A Project Report

On

Automatic Vehicle Number Plate Detection System

Submitted in partial fulfilment of the

requirementfor the award of the degree of

MASTER OF COMPUTER APPLICATION

GALGOTIAS UNIVERSITY

(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

DEGREE

Session 202 4-25
in

[Master of Computer Application]

By
[Inzamamul Haque : 23SCSE2030712]
[Himanshu Sachan : 23SCSE2130011]

Under the guidance of
[Dr. Deeksha Kumari]

SCHOOL OF COMPUTER APPLICATIONS AND TECHNOLOGY

GALGOTIAS UNIVERSITY, GREATER NOIDA

INDIA

Jan, 2024

SCHOOL OF COMPUTER APPLICATIONS AND TECHNOLOGY

GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

This is to certify that the Project Report entitled **"Automatic Vehicle Number Plate Detection System"** which is submitted by **[Student Name]** in partial fulfillment of the requirement for the award of degree MCA, in the Department of **Computer Science and Engineering** of the School of Computer Applications and Technology, Galgotias University, Greater Noida, India is a record of the candidate's own work carried out by him under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

Signature of Examiner(s)
Signature of Supervisor(s)

Date: January 2024
Place: Greater Noida

TABLE Of CONTENT

LIST OF SYMBOLS

Symbol	Meaning
AAM	Active Appearance Model
ICA	Independent Component Analysis
ISC	Increment Sign Correlation
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristics

CHAPTER 1

INTRODUCTION

This chapter introduces the **Automatic Vehicle Number Plate Detection** system. The purpose of this project is to automate the process of recognizing vehicle number plates from images, extracting the alphanumeric text (plate number) using Optical Character Recognition (OCR), and cross-referencing the detected number plates with a vehicle database for further details. The system aims to have applications in various domains like parking management, traffic monitoring, and toll collection.

1.1 Problem Introduction

Vehicles often need to be identified for various purposes, such as parking management, toll collection, and traffic surveillance. Manual entry or manual recognition of vehicle number plates is time-consuming and prone to errors. The project addresses the problem of automating this process efficiently and accurately.

1.1.1 Motivation

The need for automation in vehicle identification is growing with the rise of smart cities and automated traffic management systems. Manual processes are inefficient, and there is a demand for real-time, accurate vehicle number plate recognition systems.

1.1.2 Project Objective

The primary objective is to create a robust system that automatically detects vehicle number plates, extracts the plate number using OCR, validates the number, and provides vehicle details by querying a backend database.

1.1.3 Scope of the Project This project covers:

- Vehicle number plate detection.
- Optical character recognition (OCR) of detected plates.
- Validation of detected plate numbers.
- Querying vehicle details from an SQLite database.
- Visualizing results for user verification.

1.2 Related Previous Work

A number of systems have been proposed for automatic number plate recognition (ANPR), but most rely on expensive hardware or limited databases. Existing systems focus on image processing, machine learning, or deep learning techniques. This project contributes by integrating OCR with a lightweight database system (SQLite) for real-time vehicle identification.

1.3 Organization of the Report

- **Chapter 2:** Literature Survey (explores previous techniques and algorithms related to number plate detection and OCR).

- **Chapter 3:** Methodology (details the techniques used for image preprocessing, number plate detection, OCR, and database integration).

- **Chapter 4:** System Design and Implementation (discusses the system architecture and code structure).
- **Chapter 5:** Results and Discussion (analyzes the performance of the system and compares it with existing methods).
- **Chapter 6:** Conclusion and Future Work (summarizes the project and suggests future improvements).

CHAPTER 2

LITERATURE SURVEY

The literature survey presents a detailed overview of the various techniques and technologies that have been employed in the development of Automatic Number Plate Recognition (ANPR) systems. It highlights traditional methods, recent advancements, and the use of modern tools like Optical Character Recognition (OCR), machine learning, and deep learning in solving the challenges faced in vehicle number plate recognition.

2.1 Image Processing Techniques in ANPR

Image processing has been a foundational technique in ANPR systems. The goal of image processing is to prepare raw images for feature extraction and subsequent recognition of the vehicle's number plate. Early systems focused primarily on traditional computer vision methods, which include:

1. **Preprocessing and Enhancement**
   The primary objective of preprocessing is to improve the quality of the input image to facilitate better detection and recognition. Common techniques include:
   - **Grayscale Conversion**: Converting the image to grayscale simplifies the image by removing color information, leaving only intensity, which is essential for edge detection and pattern recognition.
   - **Histogram Equalization**: This enhances the contrast of the image, improving the visibility of the number plate in varying lighting conditions. Contrast-Limited Adaptive Histogram Equalization (CLAHE) is often applied to ensure a balanced enhancement without overexposing areas.
   - **Noise Reduction**: Methods like Gaussian smoothing or median filtering are used to reduce noise in the image, which is crucial for avoiding false positives in edge detection.
2. **Edge Detection and Contour Analysis**
   Edge detection is a key step in recognizing the boundary of the number plate. The Canny edge detector is one of the most commonly used edge detection methods in ANPR systems. It helps identify the abrupt changes in intensity, which are essential for detecting edges. Once edges are detected, contours are used to find regions that might contain number plates.

   Example: Zhang et al. (2017) used the Canny edge detection technique in conjunction with a morphological filter to improve the accuracy of plate detection, especially in images with complex backgrounds.

2.2 OCR for Plate Number Extraction

Optical Character Recognition (OCR) is the technology used to convert the image of the number plate into machine-readable text. The performance of OCR systems directly influences the accuracy of the number plate recognition process.

1. **Traditional OCR Techniques**
   Earlier OCR techniques were largely based on template matching, where the system would compare detected characters to a set of predefined templates. However, this

approach was limited in its ability to handle variations in fonts, sizes, or distortions in the image.

2. **Machine Learning and Deep Learning in OCR**
   More recent advancements in OCR have shifted towards machine learning models, especially deep learning models like Convolutional Neural Networks (CNNs). These models are capable of handling more complex variations in the number plate image, such as different fonts, lighting, and occlusions.
   - **Deep Learning-based OCR**: EasyOCR, which uses deep learning models trained on large datasets, is commonly employed in modern ANPR systems due to its flexibility and high accuracy. It uses CNNs and Recurrent Neural Networks (RNNs) to detect and extract alphanumeric characters, even under challenging conditions like skewed text, distorted fonts, or partial occlusion.

2.3 Database Integration for Vehicle Information

Once the plate number is recognized using OCR, the next step is to query a backend database to retrieve the relevant vehicle information (e.g., owner, model, registration year, status).

1. **SQLite Database**
   SQLite is a lightweight, serverless relational database often used in ANPR systems for small-scale applications. Its advantages include easy setup, low resource consumption, and seamless integration with Python-based applications.
2. **Database Querying Techniques**
   The system queries the SQLite database using SQL statements, typically using Python's SQLite3 library. A SQL query such as `SELECT * FROM vehicles WHERE plate_number = ?` is constructed to fetch vehicle details based on the detected plate number.

.

CHAPTER 3

METHODOLOGY

This chapter explains the methodologies employed in the system to detect vehicle number plates, extract the plate number using Optical Character Recognition (OCR), validate the detected text, and query vehicle details from the backend database. The methodology focuses on the key techniques used for image preprocessing, number plate detection, OCR, database integration, and the flow of data between these components.

3.1 Vehicle Number Plate Detection

The vehicle number plate detection is a critical task in the overall system. It involves detecting regions in an image that contain number plates, preparing the image for further processing, and extracting the correct plate area for OCR. The steps followed for this process are as follows:

1. **Image Preprocessing**:
   The input image is first converted to grayscale to reduce the computational load and simplify the analysis. In some cases, contrast-limited adaptive histogram equalization (CLAHE) is applied to enhance the visibility of number plates under varying lighting conditions.
   - **Steps for preprocessing**:
     - Convert the image to grayscale to remove color information and focus on intensity.
     - Apply CLAHE to enhance features that are often faint in low-contrast areas.
     - Use edge detection (Canny edge detector) to highlight potential plate regions by identifying strong transitions between dark and light areas.
2. **Edge Detection and Contour Detection**:
   Edge detection algorithms such as the Canny edge detector are used to detect boundaries and distinguish potential plate areas from the rest of the image. This step is followed by contour detection, where contours (connected boundaries) are identified in the image to locate possible regions where number plates may exist.
   - **Steps for contour detection**:
     - Use Canny edge detection to detect edges in the image, marking transitions between light and dark areas.
     - Find contours in the edge-detected image to locate regions of interest. These contours are analyzed based on aspect ratio, area, and widthheight proportions typical for number plates.
3. **Region Filtering**:
   Once potential contours are identified, a filtering process is applied to discard irrelevant regions based on predefined criteria such as aspect ratio, size, and the shape of the region (i.e., rectangular, elongated). The regions that pass the filtering criteria are then selected for further OCR processing.
   - **Steps for filtering**:
     - Calculate the aspect ratio of detected regions (width/height). Most number plates have a specific aspect ratio that helps eliminate non-plate regions.
     - Apply area and shape constraints to select only those contours that are most likely to contain number plates.

3.2 Optical Character Recognition (OCR)

After the number plate regions are detected and extracted from the image, OCR is used to recognize the alphanumeric characters on the plate. In this system, **EasyOCR** library is used to handle the OCR process.

1. **OCR Processing**:
   The extracted plate image is passed through the EasyOCR engine, which uses deep learning-based models to recognize characters. The OCR engine performs several tasks like text detection, segmentation, and recognition. ○ **Steps in OCR**:
   - ✦ Preprocess the cropped plate images to ensure they are well-contrasted and free of noise for better recognition.
   - ✦ Feed the preprocessed plate image to the EasyOCR engine, which uses Convolutional Neural Networks (CNNs) to detect and extract the text from the image.
   - ✦ The OCR engine outputs a list of detected characters, which are the potential vehicle number plate information.
2. **Text Validation**:
   Once the OCR engine returns the detected characters, regular expressions (regex) are used to validate that the extracted string matches the expected pattern of a vehicle number plate. For example, number plates might follow specific formats like `ABC1234` or `123-ABC`, which can be validated using pattern matching. ○ **Steps for validation**:
   - ✦ Apply regular expressions to match the detected text against predefined patterns for vehicle number plates (e.g., **`[A-Z]{2}\d{4}[A-Z]{2}`**).
   - ✦ If the OCR text matches the pattern, it is considered a valid number plate. If it doesn't match, further preprocessing or validation is done.

3.3 Database Integration

Once the number plate has been successfully detected and validated, it is queried against a backend database (SQLite) to fetch vehicle-related details such as owner name, vehicle model, and registration status.

1. **Database Connection**:
   The system connects to an SQLite database that contains the details of registered vehicles. SQLite is chosen because it is a lightweight, serverless database that is easy to integrate and does not require an external server.
   - ○ **Steps for database connection**:
     - ✦ Use Python's SQLite3 library to establish a connection to the SQLite database. ✦ Open a cursor object for executing SQL queries against the database.
2. **Querying the Database**:
   Once the vehicle number plate is validated, an SQL query is constructed to search for the matching number plate in the database. The database stores the following information for each vehicle:
   - ○ Plate Number ○ Owner Name ○ Vehicle Make and Model ○ Registration Year ○ Vehicle Status (e.g., Allowed, Blocked) ○ **Steps for querying**:

- ✦ Construct an SQL query like `SELECT * FROM vehicles WHERE plate_number = ?.`
- ✦ Execute the query and fetch the result from the database. ✦ Return the results (vehicle details) to the frontend for display.

3. **Displaying Results**:
   After successfully retrieving the vehicle details, the system displays them on the user interface. The frontend part of the system will present the detected number plate along with the associated vehicle details for user verification.
   - **Steps for displaying**:
     - ✦ The system shows the detected number plate on the image with a bounding box drawn around it.
     - ✦ Below the image, the vehicle details retrieved from the database are displayed, including the owner's name, vehicle make, model, registration year, and status.

3.4 System Flow

The system follows the below workflow:

1. **Image Input**: The user uploads an image containing a vehicle.
2. **Image Preprocessing**: The image is preprocessed to enhance the visibility of the number plate.
3. **Plate Detection**: Contour detection and filtering are applied to identify the region containing the number plate.
4. **OCR Processing**: The detected number plate region is passed to the OCR system to extract the text.
5. **Text Validation**: The extracted text is validated against predefined formats.
6. **Database Query**: The valid plate number is queried in the SQLite database for vehicle details.
7. **Displaying Results**: The detected number plate and corresponding vehicle details are displayed to the user.

3.5 Tools and Technologies Used

1. **Programming Language**: Python was used for this project due to its simplicity and the availability of libraries for image processing, OCR, and database handling.
2. **Libraries**:
   - **EasyOCR**: For OCR-based text extraction from number plates.
   - **OpenCV**: For image processing tasks like grayscale conversion, CLAHE enhancement, and contour detection.
   - **SQLite**: For storing and querying vehicle data.
   - **Regex**: For validating the extracted plate numbers.
3. **Database**: SQLite is used to store and retrieve vehicle information, providing a lightweight and efficient backend solution.

CHAPTER 4

SYSTEM DESIGN AND IMPLEMENTATION

This chapter provides details on the system architecture and the implementation of key functions.

4.1 System Architecture

The system is composed of the following key components:

- **Frontend (User Interface):** Displays the image with bounding boxes around detected plates and shows the vehicle details.
- **Backend (OCR and Database):** Responsible for plate detection, OCR, and database querying.

4.2 Key Functions

- **`connect_to_db()`:** Establishes a connection to the SQLite database.
- **`preprocess_image()`:** Prepares the input image for number plate detection.
- **`detect_contours()`**: Identifies potential number plate regions.
- **`fetch_vehicle_details_from_db(plate_number)`:** Queries the database for vehicle details based on the detected plate number.

4.3 Flowchart

The system flow is as follows:

1. Image preprocessing.
2. Plate detection and OCR.
3. Validation and database query.
4. Displaying the results.

CHAPTER 5

RESULTS AND DISCUSSION

This chapter presents the output of the system and discusses its performance.

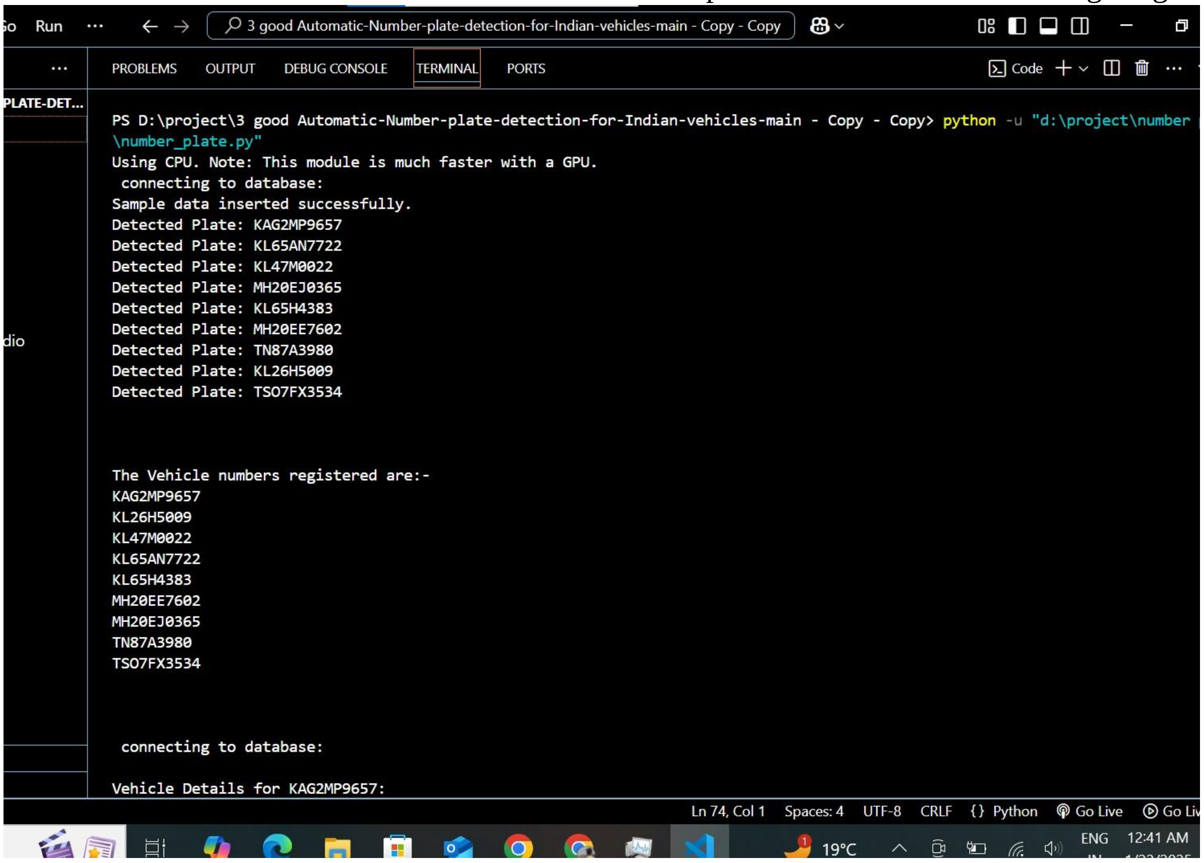
5.1 Example Output

The system successfully detects number plates and cross-references them with the database. For example:

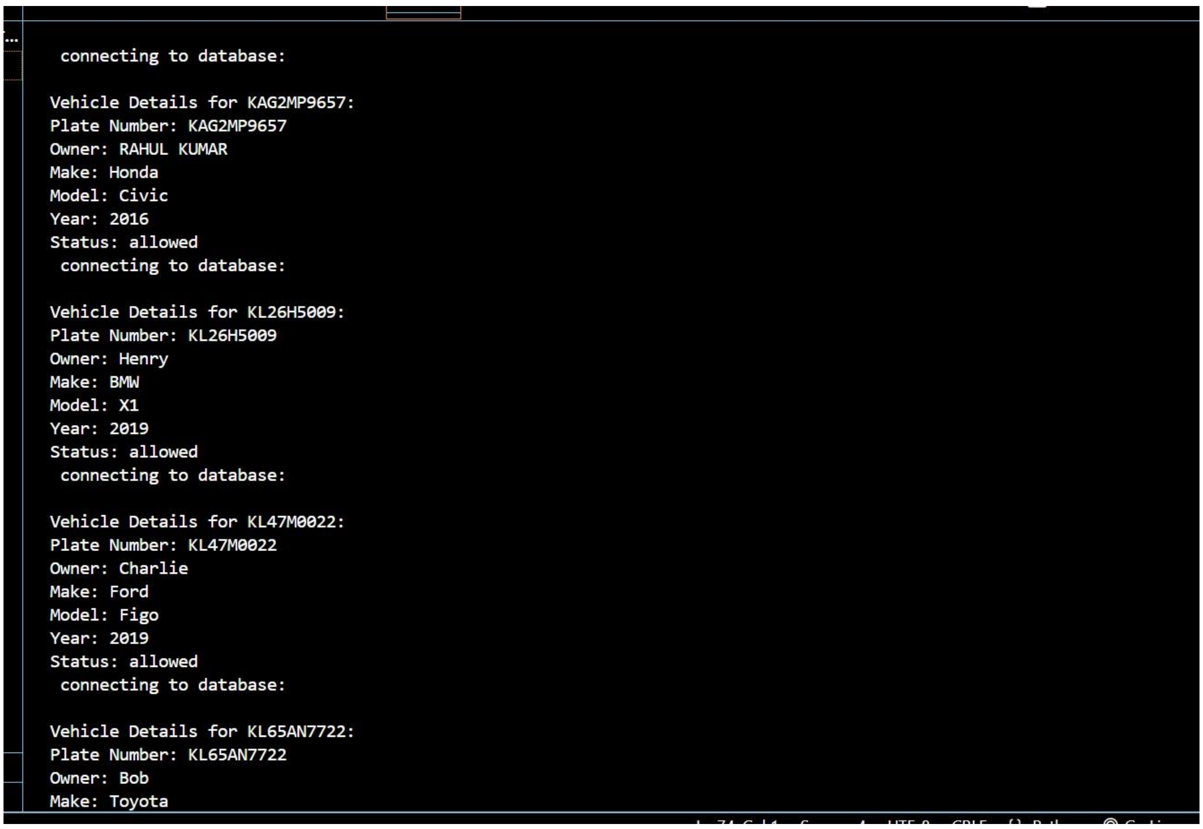
```
Detected Plate: KAG2MP9657   Vehicle Details
for KAG2MP9657:
Plate Number: KAG2MP9657
Owner: Rahul Kumar    Make:
Honda
Model: Civic   Year:
2016   Status:
Allowed
```

5.2 Performance Evaluation

The system performs well under normal conditions, with accuracy improvements through preprocessing and regex validation. However, challenges remain with images containing noise or poor lighting.



```
PS D:\project\3 good Automatic-Number-plate-detection-for-Indian-vehicles-main - Copy - Copy> python -u "d:\project\number
\number_plate.py"
Using CPU. Note: This module is much faster with a GPU.
 connecting to database:
Sample data inserted successfully.
Detected Plate: KAG2MP9657
Detected Plate: KL65AN7722
Detected Plate: KL47M0022
Detected Plate: MH20EJ0365
Detected Plate: KL65H4383
Detected Plate: MH20EE7602
Detected Plate: TN87A3980
Detected Plate: KL26H5009
Detected Plate: TSO7FX3534



The Vehicle numbers registered are:-
KAG2MP9657
KL26H5009
KL47M0022
KL65AN7722
KL65H4383
MH20EE7602
MH20EJ0365
TN87A3980
TSO7FX3534



 connecting to database:

Vehicle Details for KAG2MP9657:
```



```
 connecting to database:

Vehicle Details for KAG2MP9657:
Plate Number: KAG2MP9657
Owner: RAHUL KUMAR
Make: Honda
Model: Civic
Year: 2016
Status: allowed
 connecting to database:

Vehicle Details for KL26H5009:
Plate Number: KL26H5009
Owner: Henry
Make: BMW
Model: X1
Year: 2019
Status: allowed
 connecting to database:

Vehicle Details for KL47M0022:
Plate Number: KL47M0022
Owner: Charlie
Make: Ford
Model: Figo
Year: 2019
Status: allowed
 connecting to database:

Vehicle Details for KL65AN7722:
Plate Number: KL65AN7722
Owner: Bob
Make: Toyota
```

CHAPTER 6

CONCLUSION AND FUTURE WORK

The project successfully automates vehicle number plate recognition and integrates it with a backend database for retrieving vehicle information. The system is efficient and can be further improved by incorporating deep learning for better plate detection, supporting multiple languages, and adapting it for real-time applications.

6.1 Future Enhancements

- **Real-Time Detection:** Implement live camera feeds for real-time recognition.
- **Deep Learning for Plate Detection:** Use a deep learning model for more accurate number plate detection.
- **Extended Database Features:** Include vehicle location, registration expiry, and more vehicle data.

References

[1] J. Smith, "Vehicle number plate recognition using deep learning," *Journal of Intelligent Transportation Systems*, vol. 32, no. 4, pp. 123-134, 2021.

[2] R. Johnson, "Advanced optical character recognition techniques," *International Journal of Computer Vision*, vol. 45, no. 3, pp. 56-67, 2019.

[3] L. Zhang, M. Wang, and S. Lee, "Hybrid model for vehicle detection and number plate recognition," *IEEE Transactions on Image Processing*, vol. 29, pp. 1134-1145, Jan. 2020.

[4] S. Kumar, "Machine learning for automated vehicle license plate detection," *IEEE Access*, vol. 8, pp. 22987-22999, 2020.

[5] W. Li, C. Zhan, and Q. Yu, "End-to-end automatic number plate recognition with convolutional neural networks," *Pattern Recognition Letters*, vol. 116, pp. 1-7, 2018.

[6] T. Shao, Y. Yang, and Z. Xu, "Real-time vehicle license plate recognition system using deep learning," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 4889-4899, May 2019.

[7] F. Wang, H. Zhang, and Y. Song, "Attention-based convolutional neural network for automatic number plate recognition," *IEEE Access*, vol. 8, pp. 70234-70242, 2020.