diameter means more pixels are involved in computing the new pixel value.
17 and 17: Sigma values for colour and space respectively. The higher these values, the more significant the smoothing effect.
17 (for colour) controls how colours within the neighbourhood influence each other. Larger values mean that more colours will blend together.
17 (for space) controls how much nearby pixels (within the specified diameter) influence each other.

3.1.8 Edge Detection with Canny Edge Detector

Edge detection is a pivotal step in the process, as it identifies the boundaries of objects within the image, including the license plate. The Canny edge detection method is used (cv2.Canny(gray, 170, 200)) due to its effectiveness in detecting strong gradients, which often indicate object edges.

The method applies two thresholds the first threshold 170 is the lower threshold which helps detect edges by marking pixels with gradient intensities above this value as potential edges. . The second threshold 200 is the upper threshold any pixel with a gradient intensity above this value is considered a strong edge and retained in the final output.

Proper tuning of the threshold values (in this case, 170 and 200) is necessary to accurately capture plate edges while ignoring irrelevant details.

3.1.9 Contour Detection and Analysis

Contours are continuous lines or curves that bound the shape of objects, making them ideal for identifying regions like license plates. Using cv2.findContours() in OpenCV, contours are detected in the edge map generated from the previous step. The detected contours are then sorted by area, allowing the system to focus on larger objects likely to contain the license plate. To further refine the selection, the system filters for contours with four sides, a common characteristic of license plate shapes. This selection method enhances accuracy by discarding irrelevant shapes and narrowing down the contours to potential plate regions.

3.1.10 License Plate Extraction and Cropping

Upon identifying the most likely contour, the region of interest (ROI) corresponding to the license plate is extracted. Using cv2.boundingRect(), the coordinates of the contour are calculated to define a bounding rectangle around the detected plate area. The region within this rectangle is then cropped and saved as a separate image for further processing. This extraction isolates the license plate from the background, focusing the OCR engine on the relevant text content.

3.2 Flowchart of the proposed system

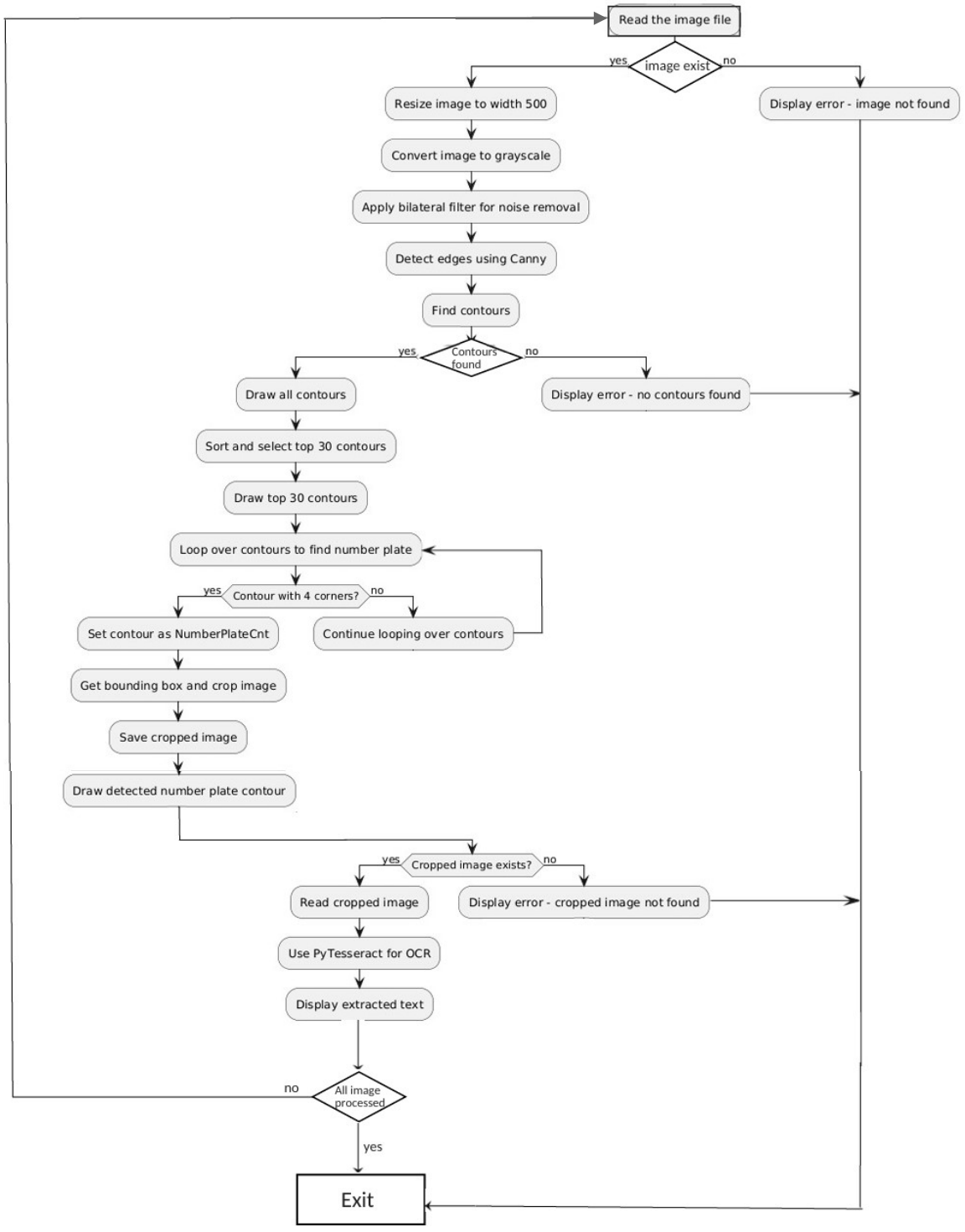

Figure 1: Flowchart of the proposed system

From this Figure 1 the flowchart describes the process of detecting and reading a vehicle's number plate from an image. The process starts by reading the image file. If the image exists it is resized to a width of 500 pixels for standardization,

then converted to grayscale. A bilateral filter is applied to reduce noise while preserving edges. Canny edge detection is used to highlight edges and contours are identified in the resulting image. If contours are found, they are drawn and sorted to select the top 30 contours, which are then drawn as well.

The program then loops over the top contours to identify a contour with four corners, which is assumed to be the number plate. When a suitable contour is found, it is set as Number Plate Count and the bounding box is extracted to crop the number plate region. This cropped image is saved and the detected number plate contour is drawn on the original image.

Next, the program checks if the cropped image of the number plate exists. If it does the image is read and OCR is applied using PyTesseract to extract text from the number plate. The extracted text is displayed completing the OCR process. If all image are being processed successful the program exits otherwise it backs to read image again.

CHAPTER 4

4.0 Experimentations and Results

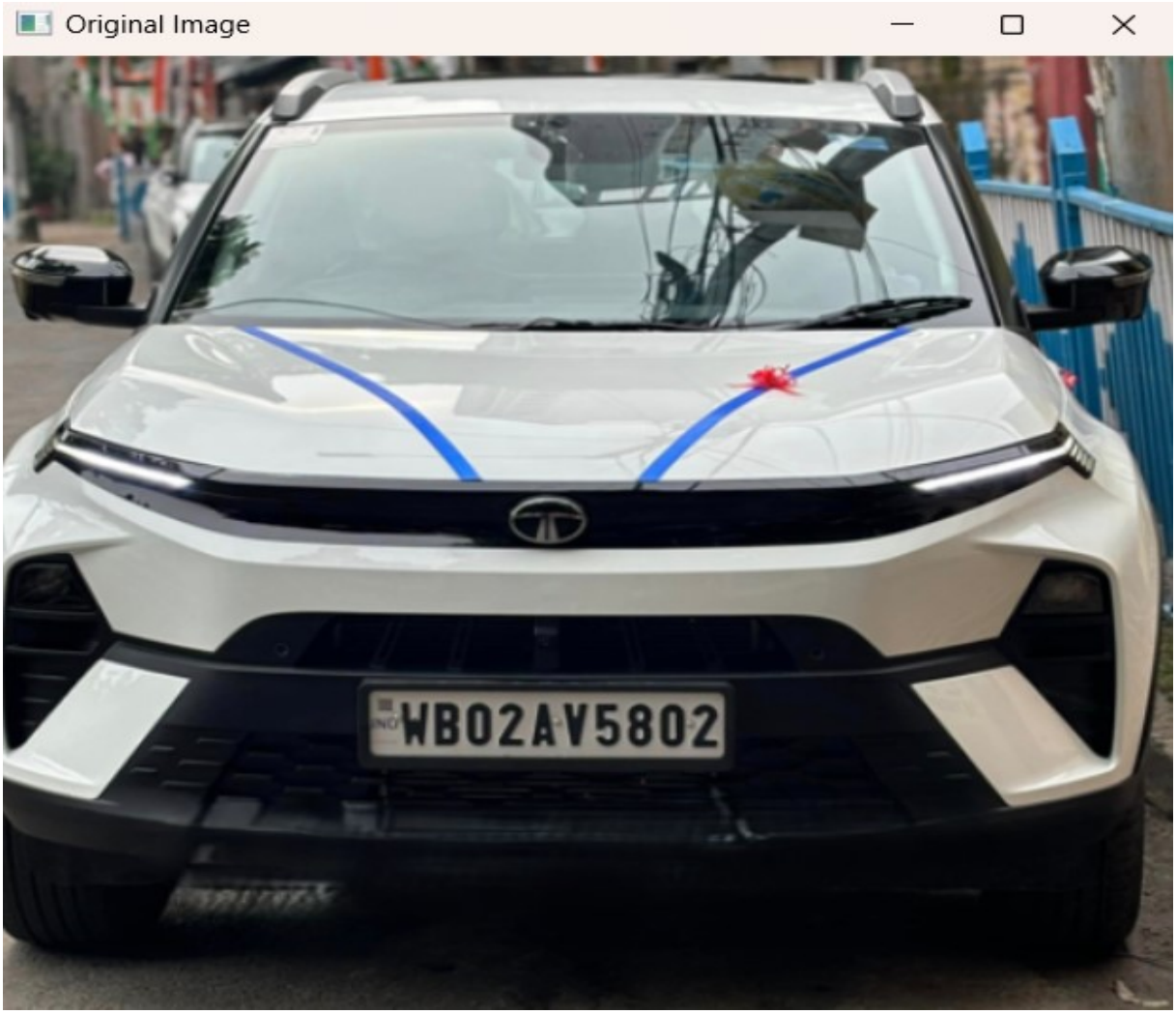

Figure 2: Original Image

In this Figure 2 Original Image the raw, unprocessed version of the image showing the scene as captured by the camera which includes the vehicle and surroundings.

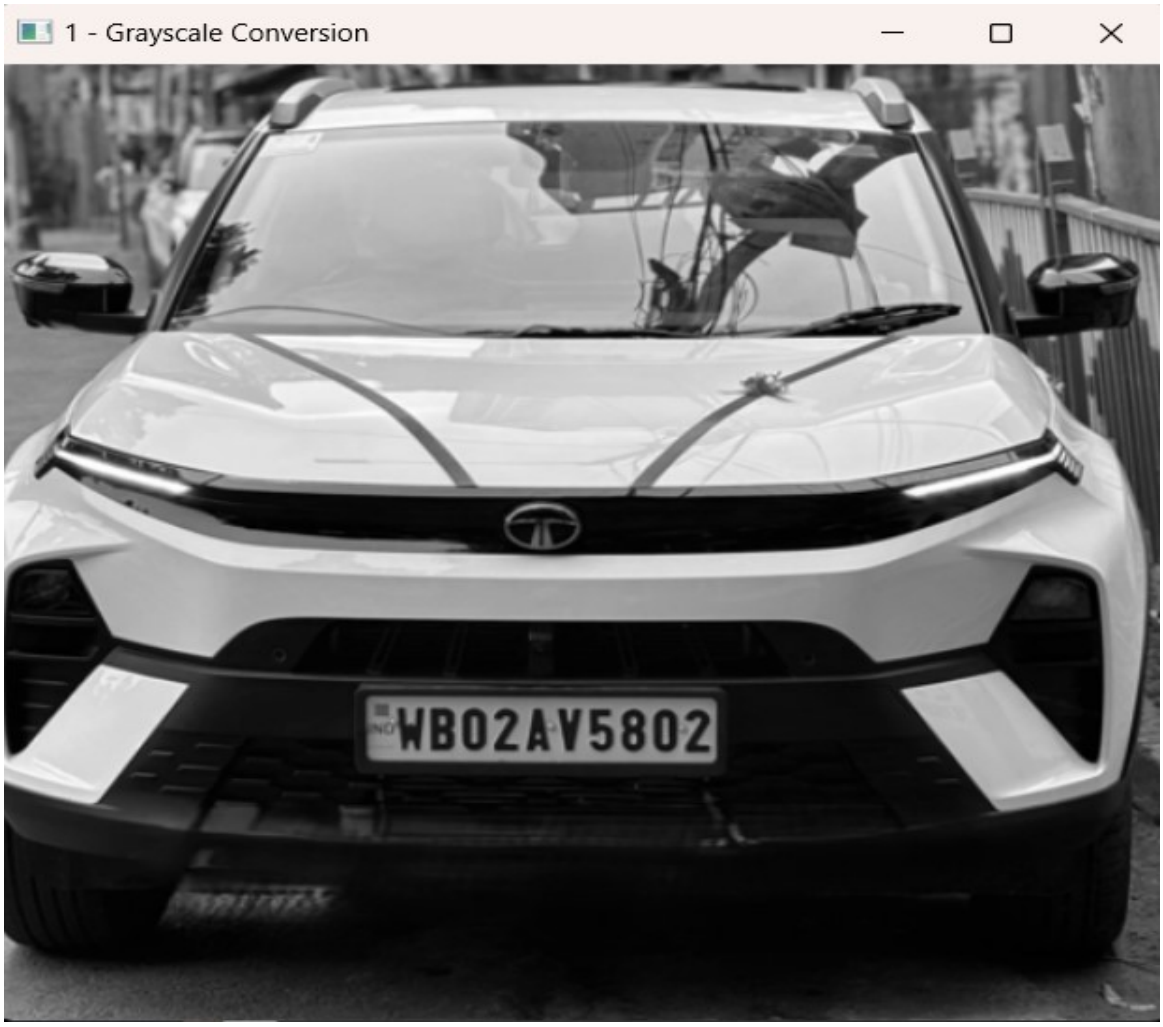

Figure 3: Original Image to Gray Image

In this Figure 3 the Original image is converted to Grayscale, removing colour information which simplifies further processing and focuses on intensity differences.

Figure 4: After applying bilateral filter

In this Figure 4 the grayscale image with a bilateral filter applied, which smoothens the image while preserving edges, reducing noise without losing important details around object boundaries.

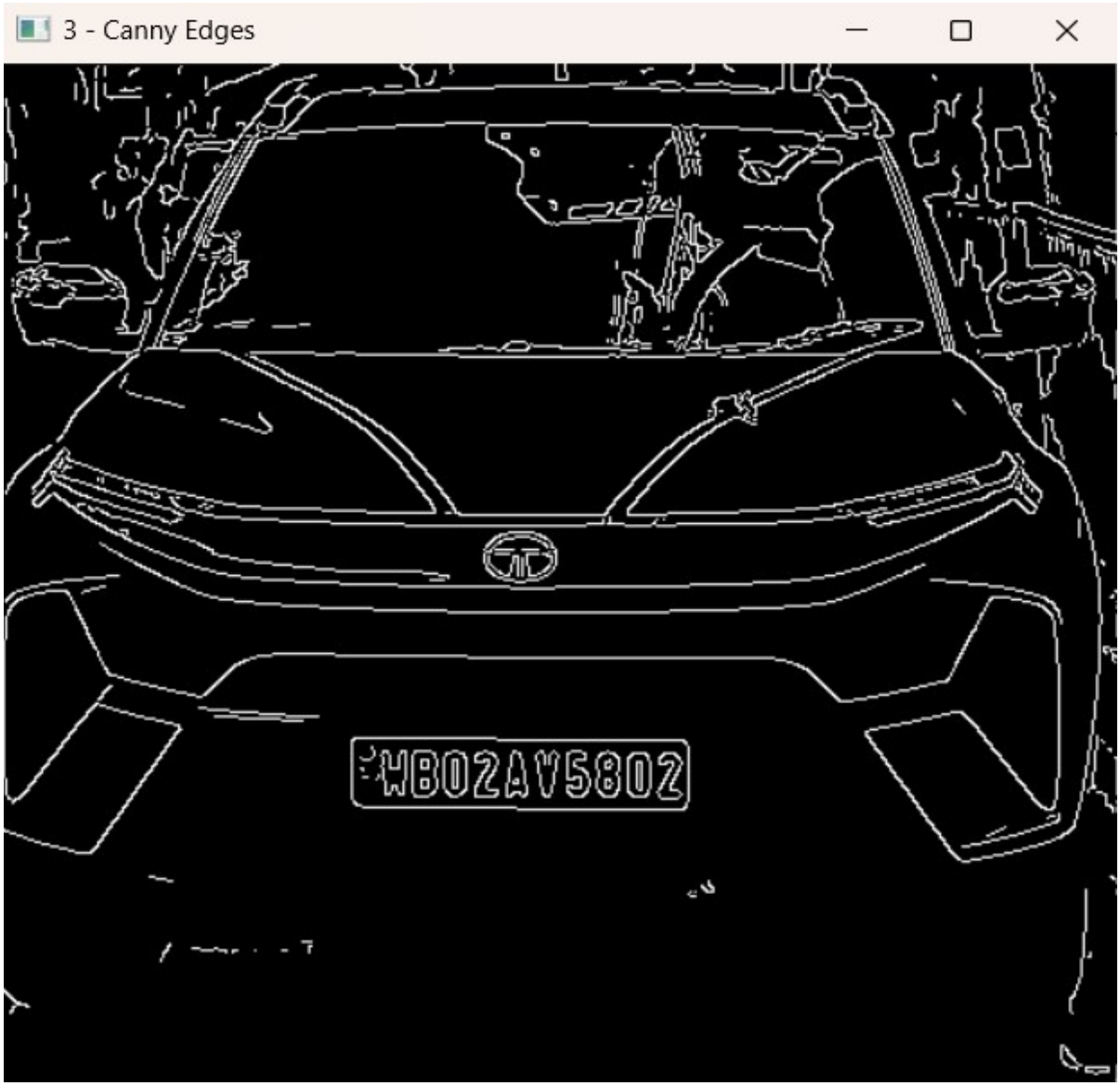

Figure 5: Applying Canny to find edges of a given image

In this Figure 5 shows the edges detected in the filtered image using the Canny edge detection algorithm highlighting potential boundaries in the image.

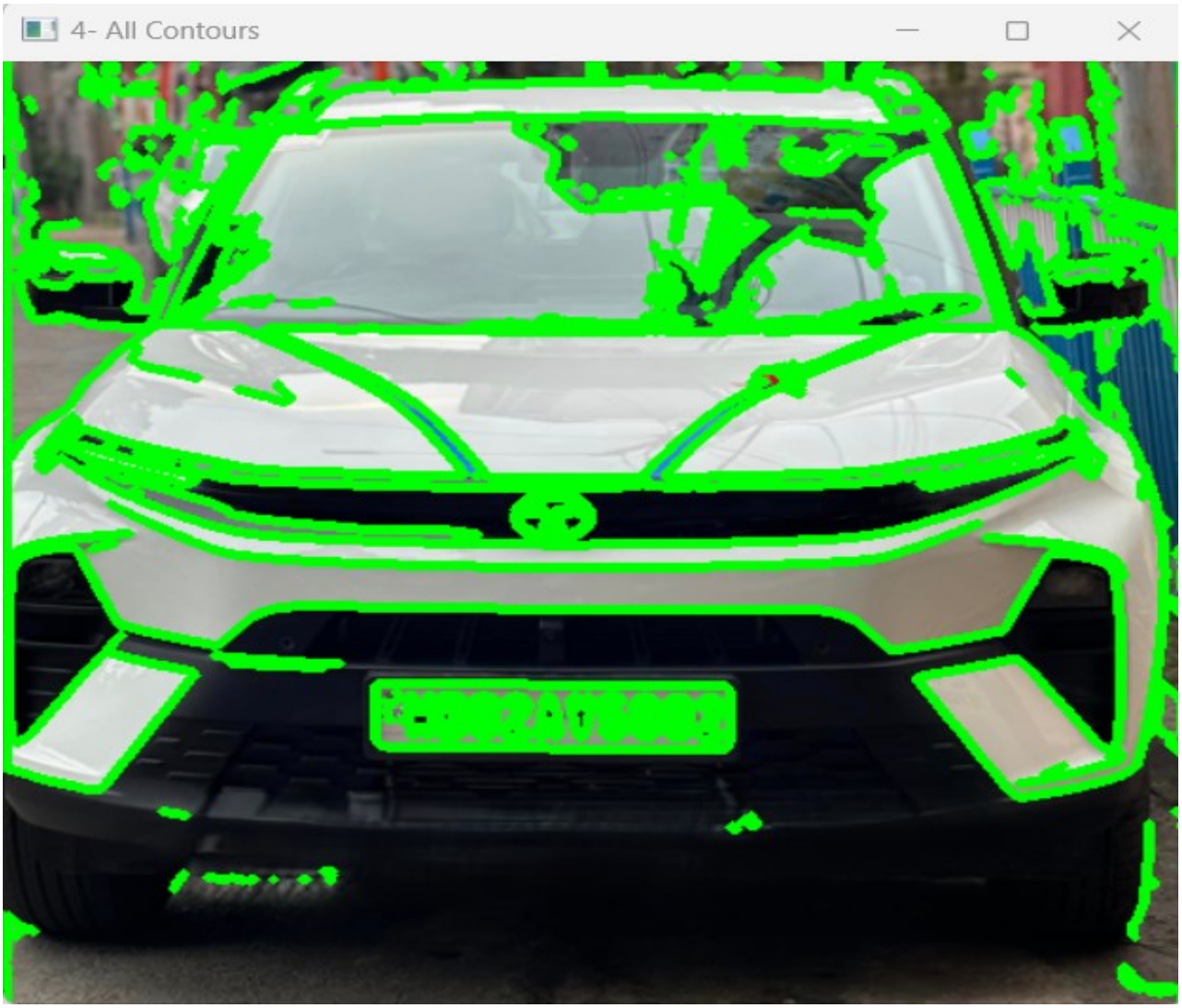

Figure 6: Detecting all contours of a given image

In this Figure 6 the image with all detected contours outlined showing various shapes and edges, including those around the vehicle and potential number plate.

Figure 7: Detecting Top 30 contours of a given image

In this Figure 7 the image filtered down to the top 30 contours which likely focuses on the more prominent edges possibly including the outline of the number plate.

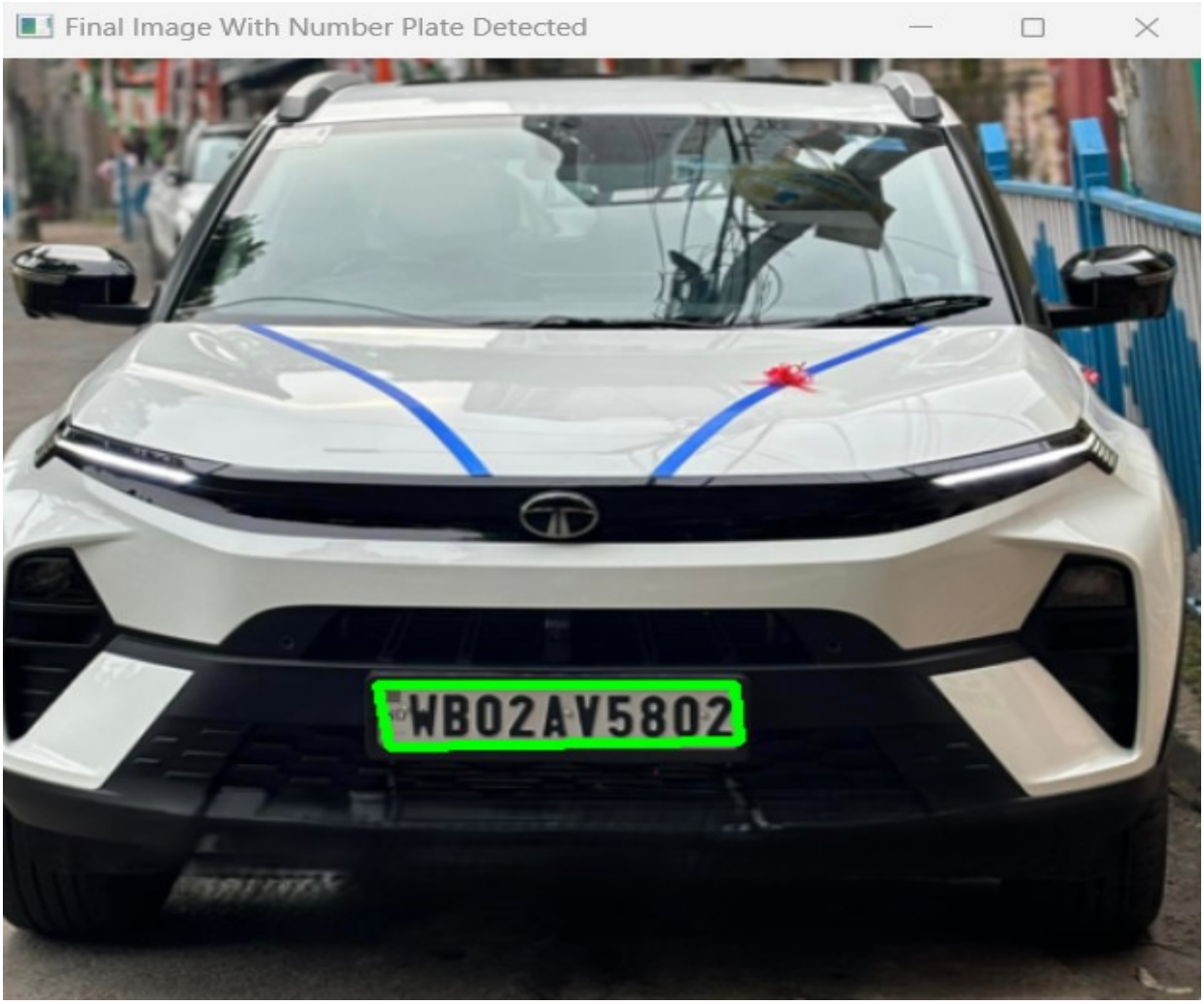

Figure 8: Detecting number plate of a given image

In this Figure 8 image highlights the detected number plate by further refining contour detection isolating the plate region on the vehicle.

Figure 9: Cropped Image of a given image

In this Figure 9 the final cropped image of the detected number plate showing only the plate area for further analysis such as OCR.

Table 1: License plate Character Detection Accuracy Table

Vehicle	Number Plate	Extraction Data	Character Accuracy (%)
Car 1	WB02AV5802	WBO2AV5802	90.0
Car 2	LR33 TEE	JLR33 TEE	
Car 3	DL7C N 5617	L7C N 5617	88.8
Car 4	MH01AV8866	MHO1AV8866	90.0
Car 5	KA 53 N 6494	KA 53 N 6494	100.0
Car 6	KL 65 H 4383	KL 65 H 4383	100.0
Car 7	CH01AN 0001	CHO1ANO001	87.5
Car 8	MH 12 NE 8922	MH 12 NE	61.5
Car 9	15-LK-10898	15-LK-10898	100.0
Car 10	MK-35-32	MK-35-32	100.0
Car 11	HR 26 BR 9044	IHR 26 BR 9044	87.5
Car 12	MH 20 EE 7598	MH 20 EE 7598	100.0
Car 13	JE- 9200	JE-9200	100.0
Car 14	KA 64 N 0099	KN 64 N G99	66.6
Car 15	MH01BG2654	MHO1BG2654	90.0
Car 16	KA02MP 9657	KA G2MP 9657	90.0
Car 17	GJ03ER0563	JO3ER0563	90.0
Car 18	MH 14F 0911	WHI4FO911	77.7
Car 19	MH01AV8866	MHO1AV8866	90.0
Car 20	HR26 BP 3543	HR26 BP3543	100.0
Car 21	MH 20 DV 2363	wH20 0V.2363	77.7
Car 22	MH 12 DE 1433	WH120E1433	77.7
Car 23	MH 14 BN 7077	NUMBER PLATE NOT DETECT	0.0
Car 24	MH 14BR 6899	MH 14BR 6889	90.0
Car 25	KL 59 T 997	NUMBER PLATE NOT DETECT	0.0
Car 26	AP05 BL 6339	P05 pLo339	
Car 27	WB 22 U 3481	NUMBER PLATE NOT DETECT	0.0
Car 28	MH 15BD8877	MH15B08677	80.0
Car 29	TN 09 BY 9726	TNO9 BY 9726	100.0
Car 30	WB 06 M 1162	WB OGM 1162	88.8

“Table 1” displays information on the results of a license plate recognition process. Each row represents a different car's number plate and includes details on the extracted plate data and the accuracy of the OCR in detecting the characters accurately.

From above table we stated that out of 30 images we get 8 images accurate result

1. Vehicle: Label indicating each car sequentially (e.g., "Car 1," "Car 2," etc.).

2. Number Plate: The actual license plate number for each vehicle, formatted with spaces or hyphens as typically found on number plates in different regions.

3. Extraction Data: The license plate characters as detected by the OCR tool. In some cases, characters are misinterpreted or partially missing, which affects accuracy.

4. Character Accuracy (%): A measure of how accurately the OCR recognized each license plate. It is calculated based on the similarity between the original plate and the extracted plate data.

    - Some entries are highly accurate, with 100% accuracy, meaning the extracted data matches the number plate perfectly.

    - Other entries have less than perfect accuracy, indicating that some characters were misidentified. For example, "CH01AN 0001" was extracted as "CHO1ANO001" with 87.5% accuracy.

    - In cases where the OCR failed entirely to detect a plate, the result is noted as "NUMBER PLATE NOT DETECT" with a 0.0% accuracy score.

Observations

- The system performed very well for several entries (e.g., "KA 53 N 6494," "KL 65 H 4383"), where the extracted text matched the number plate with 100% accuracy.

- Common misrecognitions involve substituting letters with numbers (e.g., "O" with "0") or missing certain characters.

- Some plates were completely undetected by the OCR, indicating a need for further tuning of the OCR system to handle a wider variety of plate formats and styles.

CHAPTER 5

5.0 Conclusions and Future Scopes

5.1 Conclusions

In the 21st century, nearly everything is digitally interconnected. Handwritten letters and printed texts are now rarely used or sent. In our daily lives, we rely heavily on computers, which make tasks easier and more convenient. Today, physical documents are increasingly digitized, allowing for electronic editing, manipulation, searching, management, storage, and, importantly, interpretation by machines.

Optical Character Recognition made it possible to convert text captured in images, handwritten or printed text into digitized and usable machine coded text.

In this research works we have developed an automated license plate recognition system utilizing OpenCV and Tesseract OCR. The methodology proved effective in detecting and extracting license plates from images, followed by accurate text recognition. Future enhancements could explore deep learning models for improved robustness under various conditions and integrating real-time video processing capabilities.

5.2 Future Scopes

Some potential future scopes for enhancing and expanding the given OCR-based license plate detection

Improving Detection Accuracy in Low-Quality Images: Many real-world images suffer from low resolution, poor lighting, or motion blur, making OCR challenging. Future improvements could involve using advanced denoising and deblurring techniques or applying machine learning models, such as convolutional neural networks, to enhancee image quality before processing.

Incorporating Deep Learning for Object Detection: Integrating deep learning frameworks (e.g., YOLO, Faster R-CNN) could provide more robust and flexible license plate detection, particularly in complex scenes or from various camera angles. These models can improve plate localization accuracy compared to traditional contour and edge detection methods

Implementing Real-Time Detection and Recognition: Future improvements could focus on optimizing the code for real-time applications, such as video-based license plate recognition for traffic monitoring. This would involve enhancing the computational efficiency of each step to enable continuous, fast processing of video frames.

Enhancing OCR Accuracy with Character-Level Verification: Integrating character verification techniques, such as comparison with detected text with license plate databases or using post-processing methods to correct OCR errors, could significantly improve recognition accuracy.

Optimizing for Different Environmental Conditions: Additional preprocessing methods could be introduced to handle environmental variations, such as reflections, shadows, and rain, which affect image clarity. Adaptive thresholding, contrast adjustment, or brightness normalization techniques could further improve performance under challenging conditions.

References

[1] Milan, Samantaray, et al. “Optical Character Recognition (OCR) based Vehicle's License Plate Recognition System Using Python and OpenCV”, 5th International Conference on Electronics, Communication and Aerospace Technology (ICECA), IEEE, 2021.

[2] U, Salimah, et al. “Automatic License Plate Recognition Using Optical Character Recognition”, IOP Publishing, IOP Conf. Series: Materials Science and Engineering, 2023.

[3] M.L.S.N.S, Lakshmi, et al. “License Plate Detection using Optical Character Recognition”, International Journal of Applied Engineering Research, Vol. 5, No.2, 2020.

[4] Shivani, Bansal, et al. “A Necessary Review on Optical Character Recognition (OCR) System for Vehicular Applications”, Second International Conference on Inventive Research in Computing Applications (ICIRCA), IEEE, 2020

[5] Prakhar, Sisodia, Syed Wajahat Abbas, Rizvi, “Optical Character Recognition Development Using Python”, Journal of Informatics Electrical and Electronics Engineering (JIEEE), 4(3) pp. 1-13, 2023.

[6] Mohammed, AS, Shariff, et al. “Vehicle Number Plate Detection Using Python and Open CV”, International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), IEEE, 2021.

[7] Manpreet, Kaur, et al. “Optical Character Recognition for Vehicle Number Plates Detection with Even, Odd Identification and State Detection”, International Journal of Allied Practice Research and Review (IJAPRR),. Vol. IV, Issue IV, pp. 01-08, April, 2017.

[8] Yash, Shambharkar, et al. “An Automatic Framework for Number Plate Detection using OCR and Deep Learning Approach”, International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 14, No. 4, 2023.

[9] Nithin K., Shine, Gariman, Bhutani, Tamatapu, Sai Keerthana, and G., Rohith, “An approach for improving Optical Character Recognition using

Contrast enhancement technique", Journal of Physics: Conference Series, Vol. 2466, No. 012009, 2023.

[10] Nita M., Thakare, et al. "Automatic Vehicle Number Plate Detection System", International Journal of Scientific Research in Engineering and Management (IJSREM), Vol. 07, No. 03, March 2023.

Appendix

```
import numpy as np
import cv2
import imutils
import pytesseract
import os

# Set up pytesseract path
pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"
crpimg='Cropped Images-Text/'

def image_ocr(im):
    # Read the image file
    im=im
    image = cv2.imread(im)
    if image is None:
        print("Error: Image not found or unable to load.")
        exit()

    # Resize the image - change width to 500
    image = imutils.resize(image, width=500)

    # Display the original image
    cv2.imshow("Original Image", image)
    cv2.waitKey(0)

    # RGB to Gray scale conversion
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    cv2.imshow("1 - Grayscale Conversion", gray)
    cv2.waitKey(0)

    # Noise removal with iterative bilateral filter (removes noise while preserving edges)
    gray = cv2.bilateralFilter(gray, 11, 17, 17)
    cv2.imshow("2 - Bilateral Filter", gray)
    cv2.waitKey(0)

    # Find Edges of the grayscale image
    edged = cv2.Canny(gray, 170, 200)
    cv2.imshow("3 - Canny Edges", edged)
    cv2.waitKey(0)
```

```
    # Find contours based on edges
    cnts, _ = cv2.findContours(edged.copy(), cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)

    # Create copy of original image to draw all contours
    img1 = image.copy()
    cv2.drawContours(img1, cnts, -1, (0,255,0), 3)
    cv2.imshow("4- All Contours", img1)
    cv2.waitKey(0)

    # Sort contours based on their area, keeping the top 30 contours
    cnts = sorted(cnts, key=cv2.contourArea, reverse=True)[:30]
    NumberPlateCnt = None  # We currently have no Number plate contour

    # Top 30 contours
    img2 = image.copy()
    cv2.drawContours(img2, cnts, -1, (0,255,0), 3)
    cv2.imshow("5- Top 30 Contours", img2)
    cv2.waitKey(0)

    # Create folder for cropped images if it doesn't exist
    os.makedirs('Cropped Images-Text', exist_ok=True)

    # Loop over contours to find the best approximate contour of the number
plate
    count = 0
    idx = 1
    for c in cnts:
        peri = cv2.arcLength(c, True)
        approx = cv2.approxPolyDP(c, 0.02 * peri, True)

        if len(approx) == 4:  # Select the contour with 4 corners
            NumberPlateCnt = approx  # This is our approx Number Plate Contour

            # Crop the contour and store it in Cropped Images folder
            x, y, w, h = cv2.boundingRect(c)  # Get coordinates for plate
            new_img = gray[y:y + h, x:x + w]  # Create new image
    #         cv2.imwrite(f'Cropped Images-Text/{idx}.png', new_img)  # Store
new image
            cv2.imwrite(crpimg+im, new_img)
            idx += 1
            break
```

```
    # Drawing the selected contour on the original image
    if NumberPlateCnt is not None:
        cv2.drawContours(image, [NumberPlateCnt], -1, (0, 255, 0), 3)
        cv2.imshow("Final Image With Number Plate Detected", image)
        cv2.waitKey(0)
    else:
        print("Error: No contour detected that meets the criteria.")
        exit()

    # Load the cropped image for display and OCR processing
    ##
    # Cropped_img_loc = 'Cropped Images-Text/1.png'
    # Cropped_img_loc
    cropped_image = cv2.imread(crpimg+im)
    ##
    if cropped_image is None:
        print("Error: Cropped image not found.")
        exit()
    else:
        cv2.imshow("Cropped Image", cropped_image)
        cv2.waitKey(0)
        # Use tesseract to convert image into string
        text = pytesseract.image_to_string(crpimg+im, lang='eng')
        print("Number is:", text)

    cv2.waitKey(0)  # Wait for user input before closing displayed images
    cv2.destroyAllWindows()
    return
directory_path = os.getcwd()  # Get current directory
image_files = [f for f in os.listdir(directory_path) if f.endswith(('.png', '.jpg',
'.jpeg'))]
for im in image_files:
    print(im)
    image_ocr(im)
```