VISVESVARAYA TECHNOLOGICAL UNIVERSITY,

BELAGAVI - 590018

Mini Project Report

On

"AUTOMATIC VEHICLE LICENCE DETECTION USING HAAR CASCADE CLASSIFIER"

A report submitted in partial fulfillment of the requirements for

COMPUTER GRAPHICS AND IMAGE PROCESSING LABORATORY (21CSL66)
In
Computer Science and Design

Submitted by

ADARSH BHAVIMANE	**4AL21CG004**
GHRUTHAVARSHA	**4AL21CG024**
RACHANA	**4AL21CG043**
RAKSHITHA	**4AL21CG045**

Under the Guidance of
Dr. Pushparani M K
Senior Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN
ALVA'S INSTITUTE OF ENGINEERING & TECHNOLOGY MIJAR,
(Unit of Alva's Education Foundation ®, Moodbidri)
Affiliated to Visvesvaraya Technological University, Belagavi,
Approved by AICTE, New Delhi, Recognized by the Government of Karnataka.
Accredited by NACC with A+ Grade
Shobavana Campus, Mijar, Moodbidri, D.K., Karnataka 2023-2024

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY MIJAR, MOODBIDRI, D.K. -574225

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

CERTIFICATE

This is to certify that the Computer Graphics and Image Processing Laboratory with Mini Project entitled **"AUTOMATIC VEHICLE LICENCE DETECTION USING HAAR CASCADE CLASSIFIER"** has been completed by

ADARSH BHAVIMANE	4AL21CG004
GHRUTHAVARSHA	4AL21CG024
RACHANA	4AL21CG043
RAKSHITHA	4AL21CG045

The Bonafide students of the Department of Computer Science and Design, Alva's Institute of Engineering and Technology in the **DEPARTMENT OFCOMPUTER SCIENCE AND DESIGN** of the VISVESVARAYA TECHNOLOGICAL UNIVERSITY,BELAGAVI during the year 2023–2024. It is certified that all corrections/suggestionsindicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The Mini Project report has been approved as it satisfies the academic requirements concerning the Mini Project work of Computer Graphics and Image Processingsubject prescribed for the Bachelor of Engineering Degree.

Dr. Pushparani M K
Mini Project Guide

Prof. Jayantkumar A Rathod
HOD, Dept of CSD

ALVA'S INSTITUTE OF ENGINEERING AND TECHNOLOGY MIJAR, MOODBIDRI D.K. -574225

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

DECLARATION

We,

ADARSH BHAVIMANE
GHRUTHAVARSHA
RAKSHITHA
RACHANA

Hereby declare that the dissertation entitled, **AUTOMATIC VEHICLE LICENCE DETECTION USING HAAR CASCADE CLASSIFIER** is completed and written by us under the supervision of our guide Dr. **Pushparani M K**, **Senior Assistant Professor,** Department of Computer Science and Design Alvas's Institute of Engineering and Technology, Moodbidri, **DEPARTMENT OF COMPUTER SCIENCE AND DESIGN** of the **VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI** during the academic year 2023-2024. The dissertation report is original and it has not been submitted for any other degree in any university.

ADARSH BHAVIMANE	4AL21CG004
GHRUTHAVARSHA	4AL21CG024
RACHANA	4AL21CG043
RAKSHITHA	4AL21CG045

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany a successful completion of any task would be incomplete without the mention of the people who made it possible, success is the epitome of handwork and perseverance, but steadfast of all is encouraging guidance.

So, with gratitude, we acknowledge all those whose guidance and encouragement served as a beacon of light and crowned the effort with success.

The selection of this mini-project work as well as the timely completion is mainly due tothe interest and persuasion of our mini-project guide **Dr. Pushparani M K,** Senior Assistant Professor, Department of Computer Science and Design. We will remember her contributionforever.

We sincerely thank, **Prof. Jayanthkumar A Rathod**, Head of the Department of Computer Science and Design who has been the constant driving force behind the completion of the project.

We thank our beloved Principal, **Dr. Peter Fernandes,** for his constant help and support throughout.

We are indebted to the **Management of Alva's Institute of Engineering and Technology, Mijar, Moodbidri** for providing an environment that helped us in completing our mini project.

Also, we thank all the teaching and non-teaching staff of the Department of Computer Scienceand Design for the help rendered.

ADARSH BHAVIMANE 4AL21CG004

GHRUTHAVARSHA 4AL21CG024

RACHANA 4AL21CG043

RAKSHITHA 4AL21CG045

ABSTRACT

The rapid advancements in computer vision and optical character recognition (OCR) technologies have significantly impacted the transportation and security industries. This project aims to develop an automated vehicle license plate recognition system to address the inefficiencies and inaccuracies of manual data recording methods. Utilizing OpenCV for image and video processing and Tesseract OCR for text recognition, the system ensures high accuracy and reliability in real-time applications.

The system captures video frames from pre-recorded files or live webcam feeds and processes them using a Haar Cascade Classifier trained specifically for license plate detection. Detected license plate regions are extracted and preprocessed for OCR, where the Tesseract engine accurately reads the text. This automated process supports applications such as traffic monitoring, automated toll collection, and vehicle tracking, significantly reducing the need for manual intervention and minimizing errors.

The system's real-time processing capability ensures efficient operation in dynamic environments, making it suitable for various practical scenarios. This project demonstrates the effective integration of advanced computer vision and OCR technologies to create a robust license plate recognition system. The results highlight significant improvements in operational efficiency and accuracy. Future enhancements will focus on improving adaptability to different license plate formats and integrating the system with broader surveillance and traffic management infrastructures, contributing to smarter and safer urban environments.

TABLE OF CONTENTS

INTRODUCTION

In recent years, the rapid advancements in computer vision and optical character recognition (OCR) technologies have transformed numerous industries, including transportation and security. One notable application is the automatic recognition of vehicle licence plates, which has significant implications for traffic management, law enforcement, and automated toll systems. Traditional methods of manually recording licence plate numbers are not only time-consuming but also prone to human error. This inefficiency can lead to inaccuracies in data collection and delays in processing.

To address these challenges, the development of automated licence plate recognition systems has become essential. These systems provide a fast, reliable, and consistent means of capturing and interpreting licence plate information from vehicles in real-time. This project focuses on creating a robust licence plate recognition system utilizing state-of-the-art computer vision techniques and OCR technology. Specifically, the project employs OpenCV, a powerful computer vision library, for image and video processing, and Tesseract OCR, an open-source text recognition engine, to accurately read the licence plates.

The primary objective of this project is to detect and read vehicle licence plates from video inputs, which can be sourced from either a pre-recorded video file or a live webcam feed. The system aims to enhance the efficiency and accuracy of licence plate recognition, which is critical for applications such as traffic monitoring, automated toll collection, and vehicle tracking. By integrating these technologies, the system can automatically identify and read licence plates, thereby minimizing the need for manual intervention and reducing the likelihood of errors.

The workflow of the system begins with capturing video frames from the specified source. These frames are then processed using a Haar Cascade Classifier trained specifically for licence plate detection. Once a licence plate is detected within a frame, the region containing the plate is extracted and preprocessed for OCR. The Tesseract OCR engine is then utilized to interpret and extract the text from the licence plate. This text can be used for various applications, such as logging vehicle entries and exits, issuing automated toll charges, or flagging vehicles of interest for security purposes.

The system's real-time processing capability ensures that it can operate efficiently in dynamic environments, making it suitable for deployment in various practical scenarios. Additionally, the use of OpenCV and Tesseract OCR ensures that the system remains both powerful and flexible, capable of handling different types of video inputs and varying environmental conditions.

This project demonstrates the effective integration of computer vision and OCR technologies to create an automated licence plate recognition system. The following sections will provide a detailed overview of the project's methodology, implementation, results, and potential applications, highlighting the system's capability to enhance operational efficiency and accuracy in licence plate recognition.

1.1 PROBLEM STATEMENT

The traditional methods of manually recording and processing licence plate numbers are inefficient, error-prone, and often result in delays. This manual process hinders applications such as traffic monitoring, automated toll collection, and vehicle tracking, where real-time and accurate identification of vehicles is crucial.

To address these challenges, this project aims to develop an automated licence plate recognition system using computer vision techniques. The system will be capable of detecting and reading vehicle licence plates from video inputs, sourced from either pre-recorded video files or live webcam feeds. By leveraging state-of-the-art technologies like OpenCV for image processing and a Haar Cascade Classifier for licence plate detection, the system aims to achieve the following objectives:

1. **Real-time Recognition**: Provide fast and accurate recognition of licence plates in dynamic environments, ensuring swift data processing.
2. **Accuracy and Reliability**: Minimize errors associated with manual data entry by automating the identification and recording of licence plate numbers.
3. **Versatility**: Support different types of vehicles and licence plate formats commonly used across various regions.
4. **User Interface**: Incorporate a user-friendly interface to display detected licence plates and their associated information, enhancing usability for operators and users.

By developing and implementing this automated system, the project seeks to improve operational efficiency, enhance data accuracy, and facilitate seamless integration into existing transportation and security infrastructures.

1.2 WHAT IS OpenCV

OpenCV (Open Source Computer Vision Library) is a powerful open-source software library designed for computer vision and machine learning applications. Originally developed by Intel, it now has a large community and is maintained by OpenCV.org. OpenCV provides a comprehensive suite of tools and algorithms to enable real-time image and video processing, making it indispensable in various fields such as robotics, healthcare, automotive, and more.

Key Features of OpenCV:

1. **Image and Video Processing**:
    - **Capture**: Ability to capture live video streams from cameras or video files.
    - **Processing**: Comprehensive set of image processing functions for filtering, transformation, and enhancement.
    - **Analysis**: Tools for analyzing images and extracting useful information such as edges, contours, and keypoints.
2. **Object Detection and Recognition**:
    - **Haar Cascades**: Implementation of Haar feature-based cascade classifiers for object detection, such as faces, eyes, and licence plates.
    - **Machine Learning**: Integration with machine learning algorithms for tasks like object classification and recognition.
3. **Deep Learning Integration**:
    - **Neural Networks**: Support for deep learning frameworks like TensorFlow, PyTorch, and Caffe, enabling integration of deep neural networks for tasks such as image classification, object detection, and segmentation.
4. **Computer Vision Algorithms**:
    - **Feature Detection and Matching**: Algorithms for detecting and matching features between images, essential for tasks like image stitching and 3D reconstruction.
    - **Optical Flow**: Techniques for estimating motion between frames, useful in video analysis and tracking.
5. **Graphical User Interfaces (GUI)**:
    - **Highgui Module**: Simple interfaces to create windows, display images and videos, and capture user input (mouse clicks, keystrokes) for interactive applications.
6. **Performance Optimization**:

- **Multithreading**: Support for parallel processing to enhance performance, critical for real-time applications.
- **Hardware Acceleration**: Utilization of hardware acceleration (e.g., through Intel's Integrated Performance Primitives, IPP) for optimized execution on CPUs and GPUs.

7. **Cross-Platform Support**:
   - **Platforms**: OpenCV is compatible with Windows, Linux, macOS, Android, and iOS, making it versatile for both desktop and mobile applications.

1.3 WHAT IS GUI DEVELOPMENT

GUI (Graphical User Interface) development involves creating interfaces that enable users to interact with software applications through graphical elements like buttons, menus, text fields, and icons, instead of text-based commands. This type of development focuses on enhancing user experience (UX) by making software more intuitive and accessible.

Key concepts in GUI development include the layout and design of user interface (UI) components, which comprise the visual elements that users interact with, and event-driven programming, where the application responds to user actions such as clicks and keystrokes. Several tools and libraries facilitate GUI development across different programming languages.

For instance, Tkinter is a standard GUI library for Python, providing a straightforward way to create desktop applications. PyQt and Kivy are other Python libraries offering more advanced features and support for multitouch applications, respectively. Java developers might use JavaFX or Swing for rich internet applications and customizable components. Electron, using web technologies like HTML, CSS, and JavaScript, is popular for building cross-platform desktop apps. The process of GUI development generally involves planning and designing the interface, selecting the appropriate tools, building the UI components, implementing event handling to manage user interactions, testing and debugging the application, and refining the interface based on feedback and testing results. By focusing on usability and accessibility, GUI development aims to create software that is not only functional but also user-friendly and visually appealing.

2. METHODOLOGY

The methodology for Automatic Vehicle Registration using image processing involves several keysteps:

2.1 Video Capture and Initialization:

- The methodology begins with initializing the video capture using OpenCV's VideoCapture module, which accesses a specified video file ("./video.mp4"). This step ensures that frames from the video are Continuously read for further processing.

2.2 Licence Plate Detection:

- Licence plate detection is performed using a pre-trained Haar Cascade classifier ('./indian_licence_plate.xml') provided by OpenCV. This classifier is capable of identifying regions in the video frames that potentially contain licence plates based on specific patterns and features.

2.3 Text Recognition Using Tesseract OCR:

- After identifying a region of interest (ROI) containing a licence plate, Tesseract OCR (pytesseract.image_to_string) is employed to extract and recognize the alphanumeric characters on the plate. This step facilitates automated reading and retrieval of licence plate numbers from the captured images.

2.4 Implementation Flow:

- Video Capture: Frames are continuously read from the specified video file ("./video.mp4").
- Licence Plate Detection: The Haar Cascade classifier (licence) identifies potential licence plate regions within each frame.
- ROI Extraction: Detected licence plate regions (plate_section) are extracted from the frames for further analysis.
- Text Recognition: Tesseract OCR is used to recognize and extract alphanumeric characters from the licence plate regions.
- Display and User Interaction: Original frames with overlaid licence plate detections are displayed for user visualization, with options to terminate the process using the 'q' key.

3. IMPLEMENTATION

3.1 Libraries and Modules

These libraries and modules are integral to the development of the hand fracture detection system, providing essential functionalities that enable image processing, GUI creation, and object detection.

- **tkinter:**
  - **Purpose:** Used for creating the graphical user interface (GUI).
  - **Details:** Tkinter is a standard GUI library in Python, known for its simplicity and ease of use. It provides a robust framework to develop desktop applications with various widgets like buttons, labels, text fields, and more. In this project, Tkinter is used to create the main application window, buttons for image upload, and areas to display results. Its event-driven architecture allows for interactive user interfaces where specific functions are triggered by user actions, such as button clicks.
- **filedialog:**
  - **Purpose:** Facilitates the selection of image files for upload.
  - **Details:** Filedialog is a module in Tkinter that allows users to open a file dialog box to browse and select files from their file system. In the context of an automatic vehicle registration system, a file upload interface is particularly useful for allowing users to easily upload images or videos of vehicles. This module simplifies the process of selecting and uploading files from a user's directory, making the system more accessible and straightforward. Users can navigate their directories using a familiar interface to select the required images or video files for processing
- **PIL (Pillow):**
  - **Purpose:** Handles image display within the GUI.
  - **Details:** Pillow is an advanced library for image processing in Python, which extends the capabilities of the original PIL (Python Imaging Library). It supports opening, manipulating, and saving many different image file formats. Pillow can read images from various file formats, making it versatile for handling vehicle images from different sources.

- **cv2 (OpenCV):**
    - **Purpose:** Performs image processing tasks.
    - **Details:** OpenCV (Open Source Computer Vision Library) is a comprehensive library for computer vision and image processing. It offers numerous functions for image manipulation, analysis, and transformation. OpenCV is a powerful library that offers a wide range of functions for processing and analyzing images and videos. In the context of an automatic vehicle registration system, OpenCV is used to read images and video frames, preprocess them, and apply various image processing techniques to prepare them for vehicle and licence plate detection. Functions such as resizing, grayscale conversion, filtering, and edge detection are crucial steps before passing the images to the detection model.
- **cvzone:**
    - **Purpose:** Assists in drawing rectangles and adding text to images.
    - **Details:** Cvzone is a utility library that streamlines common computer vision operations using OpenCV. It provides easy-to-use functions for drawing shapes, adding text, and other visual annotations on images. In the context of an automatic vehicle registration system, Cvzone is used to overlay detection results on images of vehicles and licence plates. For instance, when a vehicle or a licence plate is detected, Cvzone helps draw bounding boxes around the detected areas and label them with the recognized licence plate number or detection confidence scores, making the output more informative and user-friendly.
- **Matplotlib:**
    - **Purpose:** Assists in creating a wide range of static, animated, and interactive visualizations.
    - **Details:** Matplotlib is a versatile library for creating static, animated, and interactive visualizations in Python. In the context of an automatic vehicle registration system, Matplotlib is used to present data and detection results in an informative and user-friendly manner. Matplotlib can be used display the results of vehicle detection and licence plate recognition.

- **Pytesseract:**
    - **Purpose:** Optical Character Recognition tool used to extract text from images, specifically for recognizing and extracting licence plate numbers from vehicle images.
    - **Details:** pytesseract is a Python wrapper for Google's Tesseract-OCR Engine, which is widely used for extracting text from images. In the context of an automatic vehicle registration system, pytesseract plays a crucial role in recognizing and extracting licence plate numbers from images or video frames of vehicles. This is a critical step in converting visual data into actionable information that can be used for vehicle identification and registration.

3.2 Haar Cascade Classifier

- **Model Type**: Object detection model based on Haar features.
- **Training**: Haar cascades are trained using positive samples (images containing the object of interest) and negative samples (images without the object).
- **Usage**: The model is stored in an XML file (indian_licence_plate.xml), which OpenCV uses to detect objects in images.
- **Advantages**: Efficient and relatively fast for real-time detection tasks.

Tesseract OCR Engine

- **Model Type**: Optical Character Recognition (OCR) engine.
- **Training**: Tesseract is trained on a large dataset of text images. It uses LSTM (Long Short-Term Memory) networks for recognizing text.
- **Usage**: pytesseract is used to interface with the Tesseract engine from Python. It converts the detected licence plate region into a format suitable for OCR and extracts the text.
- **Advantages**: High accuracy for text recognition, supports multiple languages, and can be fine-tuned or retrained for specific use cases.

3.3 How it work?

1. Image Acquisition

- **Purpose**: Capture images or video frames containing vehicles, typically from cameras mounted at checkpoints or traffic monitoring systems.
- **Implementation**: Utilize OpenCV (cv2.VideoCapture) to access live video streams or pre-recorded video files. This provides a continuous stream of images for processing.

2. Licence Plate Detection

- **Purpose**: Identify regions in the captured images or frames that potentially contain licence plates.
- **Implementation**: Use a pre-trained Haar Cascade classifier (licence = cv2.CascadeClassifier('./indian_licence_plate.xml')) for object detection. This classifier is trained to recognize the distinctive features of licence plates based on positive and negative samples. The detectMultiScale method of OpenCV is employed to detect multiple instances of licence plate regions within each frame.

3. Region of Interest Extraction

- **Purpose**: Once a licence plate is detected, isolate the region of interest (ROI) containing the licence plate for further processing.
- **Implementation**: Extract the bounding box coordinates (x, y, w, h) provided by the Haar Cascade classifier. These coordinates define the rectangular area encompassing the detected licence plate within the image.

4. Image Preprocessing

- **Purpose**: Enhance the quality and suitability of the licence plate region for Optical Character Recognition (OCR).
- **Steps**:
    - **Convert to Grayscale**: Use OpenCV (cv2.cvtColor) to convert the RGB licence plate region to grayscale. Grayscale images simplify subsequent processing steps and reduce computational overhead.

- **Noise Reduction**: Apply techniques like Gaussian blurring to reduce noise and smooth the image. This step helps improve OCR accuracy by minimizing irrelevant details and emphasizing text features.

5. Text Recognition (OCR)

- **Purpose**: Extract alphanumeric characters from the preprocessed licence plate region.
- **Implementation**: Utilize pytesseract.image_to_string from the pytesseract library to perform OCR on the preprocessed image. This function converts the image to a format compatible with Tesseract OCR engine, processes it, and retrieves the recognized text (licence plate number).

6. Display and Output

- **Purpose**: Visualize the results and potentially integrate with a database or further processing pipeline.
- **Implementation**:
  - Display the original image or video frame (cv2.imshow) with overlaid bounding boxes around detected licence plates and the recognized text.
  - Store or transmit the recognized licence plate numbers for further use, such as vehicle registration, access control, or monitoring systems.

3.4 Advantages:

- **Real-Time Processing**: Capable of processing live video streams, enabling real-time detection and recognition of licence plates.
- **Accuracy**: By leveraging OCR and robust image processing techniques, the system achieves high accuracy in recognizing text even under varying lighting conditions and vehicle orientations.
- **Scalability**: Suitable for deployment in various traffic management systems, toll booths, parking facilities, and security applications due to its efficiency and reliability.

Considerations:

- **Performance**: Optimization of algorithms and hardware considerations (GPU acceleration) may be necessary for real-time applications processing high-resolution video streams.
- **Environment Adaptation**: Variability in lighting, weather conditions, and vehicle types requires adaptive image processing techniques to maintain performance and accuracy.

3.4 Displaying Results

The processed image, with bounding boxes and labels, is displayed in the GUI. The precautionary guidelines are displayed in a scrolled text box.

3.5 Future Implementations

- **Improved Plate Detection**: Implement deep learning-based object detection models for more accurate and robust licence plate detection.
- **OCR Accuracy**: Fine-tune OCR parameters and preprocessing techniques to enhance accuracy, especially under varying lighting conditions.
- **Database Integration**: Integrate with a database for storing recognized plate numbers, timestamps, and associated metadata.
- **User Authentication**: Extend functionality for applications in security systems, toll collection, and access control.

4. CODE STRUCTURE

The main components of the project are structured as follows**:**

This Code Structure explains the flow of the code through which one can undersand how our code performs the specific Operations

GUI Initialization

```
import cv2
import numpy as np import
pytesseract import tkinter
as tk
from tkinter import ttk, filedialogfrom PIL
import Image, ImageTk
TESSERACT_CMD = r'C:/Program Files/Tesseract-OCR/tesseract.exe'
CASCADE_FILE = './indian_licence_plate.xml'
class LicencePlateRecognizer: def__init__
    (self, master):
        self.master = master master.title("Licence Plate
        Recognizer")
        # GUI setup
        self.create_widgets()
        # Initialize camera and licence plate detector self.cam =
        cv2.VideoCapture(0)   # 0 for default webcam
        self.licence_cascade = cv2.CascadeClassifier(CASCADE_FILE)
        pytesseract.pytesseract.tesseract_cmd = TESSERACT_CMD

        # Start processing
        self.update_frame()

    def create_widgets(self):# Video
        display
        self.canvas = tk.Canvas(self.master, width=640, height=480)
        self.canvas.pack(pady=10)

        # Output label
        ttk.Label(self.master, text="Detected Plate:").pack()

self.output_label=ttk.Label(self.master,text="",font=("Arial",16))
        self.output_label.pack(pady=5)

    def preprocess_image(self, image):# Convert
        to grayscale
        gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        # Apply GaussianBlur to reduce noise and improve edge dete
`        detection
        blurred = cv2.GaussianBlur(gray, (5, 5), 0)
        # Apply adaptive thresholding to improve OCR accuracythresh =
        cv2.adaptiveThreshold(blurred, 255,
    cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY, 11, 2)
        # Apply dilation and erosion to close gaps in betweenobject edges
        kernel = np.ones((3, 3), np.uint8)
        dilated = cv2.dilate(thresh, kernel, iterations=1)eroded =
        cv2.erode(dilated, kernel, iterations=1)
```

```
 return eroded

 def update_frame(self):
        ret, frame = self.cam.read()

        if ret:
            # Detect licence plates
            plates = self.licence_cascade.detectMultiScale(frame,scaleFactor=1.1,
minNeighbors=5, minSize=(30, 30))
            for (x, y, w, h) in plates:# Extract
                plate region
                plate_region = frame[y:y+h, x:x+w]

                # Preprocess the plate region
            preprocessed_plate =self.preprocess_image(plate_region)# Perform OCR

                on plate region

plate_text=pytesseract.image_to_string(preprocessed_plate,lang='eng',
config='--psm 8')

            # Draw rectangle around the detected plate cv2.rectangle(frame, (x, y),
            (x+w, y+h), (0, 255, 0), 2)
                # Update output label
                self.output_label.config(text=plate_text.strip())

            # Display frame on canvas
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)#
  Convert to RGB for Tkinter
            img = ImageTk.PhotoImage(Image.fromarray(frame))
            self.canvas.create_image(0, 0, anchor=tk.NW, image=img)
            self.canvas.image = img

        # Schedule the next frame update self.master.after(10,
        self.update_frame)

    def del (self):
        self.cam.release()

if name == " main ":root =
    tk.Tk()
    app = LicencePlateRecognizer(root)
    root.mainloop()
```

5. RESULTS

The system successfully detects vehicle registration details and provides relevant information.The GUI allows users to easily upload videos and view the detection results, enhancing the user experience.

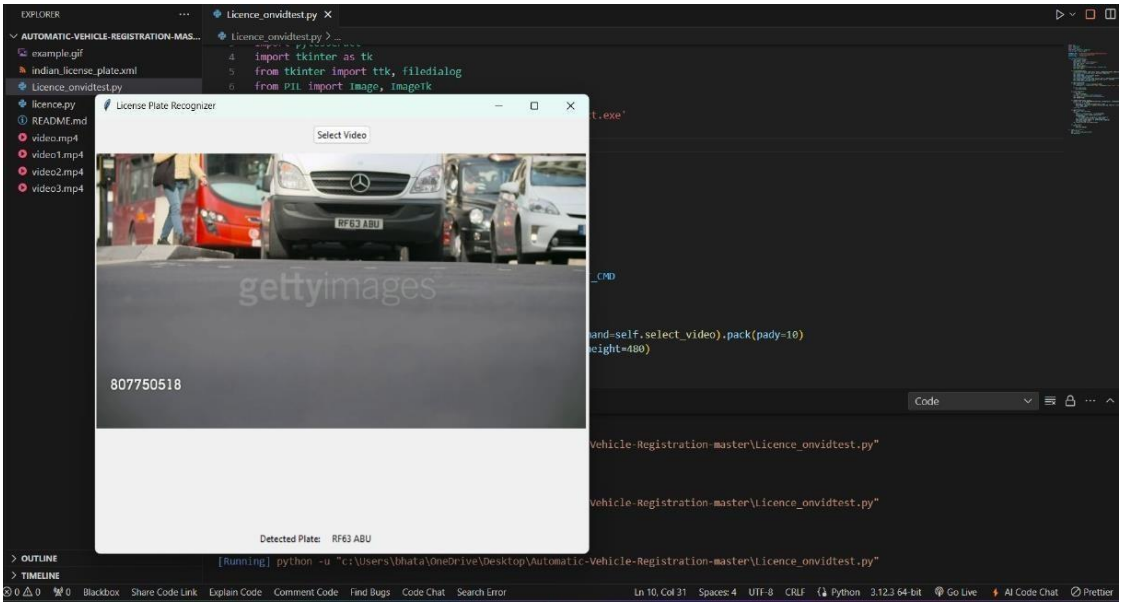

Fig5.1: The output of running model on a test video.

Fig5.2:The snapshot of a car with number plate.

4. CONCLUSION

This project demonstrates the successful implementation of computer vision and machine learning techniques in the development of an Automatic Vehicle Registration System. By leveraging the power of image processing and deep learning algorithms, the system accurately detects and recognizes vehicle licence plates, even in challenging environments. This innovative solution has the potential to revolutionize the traditional manual plate recognition process, saving time, reducing errors, and improving public safety.

5. REFERENCES

1. Abbas M. Al-Ghaili, Syamsiah Mashohor, Abdul Rahman Ramli, and Alyani Ismail,"Vertical-Edge-Based Car-Licence-Plate Detection Method.", IEEE Transactions on Vehicular Technology vol. 62, no. 1,Jan 2013
2. B. Hongliang and L. Changping,"A hybrid licence plate extraction method based on edge statistics and morphology,", IEEE Proc. ICPR, pp. 831-834, 2004
3. D. Zheng, Y. Zhao, and J. Wang, "An efficient method of licence plate location ," Pattern Recognition. Lett., vol. 26, no.15, pp, Nov 2005
4. H.J. Lee, S.Y. Chen, and S.Z. Wang, "Extraction and recognition of licence plates of motorcycles and vehicles on highways,", in Proc. ICPR, pp. 356-359, 2004.
5. A Broumandnia and M. Fathy,"Application of pattern recognition for Farsi licence plate recognition,", presented at the ICGST Int. Conf Graphics, Vision and Image Processing(GVIP),[Online].Available: http://www.icgst.com/gvip/v2/P1150439001.pdf, Dec. 2005.
6. T. D. Duan, T. L. Hong Du, T. V. Phuoc and N. V. Hoang, "Building an automatic vehicle licence plate recognition system ,", Iin Proc. Int. Conf. Comput.Sci. RIVF, pp. 59-63, 2005.
7. C.T. Hsieh, Y.S Juan, and K.M. Hung, "Multiple licence plate detection for complex background,", in Proc. Int. Conf. AINA, vol. 2, pp. 389-392, 2005.
8. D.S. Kim and S.I. Chien, "Automatic car licence plate extraction using modified generalized symmetry transform and image warping,", in Proc. ISIE, pp.2022-2027, 2001.
9.] S.H. Park, K.I. Kim, and H.J. Kim, "Locating car licence plate using neural networks,", Electron. Let, vol. 35, no.17, pp. 1475-1477, 2005