CHAPTER1: INTRODUCTION

1.1 INTRODUCTION

We force the introduction of robots able to figure out styles in herbal conduct and addiction. automatic optical man or woman popularity, facial and speech popularity, fingerprint and DNA collection identification, and plenty of other issues are the announcing that accurate and reliable sample reputation would be extremely beneficial to machine mastering. If you want to create a laptop gadget that may mechanically extract and interpret textual content from images, studies into optical person popularity are now underway. These days, there's an awesome need to keep statistics from the various resources—whether handwritten or revealed—ona computer storage disc for later use. Consequently, a way for robotically extracting and saving data—especially textual content—from photo files is required. However, this particular undertaking is not very easy. But if successful, this could have a vast impact on nearly every area of image processing observed.

1.2 PROBLEMSTATEMENT

The aim for us is to utilize the optical character recognition (OCR) and conduct experiments so that we can get optimal outcomes. We initially create a technique for automatically identifying license plates. Through thorough investigation and testing, we could improve the accuracy of the system and extend its application to other Optical Character Recognition domains and real-time activities.

1.3 OBJECTIVES

- To optimize pictures for OCR, use OpenCV for image preparation. Resize, denoise, and improve photos

- Use the library for Easy OCR: To extractal phanumeric characters from preprocessed license plate photos, enable optical character recognition.
- Give accuracy first priority: Accurately recognize license plate numbers in a variety of lighting and ambient circumstances with accurate character recognition and localization.
- Develop a fast and accurate system for processing video, combining YOLO v8 and tesseract.
- Utilizing the COCO model, train the system on a variety of data sets, and monitor its performance over time using stringent assessment metrics such as precision and recall.
- For database support, integrate with Pandas to facilitate easy data management. For simple deployment and monitoring, offer a user-friendly interface with strong error handling and privacy features.

1.4 SIGNIFICANCE AND MOTIVATION OF THE PROJECT WORK

- By making it possible to more quickly identify and apprehend vehicles implicated in criminal activity, the use of number plate recognition in both pictures and videos facilitates law enforcement efforts and increases public safety.
- By automatically identifying cars from images and videos, the system helps traffic management authorities enforce laws like speeding and illegal parking, which improves road safety and traffic flow.
- Number plate recognition technology makes automated toll collecting procedures possible. This eliminates the need for actual toll booths, reduces traffic, and increases the efficiency of transportation as a whole.

- Effective vehicle tracking and monitoring benefits businesses in the fleet management, logistics, and public transportation industries. This promotes more efficient operations, better asset protection, and higher-quality customer service.

- The advancement of optical character recognition methods, object detection algorithms, and creative approaches to managing massive amounts of video data are all fueled by the development of number plate recognition technologies, which are applicable to both photos and videos. These developments further the domains of computer vision and artificial intelligence.

1.5ORGANIZATION OF PROJECT REPORT

The five chapters that make up this project report are as follows:

Chapter 1:

Automated Optical Character Recognition (OCR) covers face, speech, and the fingerprint identification. It is a key component in the quest for advanced pattern recognition. With extensive research and testing, this project aims to create a reliable optical character recognition (OCR) system for automatic license plate recognition, increasing accuracy and expanding applications to real-time operations.

Chapter 2:

This chapter incorporates facts on estimate techniques from previous research. Neural networks, device mastering, and deep getting to know are also protected in this bankruptcy. numerous journals and pertinent publications that offer an outline of preceding paintings had been released. The section covered the many models that researchers have attempted to apply in their quest to develop a deep gaining knowledge of model that works. The strategies we need to use to teach/create the version are decided by means of the techniques and consequences discussed.

Chapter 3:

The suggested method makes it easier to extract the text from the images by superimposing the result output over the original input. It utilizes filtering and grey scalability as pre-processing techniques, leveraging OpenCV for real-time computer vision. The system improves license plate identification accuracy by using EasyOCR , tesseracet and yolov8 , an optical character recognition technology. NumPy, COCO, Matplot, PyTorch, ultralytics and Python are examples of supporting tools for effective image processing, machine learning, and user-friendly interfaces. Filtering ,binarization, sharpening, masking, and edge detection techniques are among the image processing approaches that improve the system's performance

Chapter 4:

Pre-processing is the first step in the Optical Character Recognition (OCR) process, and it deals with issues caused by different fonts and writing styles. Image readability is improved by methods like binarization, de-skewing, and de- speckling. Analyzing character strokes or applying pattern recognition are two methods used in feature extraction. Vocabularies are used in post-processing to improve accuracy by fine-tuning OCR output. Techniques for correcting errors, such as nearest neighbor analysis, are utilized. For the purpose of identifying license plates, the workflow entails image acquisition, reading, pre-processing, edge detection, and segmentation..

Chapter 5:

This chapter contains the entire work that is the subject of the project summary. Our all-encompassing work addressed OCR processes, key phases, architecture, and techniques, particularly as they relate to automatic license plate recognition (ANPR). We developed a method for identifying license plates from moving vehicles usingOpenCV and EasyOCR, acknowledging the inherent complexities in ANPR due to various factors. We sought to reduce errors and improve OCR system performance while acknowledging current approaches.

CHAPTER2: LITERATURE REVIEW

Paper Title	**Journal/ Conference (Year)**	**Tools/ Techniques/ Dataset**	**Results**	**Limitations**
[1]Automatic License Plate Recognition System for Vehicles Using a CNN	Tech Science Press (2021)	CNN technique is used.	High recognition rate of 98.13% when testedon160images	The CNN can not distinguish among the plate region and the grills of the motor car
[2]Vehicle Number Plate Detection and Recognition Techniques	Advances in Science, Technology and Engineering Systems Journal(2021)	Techniques are used CNN, SVM,YOLOv2	90.50% to mean average accuracy (mAP)andrecall of 0.86duringtraining	Worked limited images
[3]Automatic number plate recognition using Deep learning	ICCSSS(2020)	An efficient deep learning model such as You Only Look Once (YOLO) is used for object detection.	Accuracyof98%for number plate localization and accuracy90% achieved for CR.	IMAGEAI framework is used which is very much efficient than YOLO object detection
[4]Automatic Vehicle Number Plate RecognitionSystem Using Machine Learning	CHSN(2020)	OCRMETHODISUSED	Accuracy92%in this model	No Data set is used

[5]A Reinforcement Learning-based Offload Decision Model (RL-OLD) for Vehicle Number Plate Detection	I. J. Engineering and Manufacturing,(2021)	TechniquesisusedYOLOV3, SSD Lite MobilenetV2,Reinforcement Learning	Recognition accuracy reached 93.15%	The limitation of models that are only deployed in the cloud
[6]Recognition of Vehicle Number Plate Using Matlab	Journal of University of Shanghai for Science and Technology(2021)	SVM,MATLAB	Accuracy87%in this model	Image processing and other hardware requirements
[7]Vehicle Classification Based on CCTV Video Recording Using Histogram of Oriented Gradients Local Binary Patterns and Hierarchical Multi-SVM	IOP Conference Series(2021)	Using Hierarchical Multi-SVM	Accuracy of 80.28%, precision of 94.27%, recall of 73.79%, and F-measure of 82.76%.	Use efficient algorithms and hardware for speed and resource management.
[8]Facial Expression Recognition on Video Data with Various Face Poses Using Deep Learning	ICITEE(2021)	Using YOLO and CNN	Accuracy of 73%.	Used less set of videos that why result is less
[9]An algorithm for accuracy enhancement of license plate recognition	Journal of computer and system sciences (2021)	Using local haar feature and open source OCR	Recognition rate is 94.03%	It works on fixed sized of number plate.

[10]Automatic Recognition of Non-standard Number Plates using YOLOv8	INDIAcom(2024)	Using YOLOv8 and easy OCR	Giving 90% of accuracy	Not working on all number plates

Table 1: LITERATURE REVIEW

CHAPTER3: SYSTEM DEVELOPMENT

3.1 REQURIMENTS AND ANALYSIS

- This device can utilize a picture as an input and output the extracted text as an overlay at the image.
- It applies strategies for pre-processing to the enter image, including filtering, polishing, masking, grey scalability and conversion to RGB, amongst others.
- It makes use of the car range plate input photographs enter, locates the plate's role, and outputs the plate quantity.

3.1.2 OPENCV

The development of libraries for real-time computer vision applications has advanced significantly, most notably with the introduction of the Go platform. For these applications, object and face identification are important focal points, and image processing, video recording, and analysis receive a lot of attention. One popular library in this field is OpenCV, which is supported on a broad range of platforms, including Windows, Linux, OSX, Android, and iOS. It also supports a wide variety of programming languages, such as Java, Python, and C++. More performance improvements are planned, especially for high-speed GPU applications, and CUDA and OpenCL-based interfaces are being developed. Specifically, OpenCV-Python, the OpenCV Python API, combines the finest aspects of the Python programming language with the powers of the OpenCV C++ API, giving computer vision developers a strong and adaptable toolkit.

3.1.3 NUMPY

A wide range of complex mathematical operations may be supported by the NumPy library for the Python programming language when working with large, multi-dimensional arrays and matrices. Despite not being designed with numerical computation in mind, the Python programming language quickly gained popularity within the medical and technical communities. A special set of programs called Matrix-sig was established in 1995 to provide an environment for array computation. Guido van Rossum, the creator and maintainer of Python, was one of its members. To make array computation simpler, he advanced the language's grammar, particularly the indexing syntax.

3.1.4 MATPLOT

Our applications can incorporate diagrams through the use of a general-purpose GUI toolkit including, but not limited to, Tkinter, WXPython, Qt, or GTK+. This is the plot package for the Python programming language and the NumPy math numerical package. Matplotlib is used in SciPy. There is a thriving open-source developer community around matlotlib built with Hunters, similar as bsd. Thomas Caswell, and Michael Droettboom before the death of John Hunter in August, 2012 managed the matplotlib project together.

3.1.5 PYTORCH

The middle of the open-source PyTorch device learning (ML) platform is Python and the Torch library. Torch is an open-supply device learning device built on pinnacle of the Lua programming language this is used to create deep neural networks. that is one of the most famous deep gettingto know studies systems. The framework objectives to facilitate a quicker transition from research prototyping to implementation. Over 2 hundred one of a kind kinds of mathematical operationsaresupportedwiththeaidofthePyTorchframework.PyTorchisturninginto increasingly famous because it makes the manner of making fashions for artificial neural networks a lot easier. information scientists who paintings on synthetic intelligence (AI) programs and research are those who use PyTorch the maximum. A changed model of the BSD license governs PyTorch's release.

3.1.6 EASYOCR

Projects can be done by using the EasyOCR library for optical character recognition. Via OCR analysis, digital picture input, which in turn can also be in the form of printed material or hand writings, is converted into a machine readable digital text format. They are again separated into their smallest parts and matched against character dictionary in search of the text, words, and individual characters. This happens after OCR splits in the digital image. This problem can only be fixed in an environment offered by Python which is a programming language. Never the less, our big stock house of OCR is available for imports and usage. Python has several uses including algorithm design, computation, analysis among others. This solves the problem as it comes and lays out a way forward for us.

3.1.7 PYTHON

Python is a great option for both novice and experienced programmers because of its exceptional simplicity and ease of use. Python's simple syntax and uncomplicated structure reduce needless complexity, making it possible for learners to pick up the language more quickly. Its source code is very clear and easy to comprehend, which makes maintenance simple. Furthermore, Python's flexibility is demonstrated by its ability to run on a variety of operating systems, such as Windows, Macintosh, and UNIX. The language facilitates testing and mistake repair by providing a variety of interaction mechanisms. Furthermore, Python has a similar UI across platforms, which makes it more portable. Because of its modular architecture, expanding modules may be easily integrated, giving system designers more flexibility to improve the efficiency of their instruments. Additionally, Python makes it easier to create user interfaces for applications such as Macintosh MFC, Windows MFC, and Windows Unix X. Python's scalability and usability make it a significant tool for large-scale projects, even though its build and support may not be necessary for every project.

3.1.8 PANDAS

Mostly used for data analysis and processing, Pandas is a flexible Python package that has a somewhat unrelated but useful use in number plate recognition projects. Pandas has features that can help manage and preprocess data related to the recognition task, even if they are not directly engaged in the recognition task itself. It is excellent at batch processing jobs, arranging data for analysis, and organizing image-related metadata or annotations. Pandas also has the ability to analyze and visualize data, thus it may be used to investigate the performance of recognition models, the distribution of characters in training datasets, and the general development of the recognition system. Its smooth interaction with other Python libraries, such NumPy and scikit-learn, which are frequently used in computer vision and machine learning, making it an invaluable tool in the preprocessing and arrangement of data before to using it in machine learning models that are trained to recognize license plate images. Pandas is a key supporting component in streamlining data administration and analysis in the recognition pipeline, even if it is not directly engaged in recognition algorithms.

3.1.9 YOLOv8

Because of its outstanding speed, accuracy, and adaptability, YOLOv8, or You Only Look Once version 8, stands out as a preferred option in number plate recognition systems. Because it is designed for real-time operation, it processes pictures quickly, which is an essential characteristic for applications where quick analysis is required, such as surveillance or traffic monitoring. Even in fast-moving environments with occlusions and changing illumination, YOLOv8 consistently detects and locates objects—including number plates—with high accuracy levels. Its end-to-end approach removes the need for extra processing stages and streamlines identification processes. Additionally, YOLOv8 provides versatility by allowing developers to customize parameters and structures, enabling them to optimize models for particular datasets or circumstances. With the help of a thriving community and a wealth of resources, like as pre-trained models and thorough documentation, it can be quickly implemented and troubleshooted. YOLOv8 stands out as a strong option for number plate recognition systems because of its combination of real-time performance, accuracy, adaptability, and community support. It also promises effective and efficient object identification in a variety of settings.

3.1.10 CVZONE

For number plate recognition systems, CVZone, a Python package with a set of tools for computer vision applications, is a good choice because of a few important benefits. Because of its high-level abstractions and wrappers around OpenCV functions, complicated jobs may be implemented more quickly and with less complex code, which speeds up development. Furthermore, CVZone offers additional features not found in OpenCV, which might improve number plate recognition systems by using sophisticated image processing methods and pre-trained models. Its simple architecture, which comes with comprehensive documentation and examples, makes it easy to prototype and implement quickly. This is especially helpful for developers who are not experienced with computer vision. Though it may not have the same level of community support as other, more well-known libraries, CVZone still has a user and development community that promotes cooperation and helps with problem solving. Although not the only option With its easier-to-use interface, improved functionality, community support, and streamlined implementation for developers looking for effective computer vision solutions, CVZone is a vital tool for number plate identification.

3.1.11 TESSERACT

Because of its precision, adaptability, and accessibility, Tesseract is an open-source optical character recognition (OCR) engine that is frequently used in number plate identification systems. Tesseract, which is always being improved upon, has a high degree of accuracy when it comes to removing text from photographs, especially in difficult situations when there are several typefaces and orientations. It is appropriate for a variety of number plate styles throughout the world due to its capability for many languages and character sets. Furthermore, Tesseract removes financial barriers by being free and open-source, allowing developers to include powerful OCR features into recognition systems without having to pay for proprietary software. Tesseract is backed by a thriving community that provides regular updates, bug patches, and support, guaranteeing its dependability and applicability in applications involving number plate recognition. Because of its easy interaction with Python thanks to libraries like pytesseract, development is made simpler and it can be quickly added to pipelines that already exist. additional machine learning and image processing components. Fundamentally, Tesseract's reputation as the go-to OCR engine for text extraction from license plate photos in recognition systems is cemented by its combination of precision, adaptability, accessibility.

3.2 PROJECT DESIGN AND ARCHITECTURE

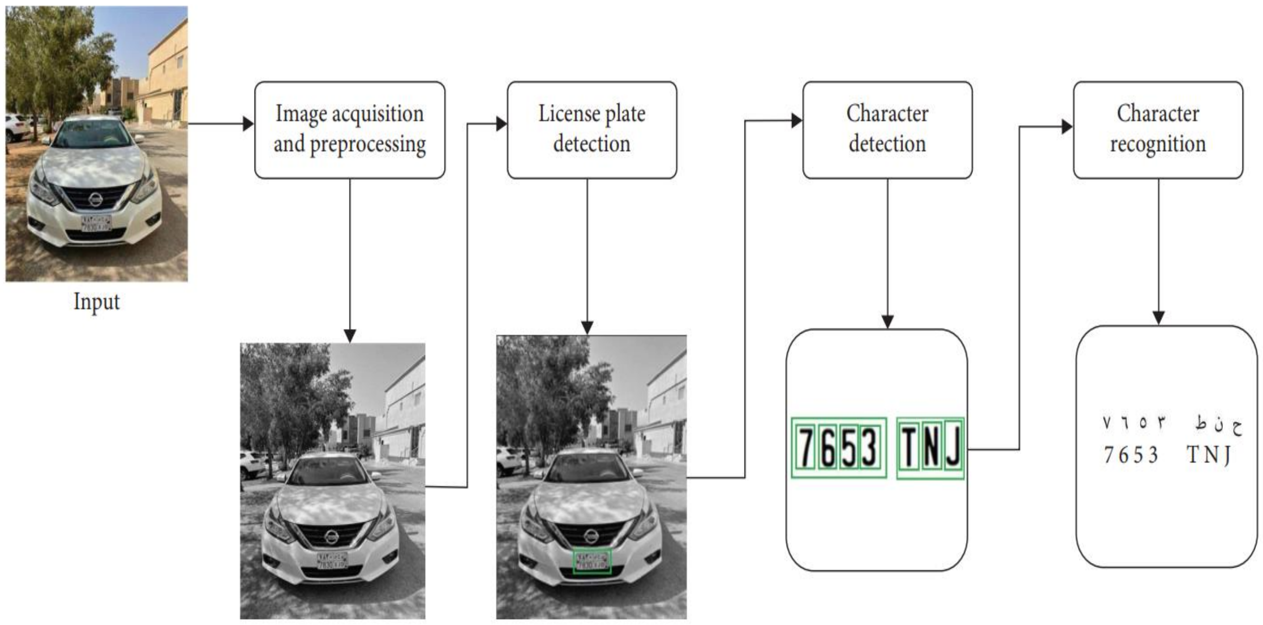


Figure 1 : flow chart license plate detection and recognition

The process begins by capturing an image of a vehicle's license plate. This image might come from a stationary camera or one mounted on a moving vehicle. The raw image can be noisy and unsuitable for direct character recognition. Here, the system improves the image quality through various techniques. This might involve converting the image to grayscale, sharpening edges, and reducing background clutter. Not every image will perfectly center the license plate. This step identifies the specific region within the image containing the license plate itself. Algorithms often look for rectangular shapes with contrasting colors (like dark text on a light background) to isolate the plate. Once the license plate is located, individual characters need to be separated for recognition. This might involve techniques that analyze the spacing and connectedness of pixels within the plate region. Each isolated character is fed into an Optical Character Recognition (OCR) engine. The OCR engine compares the character's shape and features to a database of known characters and outputs the most likely alphanumeric representation.

3.3 DATA PREPARATION

To ensure that the image can be read,

3.3.1PRE-PROCESSING OF THE IMAGE

Techniques include:

- De-skewing
- De-speckle
- Binarization
- Line removal
- Layout an alysis or zoning
- Line and word detection
- Script recognition
- Character isolation or segmentation
- Normalization

3.3.2 FEATURE EXTRICATION

To extract features in OCR, there are mainly two methods:

- The first technique utilizes the feature detection algorithm which determines if the characters have their strokes and then detects them.
- In this second way, the whole character is used to find out what it does and a pattern recognition.

A line of text is found by looking for white pixel rows that are separated by black pixels. Similarly, one may distinguish the start and finish of a character.

3.3.3POST PROCESSING

A vocabulary enables one to filter OCR output to improve it, thereby limiting it (to a list ofterms acceptable to a certain document). These could include a list of all the words used in English or a highly specific wordbook on a particular subject area. However, this strategy may not work for texts that contain names (proper nouns), numbers and other non-lexical words. Fortunately, there exist freely available, highly accurate OCR libraries for that purpose. Through the dictionary in the Tesseract system, characters aresegmented.

3.3.4 BUILDING THE AUTOMATIC LICENSE PLATE RECOGNITION SYSTEM

The plurality of the number plate identification algorithms have emerged through various methods and form several categories. The following components are:

- The plate picture of a car may vary in size.
- Plate location: It would therefore pose a significant hazard in any part of the car including the seat, dashboard and passenger compartment.
- The color of the backdrop may vary depending on the type of vehicle. Such as, on a government-owned car’s number plate there could be different background of color than any other public cars.

3.3.5 METHODOLOGICAL PROCESS

A. Image Acquisition– It is getting the image that will serve as an input for the system. This may include a purchased image database or a manually developed actual set of pictures.
B. Image Reading – openCV in Python reads the original image to be used as input. Visualizing the original image for subsequent processing and the following evaluation.
C. Image Preprocessing – entails several steps in altering and reformatting the original image to ensure its suitability for processing. Different techniques for preprocessing the images are:

- Image binarization – It means converting a particular colored image into black and white. Some of the pixels in this method are deemed to be either black or white depending upon a particular threshold value. The principal problem is to choose a suitable threshold level within an assigned image. Automatic thresholding refers to the process of selecting a threshold using an algorithm instead of manually.

- Image Filtering – Practice of applying different filters to theimages in order to remove noise, blur, sharpen, or otherwise make the image better for processing is known as image filtering. The Laplacian filter is the one used in this project to sharpen the image and make edge detection easier.

- Edgedetection– Edge detection remains the most important technique of feature extraction or feature detection. Often, the effect is similar to a connected curve-like contour describing an object's boundary. This method is very challenging especially when working on complex images, as it can result into having disjointed contour of objects.

- Blob Detection / Contour Detection – Blob detection enables detecting blobs as parts with strong differences in hue or brightness from the surrounding regions. The main objective of this approach is to identify additional complementarity regions missed out in corners and edges detection. Commonly used blob detectors are also referred to as maximally stable extremal area, difference of the gaussians and determinant of hessian, laplacian of gaussian.

- Segmentation –It is possible to extract a license plate using image segmentation algorithms. Many books include lists of various image segmentation techniques. Most methods use picture binarization to convert color images into grayscale. Segmentation may take the following forms:

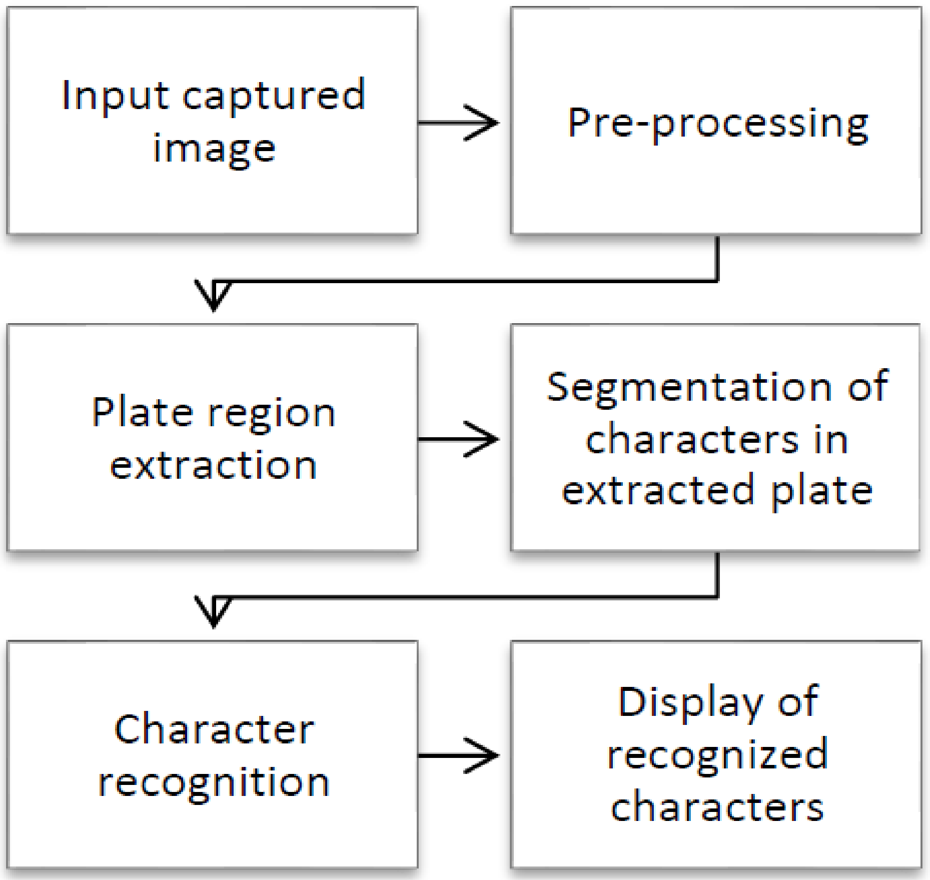


Figure 2 : Flow Diagram for ANPR

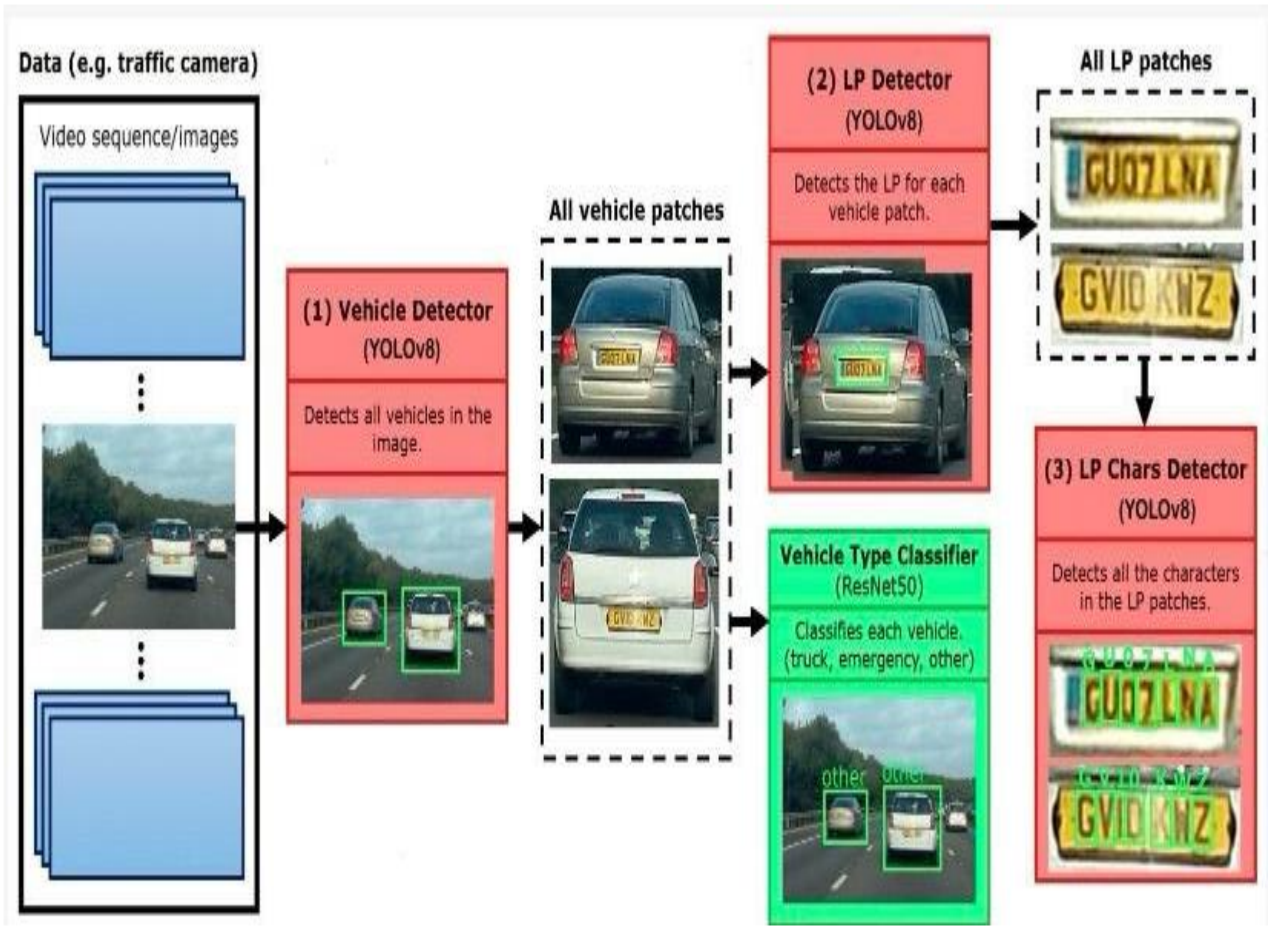

Data (e.g. traffic camera)
Video sequence/images
(1) Vehicle Detector
(YOLOv8)
Detects all vehicles in the image.
All vehicle patches
(2) LP Detector
(YOLOv8)
Detects the LP for each vehicle patch.
Vehicle Type Classifier
(ResNet50)
Classifies each vehicle. (truck, emergency, other)
other
other
All LP patches
GU07 LNA
GV10 KWZ
(3) LP Chars Detector
(YOLOv8)
Detects all the characters in the LP patches.

Figure 3: flow chart license plate detection and recognition for video

Data preprocessing from video for lincense plate dectection:

1) Capture Images from video using CV2:
   a) To read the video file in Python, use the OpenCV (cv2) package.
   b) Take frames out of the movie at predetermined intervals or every nth frame.
   c) The retrieved frames and photos should be saved to a specific folder

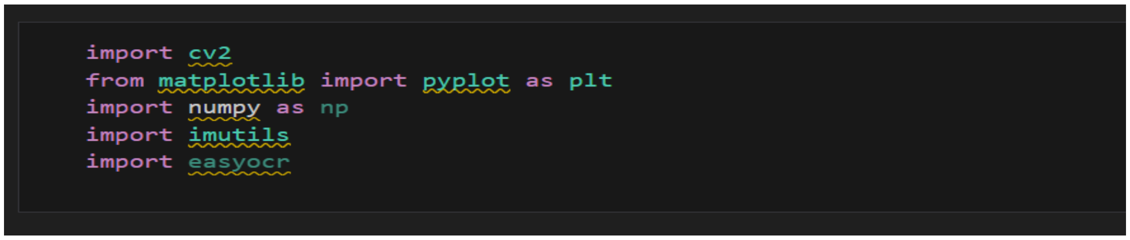
2) Number plate with labeling:
   a) Apply a hand label to the number plates in the pictures using a program such as LabelImg.
   b) Enclose the license plates in bounding boxes and designate them with the label "number_plate".
   c) The annotations can be saved in XML or any other format that works.

3) Create Dataset
   a) Combine the folder's unlabeled and labeled photos (including number plate annotations).
   b) Put these picture in a training-ready dataset structure.

4) Train dataset with YOLOv8
   a) Put the You Only Look Once version 8 (YOLOv8) object detection method into practice.
   b) Utilizing the labeled dataset produced in the preceding phase, train the model.
   c) If required, adjust the model to increase accuracy.

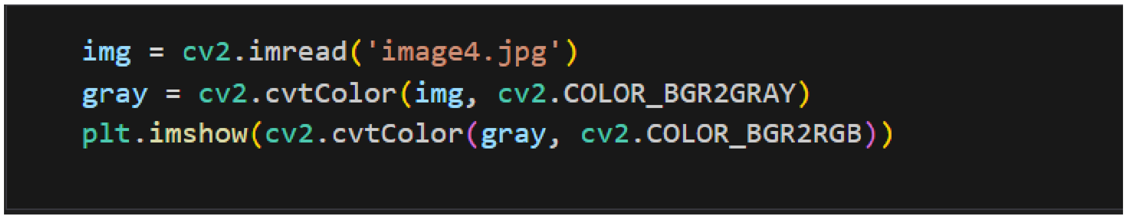
3.4 IMPLEMENTION

3.4.1 CODE SNIPPETS FOR IMAGE DETECTION



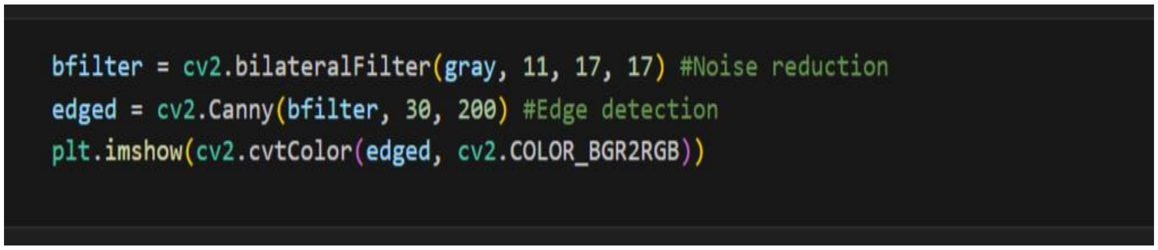
```
import cv2
from matplotlib import pyplot as plt
import numpy as np
import imutils
import easyocr
```

Figure 4: Import and install dependencies for license plate detection



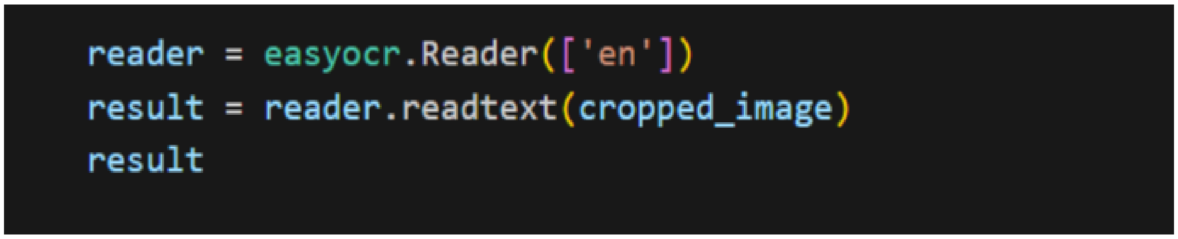
```
img = cv2.imread('image4.jpg')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
plt.imshow(cv2.cvtColor(gray, cv2.COLOR_BGR2RGB))
```

Figure 5: To read a image used grayscale and blur



```
bfilter = cv2.bilateralFilter(gray, 11, 17, 17) #Noise reduction
edged = cv2.Canny(bfilter, 30, 200) #Edge detection
plt.imshow(cv2.cvtColor(edged, cv2.COLOR_BGR2RGB))
```

Figure 6: Used canny to find the edge of plate



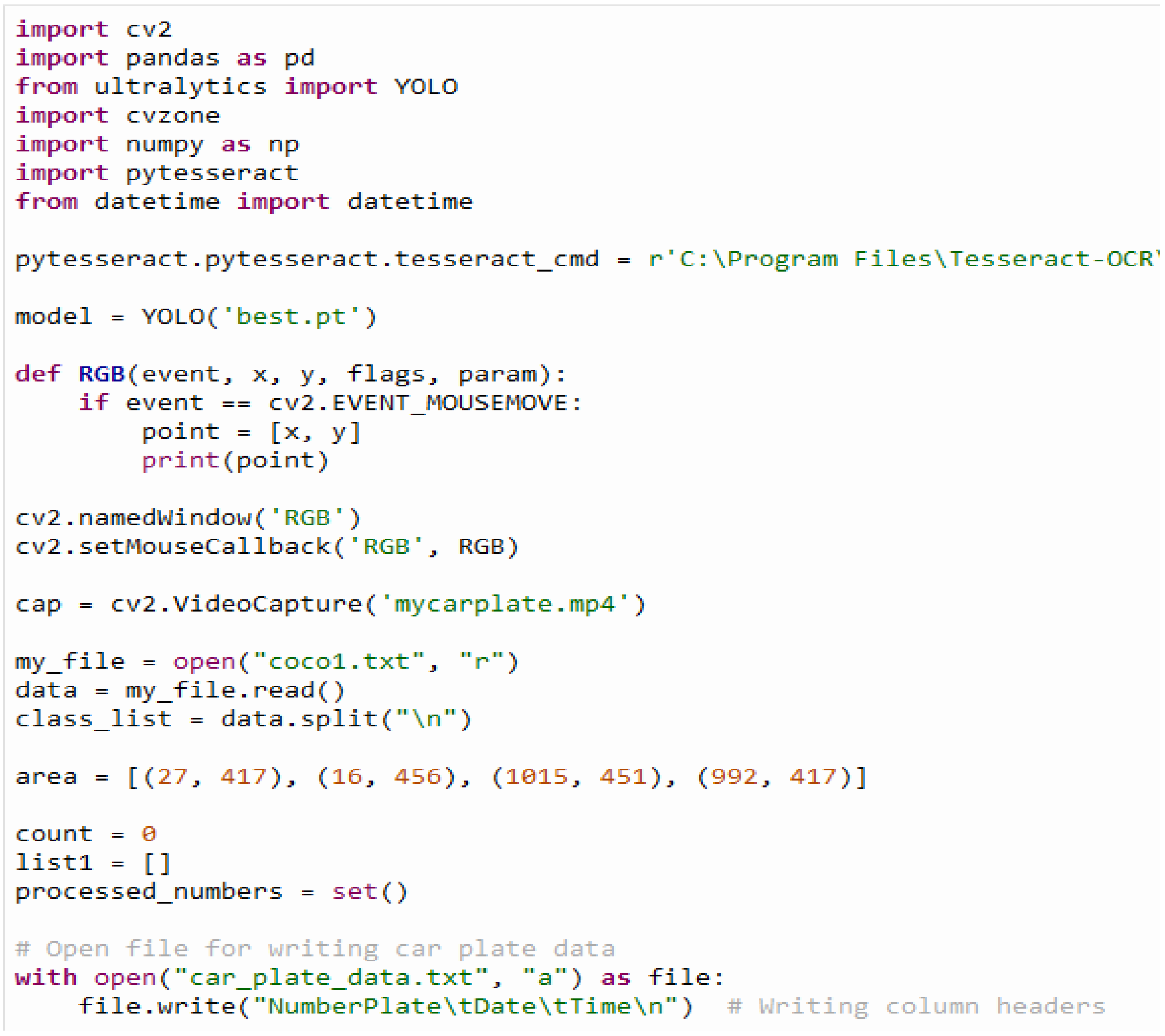
```
reader = easyocr.Reader(['en'])
result = reader.readtext(cropped_image)
result
```

Figure 7: Used EasyOCR

3.4.2 Tools, algorithms and techniques used for image license plate detection

Preprocess the photos to improve features and minimize noise by utilizing NumPy for effective array operations and OpenCV's capabilities for image processing and manipulation. With the help of Imutils' efficient image processing, you may identify regions of interest that correlate to number plates in the photos by using techniques like contour and edge recognition. Following the identification of these locations, text recognition is performed using EasyOCR, a Python tool based on the Tesseract OCR engine, to extract characters from the number plates. Steps in post-processing might come next, honing the retrieved text for greater precision. Your project successfully and accurately detects and extracts number plates from photos by utilizing these tools and approaches.

3.4.3 CODE SNIPPETS FOR VIDEO DETECTION

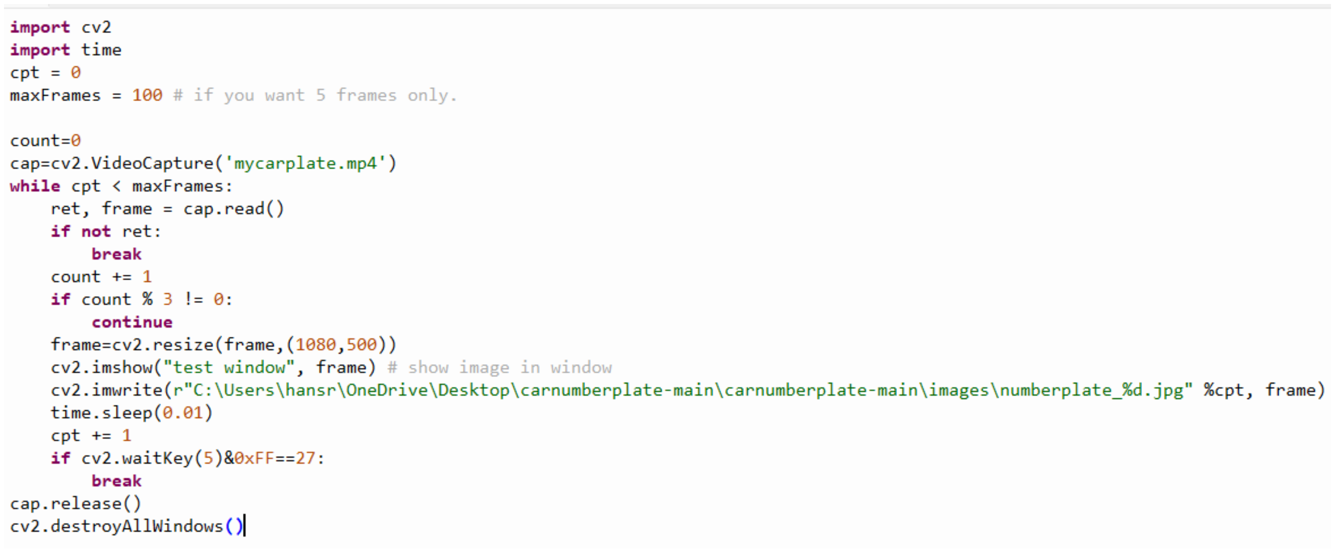


```python
import cv2
import pandas as pd
from ultralytics import YOLO
import cvzone
import numpy as np
import pytesseract
from datetime import datetime

pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR'

model = YOLO('best.pt')

def RGB(event, x, y, flags, param):
    if event == cv2.EVENT_MOUSEMOVE:
        point = [x, y]
        print(point)

cv2.namedWindow('RGB')
cv2.setMouseCallback('RGB', RGB)

cap = cv2.VideoCapture('mycarplate.mp4')

my_file = open("coco1.txt", "r")
data = my_file.read()
class_list = data.split("\n")

area = [(27, 417), (16, 456), (1015, 451), (992, 417)]

count = 0
list1 = []
processed_numbers = set()

# Open file for writing car plate data
with open("car_plate_data.txt", "a") as file:
    file.write("NumberPlate\tDate\tTime\n")  # Writing column headers
```

Figure 8: Import and install dependencies for license plate detection

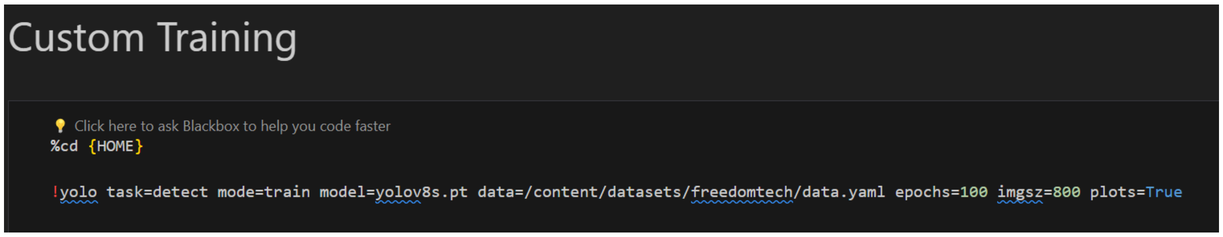
The foundation for image and video processing jobs is OpenCV (cv2), which provides features like mouse event handling and video file reading that are essential for user interaction with the UI. Although NumPy isn't seen in the clip, it probably plays a crucial part in numerical operations on arrays, which are necessary for image processing jobs. The ultralytics library's YOLO (You Only Look Once) model is notable for its deep learning-based approach to object recognition, which makes it possible to quickly and accurately identify objects—including license plates. In terms of optical character recognition, Tesseract OCR (pytesseract) is superior as it enables the extraction of text from photos, which is especially helpful for obtaining alphanumeric characters.

Although it isn't mentioned in the sample specifically, CVZone is a promising library for computer vision jobs that could provide more features. Pandas is essential for data manipulation and analysis even when it isn't used directly; it probably helps with handling data structures or processing data from files. Simple file handling methods make data access and storage easy, but event handling features—especially with OpenCV—improve user engagement and promote a more natural experience.



```python
import cv2
import time
cpt = 0
maxFrames = 100 # if you want 5 frames only.

count=0
cap=cv2.VideoCapture('mycarplate.mp4')
while cpt < maxFrames:
    ret, frame = cap.read()
    if not ret:
        break
    count += 1
    if count % 3 != 0:
        continue
    frame=cv2.resize(frame,(1080,500))
    cv2.imshow("test window", frame) # show image in window
    cv2.imwrite(r"C:\Users\hansr\OneDrive\Desktop\carnumberplate-main\carnumberplate-main\images\numberplate_%d.jpg" %cpt, frame)
    time.sleep(0.01)
    cpt += 1
    if cv2.waitKey(5)&0xFF==27:
        break
cap.release()
cv2.destroyAllWindows()
```

Figure 9 :Used cv2 for max. frames of images



Custom Training

```
%cd {HOME}

!yolo task=detect mode=train model=yolov8s.pt data=/content/datasets/freedomtech/data.yaml epochs=100 imgsz=800 plots=True
```

Figure 10 :Training process for YOLOV8s object dection model

3.5 KEY CHALLENGES

There are a lot of difficulties in creating a system that can identify license plates in both photos and videos, such as the existence of hazy photos and the dearth of reference materials for systems that recognize license plates in movies. The system's ability to correctly recognize number plates can be severely hampered by blurry photos since the features required for recognition may be blurry or unclear. The absence of adequate research and reference materials pertaining to video-based number plate identification systems further complicates the development process because there may be few documented best practices

Despite these difficulties, the development process concentrates on finding creative solutions and giving them significant thought. Preprocessing methods like noise reduction and picture enhancement can be used to increase image clarity prior to number plate identification in order to lessen the impact of fuzzy photographs. To properly train the detection models, efforts are also undertaken to collect and use a variety of datasets that comprise both clear and fuzzy pictures. Additionally, to ensure the system's durability and efficacy in the lack of comprehensive reference papers, the development team conducts thorough experimentation and testing to investigate and improve algorithms designed especially for video-based number plate identification. Developing tactics to tackle these obstacles head-on will continue to lead to the delivery of a dependable and precise number plate identification system

Making sure our system operates in real-time, particularly for video processing, is another significant difficulty. When processing frames from a video stream, we don't want the system to lag or freeze. This implies that we must ensure that we are efficiently utilizing our computational resources and optimizing our methods. In order to correctly recognize the text on a plate once it has been identified, we also need to link our number plate detection with optical character recognition (OCR). This entails managing noise, distortion, and font style variationsall of which can impair OCR performance. Accuracy and speed must be balanced, but with rigorous research, development, and testing.

CHAPTER 4 : TESTING

4.1TESTING STRATEGY

1) Unit Testing : Unit testing is the process of testing separate system modules or components separately. Unit testing in the context of our project confirms that crucial modules like optical character recognition (OCR), character segmentation, and picture preprocessing work as intended. This guarantees that every module functions as planned and combines with other parts in an easy-to-use manner.

2) Integration Testing: Integration testing assesses how various modules interact with one another to make sure everything works as it should. Verifying the connectivity between the modules in charge of live video processing, frame extraction, and real-time number plate recognition is the main goal of integration testing for our system. This testing stage verifies that there are no compatibility problems and that data moves between components appropriately.

3) Testing for functionality: Functionality testing evaluates if the system satisfies the criteria and effectively completes its intended duties. When evaluating the operation of a live video system for number plate identification, several live video feeds with variable illumination, angles, and backdrops are fed into the system. A detailed evaluation is conducted to determine how well the system can detect and recognize number plates in these various settings.

4) Testing for Accuracy:The goal of accuracy testing is to evaluate the system's real-time capacity to accurately recognize and extract characters from number plates. In order to assess the system's recall rates, accuracy, and precision, the identified number plates are compared to ground truth data during this testing phase. To improve the overall accuracy, any false positives or false negatives are also thoroughly examined and dealt with.

5) Evaluation of Performance:Performance testing assesses how well the system responds, scales, and remains stable under various workload scenarios. Performance testing, as it relates to our project, gauges the system's latency, processing speed, and resource use during real-time video analysis. This testing stage ensures seamless functioning in real-world circumstances, optimizes algorithmic efficiency, and helps locate any bottlenecks.

6) Testing for user acceptance (UAT):Verifying the usability, intuitiveness, and user satisfaction of the system is the goal of user acceptance testing. End users who will be using the number plate recognition system in real-world situations, including parking management organizations or law enforcement agencies, may be consulted during this testing phase to provide input. Every problem or recommendation brought up during UAT is fixed to improve the user experience as a whole.

7) Testing for Robustness:Robustness testing evaluates how resilient the system is against mistakes, unforeseen inputs, and hostile circumstances. During this testing phase, the system is put through edge scenarios, such severe weather, dimly lit areas, or obscured license plates, in order to assess its resilience and error-handling skills. Potential vulnerabilities can be found and fixed to help the system function reliably and more robustly in the actual world.

8) Testing for security:Security testing looks at how resilient the system is against security risks such tampering, illegal access, and data leaks. Security testing is essential to ensuring the confidentiality, availability, and integrity of data processed by the number plate recognition system since vehicle registration data is sensitive. To reduce security threats, strategies including intrusion detection, access restrictions, and encryption are assessed.

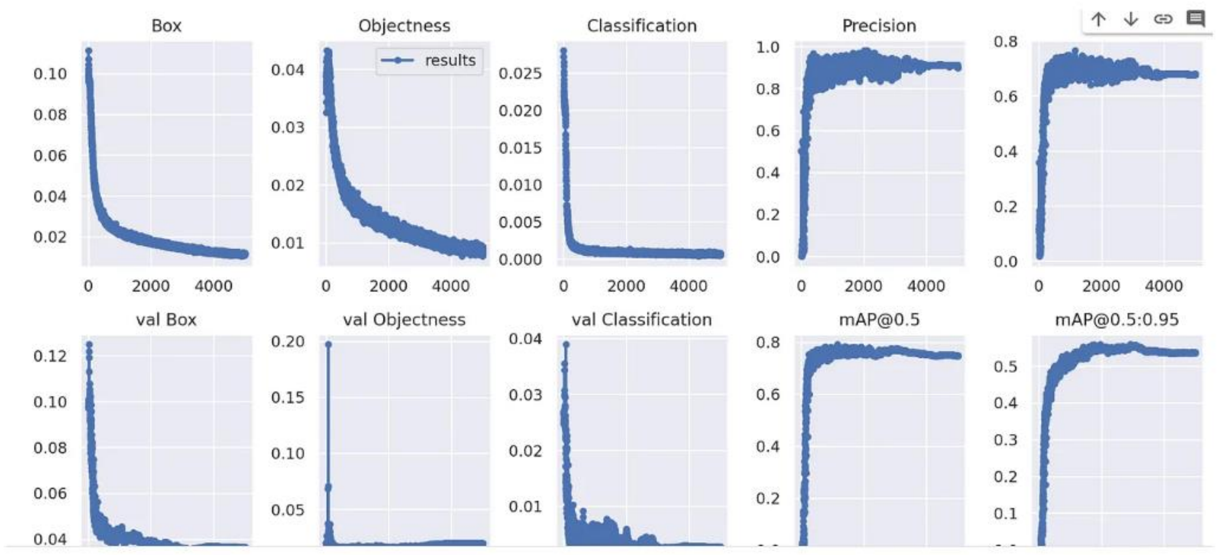


Figure 11: Mean Average

4.2TEST CASES AND OUTCOMES

1) Test Case: Confirm Accuracy of Number Plate Detection .Verify that the system accurately detects and identifies the license plates of the cars in the live feed.
   a) Send a video stream to the system that shows cars with clearly visible license plates.
   b) Check to see if the system correctly extracts and shows the alphanumeric characters for every number plate that is identified.
   c) To ensure accuracy, compare the recognized license plates with the ground truth data.

2) Test Case: Assessing Efficiency in Varying Lighting Situations .Evaluate the system's functionality under different lighting scenarios.
   a) Record live video broadcasts in a variety of lighting scenarios, including day, twilight, and night.
   b) Check to see if the system can detect license plates correctly in all lighting conditions.
   c) For every lighting environment, measure and log the system's recognition accuracy and processing speed.

3) Test Case: Testing Accuracy at Various Vehicle Angles and Speeds.Analyze the accuracy of the system while cars are traveling at various speeds and angles.
    a) Record video streams of moving cars from several angles (front-facing, side-facing), and at different speeds (slow, moderate, rapid).
    b) Make sure the system can reliably identify and locate license plates from cars at any speed or angle.
    c) Verify the system's functionality by contrasting the verified license plates using ground truth information.

4) Test Case: Evaluate Sturdiness Against Invisible Number Plates ,Test the system's capacity to process partially obscured license plates.
    a) Describe situations when dirt, stickers, or colored coverings partly conceal the number plates.
    b) Check to see if the system is still able to identify and decode alphanumeric characters from partially obscured license plates.
    c) Examine the correctness and dependability of the system in such demanding circumstances.

5) Test Case: Assessing Performance in a High-Traffic Area , Analyze how well the system performs in situations with high traffic density.
    a) Record live video broadcasts in high-traffic places, including cities or highways, especially during rush hours.
    b) Make that there is no appreciable performance decrease when numerous cars' number plates are detected and recognized simultaneously by the system.
    c) Check the correctness and speed of the system's processing under

6) Test Case: Verify the Accuracy of the Timestamp and Data Logging ,Check that the time stamping and data recording functions are accurate.
    a) Make that the retrieved number plates are recorded in the system's data recording mechanism, together with the relevant date and time.

b) To ensure accuracy, compare the timestamps and recorded number plates with the real video footage.

c) Verify that the time stamps that were recorded correspond with the moment that each number plate was found and identified.

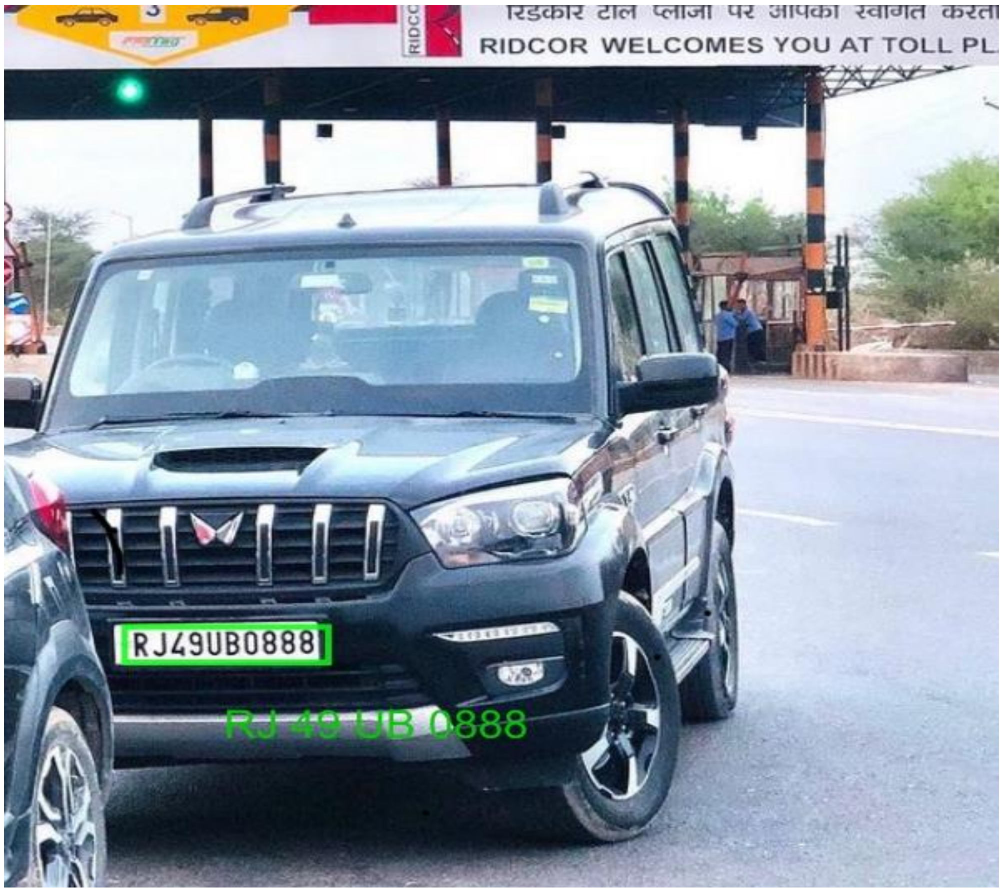

रिडकार टोल प्लाजा पर आपका स्वागत करता
RIDCOR WELCOMES YOU AT TOLL PL
RJ49UB0888
RJ 49 UB 0888

Figure 12 : Testing for image

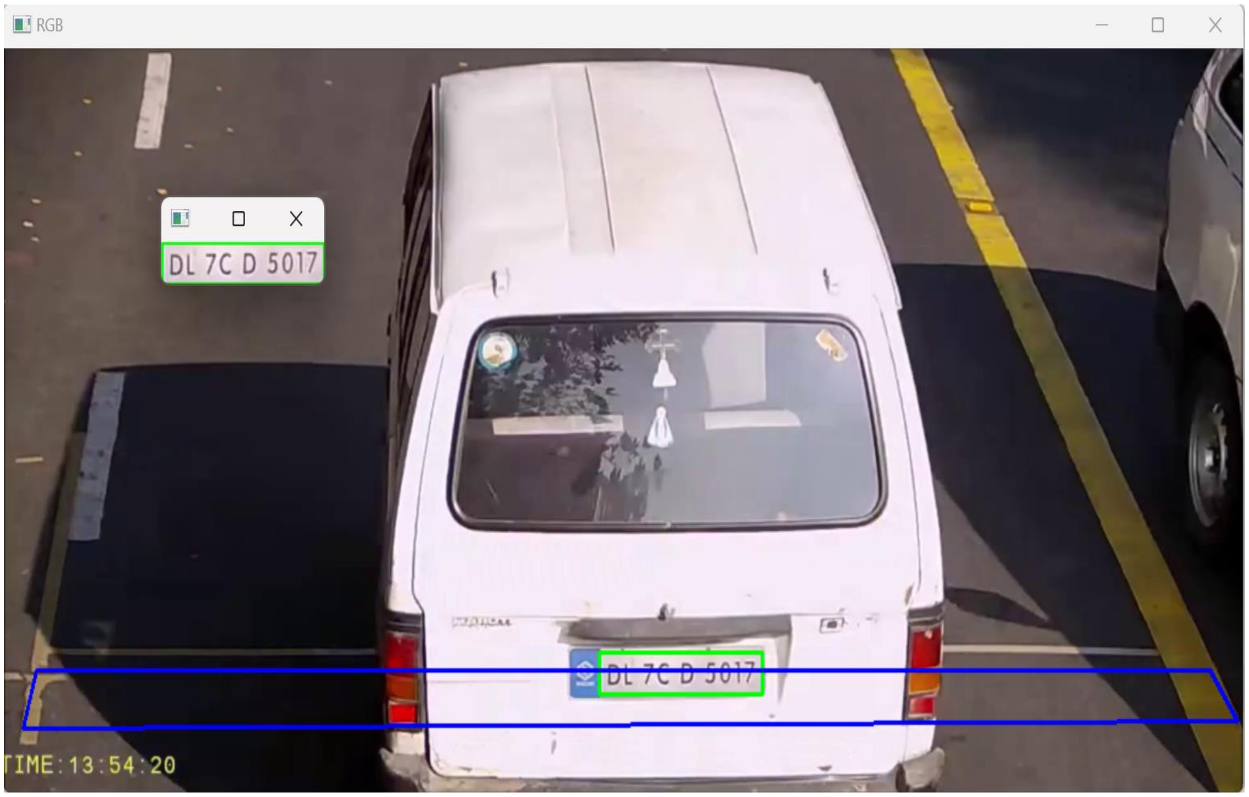


Figure 13 : Testing for video

CHAPTER 5: RESULTS AND EVALUATION

5.1 RESUTLS

Results for image that are shown below :

PART1 – OCR

- First,we install and call on our libraries and dependencies
- Then, using Python, we will read the image in OpenCV.
- We the read the text with in the image and draw a box around it.
- Lastly, overlay the extracted text on top of our picture.

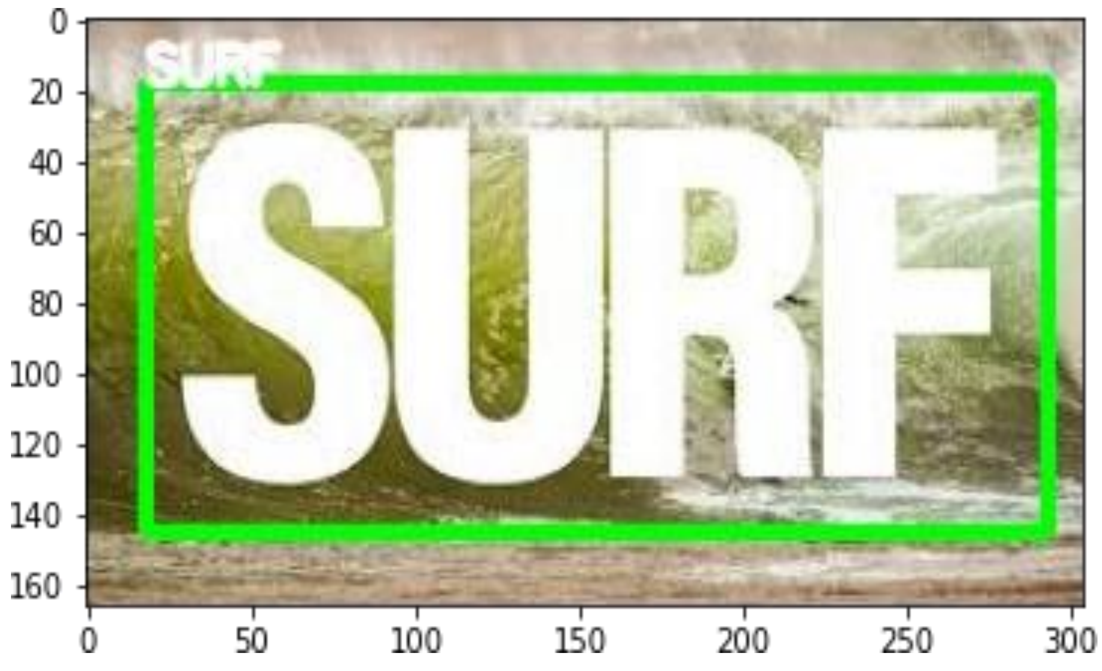


Figure 14 : Object detection

HANDLING MULTIPLE LINES

- To handle more than one lines we use the coordinate point of the bounding rectangle.
- Then we create at to store the words.

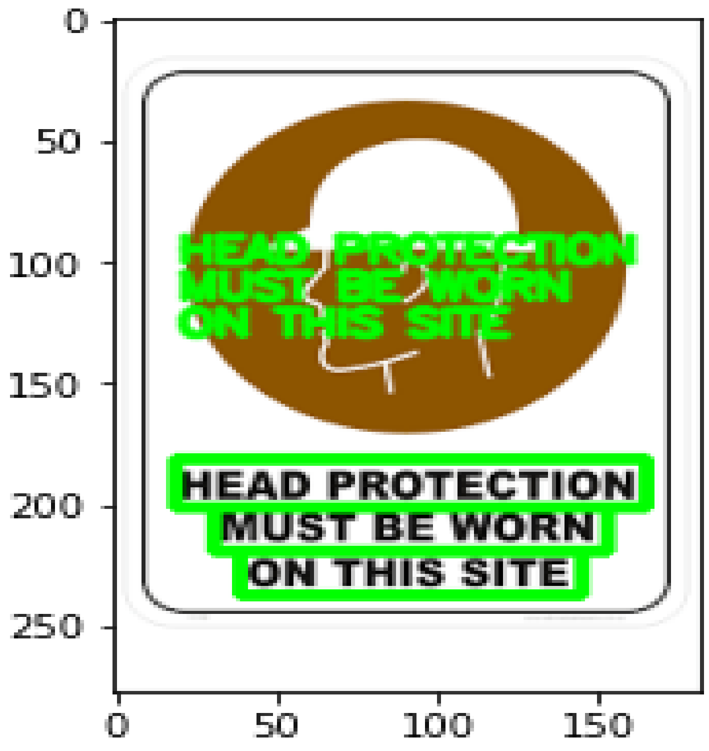


Figure 15 : Character detection

PART 2 – ANPR

- First,we read the picture in OpenCV using python language.
- Then we will gray scale the RGB image to make it fit for the pre-processing.

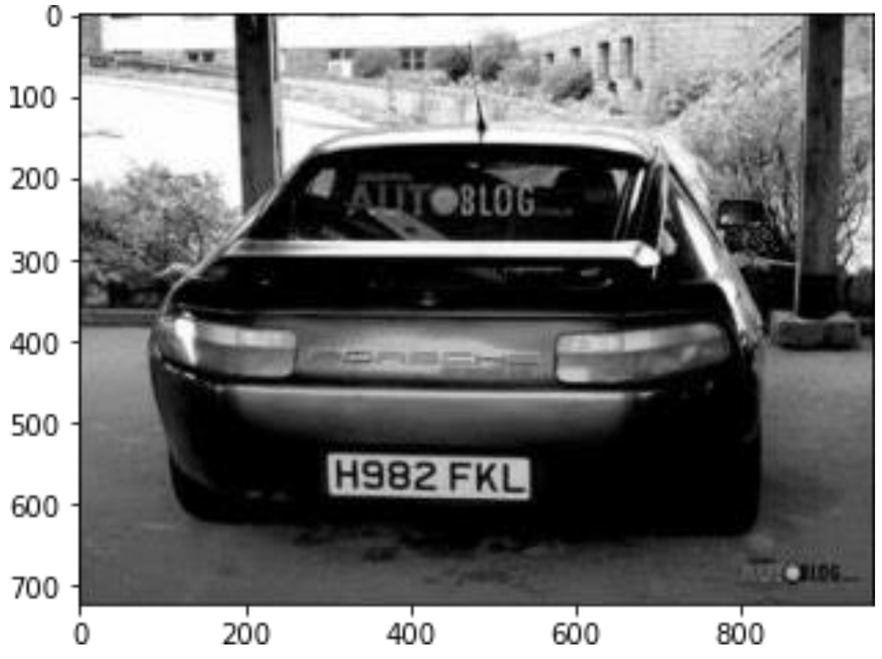


Figure 16: Gray scale Image

Next,we apply the pre-processing techniques, to make the image clearer and fit for detection of edges.

SHARPENING KERNAL

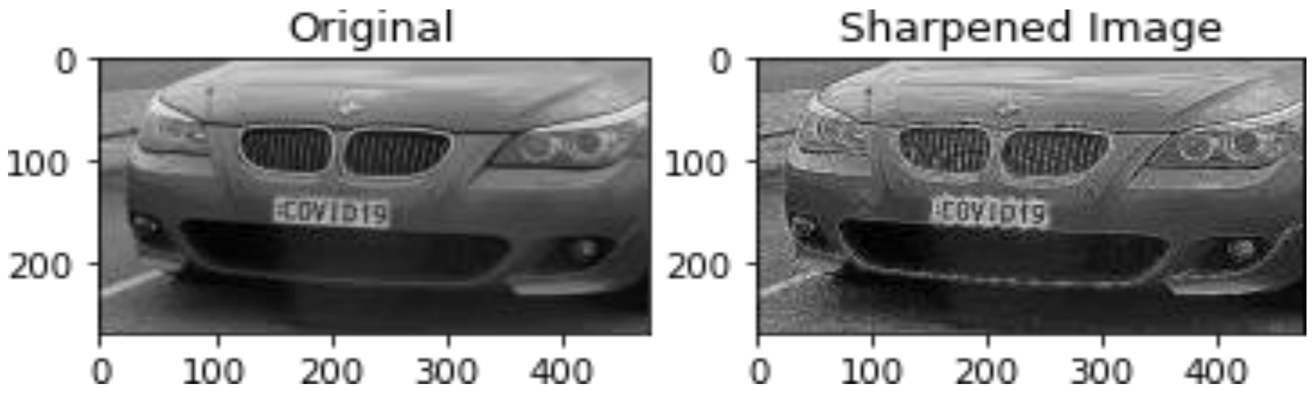


Figure 17: sharpen the image

Then, we will do the edge detection and the contour detection that are going to help us in locating the number plates within an image.

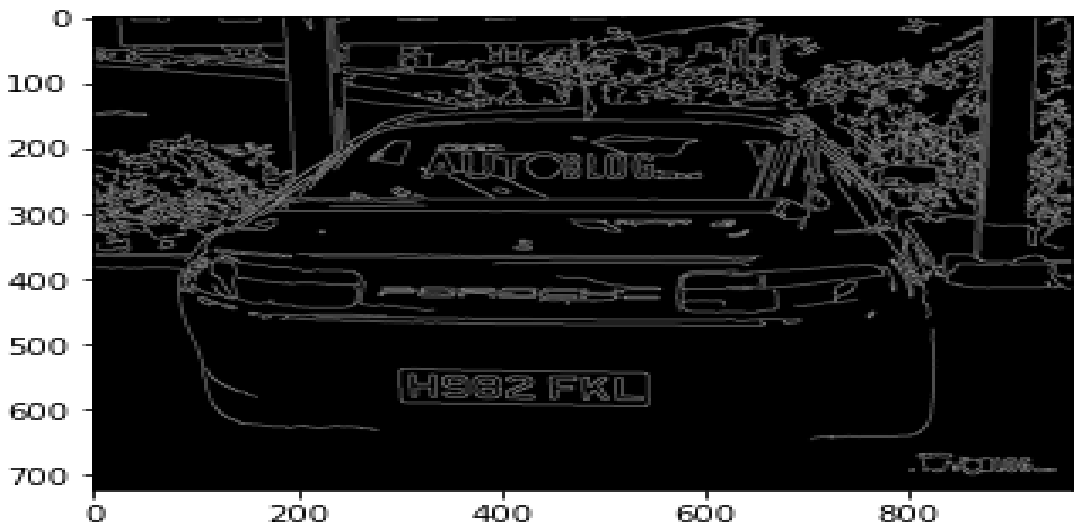


Figure 18: Contour Extraction

Then,we use a Boolean AND with an inverse for masking of plate with in an picture.

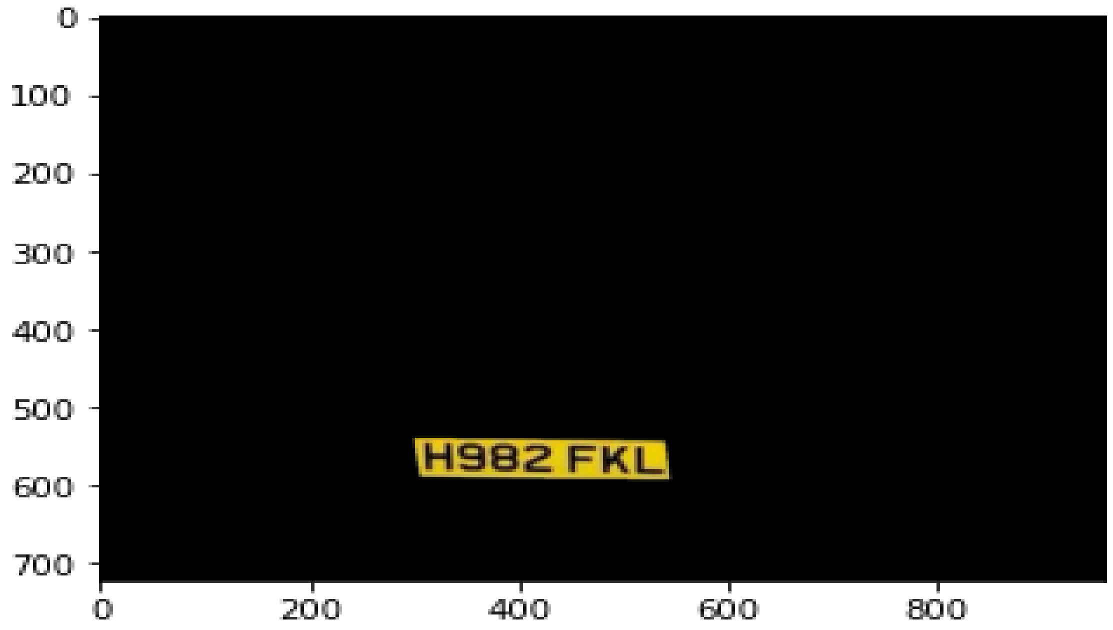


Figure 19: Applying Mask

Lastly, one uses the EasyOCR library to extract the text off the number plate and display it overlaid on   the picture.

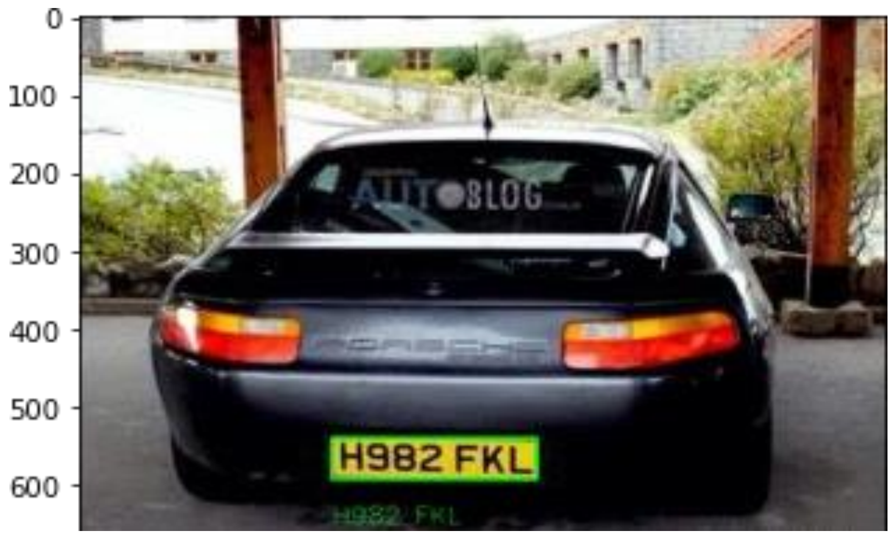
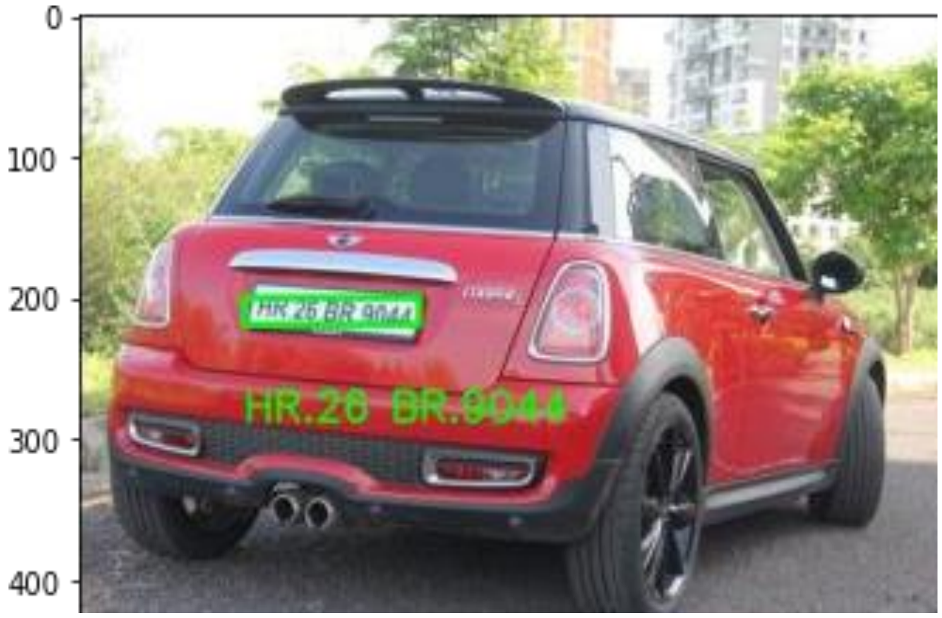


Figure 20: Number Plate Detection



RESULTS: Visualizing The Output

4.3.1 RESULTS

Results for video that are shown below :

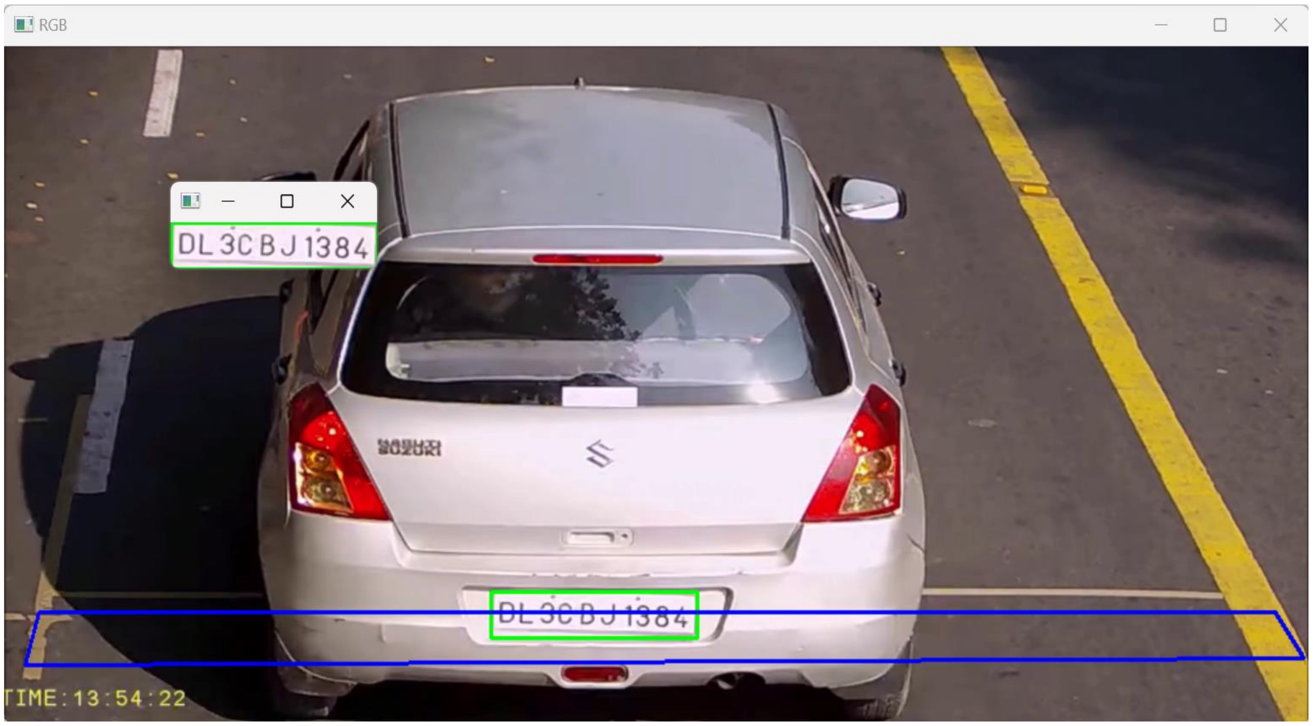


Figure 21: Number Plate Detection using YOLOv8

Using CV2, Capture Images from Video:

To read the video file in Python, use the OpenCV (cv2) package.Take frames out of the movie at predetermined intervals or every nth frame. The retrieved frames and photos should be saved to a specific folder.

Plates bearing numbers with labeling:

Apply a hand label to the number plates in the pictures using a programs , Enclose the license plates in bounding boxes and designate them with the label "number_plate". The annotations can be saved in XML or any other format that works. With labeling, users can easily annotate photos by marking and labeling particular things of interest.
Human input is required for the manual labeling of number plates, which guarantees precise identification and delineation of license plates.

Build a dataset

Combine the folder's unlabeled and labeled photos (including number plate annotations).Put these pictures in a training-ready dataset structure. Combining the annotated number plate-containing labeled photos with the unlabeled images facilitates the development of an extensive dataset. Partitioning these pictures into discrete subsets, such training, validation, and testing sets, is necessary to organize them into a structured manner appropriate for training.
Training a model that can effectively generalize to new cases requires striking a balance between labeled and unlabeled data. The cornerstone for developing reliable and accurate machine learning models for license plate identification is a well-curated dataset.

YOLOv8-Trained Dataset:

The cutting-edge object identification system known for its quickness and precision is called YOLOv8. In order to reduce detection errors, the model must be trained by subjecting it to the labeled dataset and repeatedly modifying its parameters. To improve the model's performance, a number of techniques are used for fine-tuning, including as data augmentation, transfer learning from pre-trained models, and hyperparameter tweaking.Monitoring the model's development and pinpointing areas for improvement is made easier by assessing its performance using validationdata.

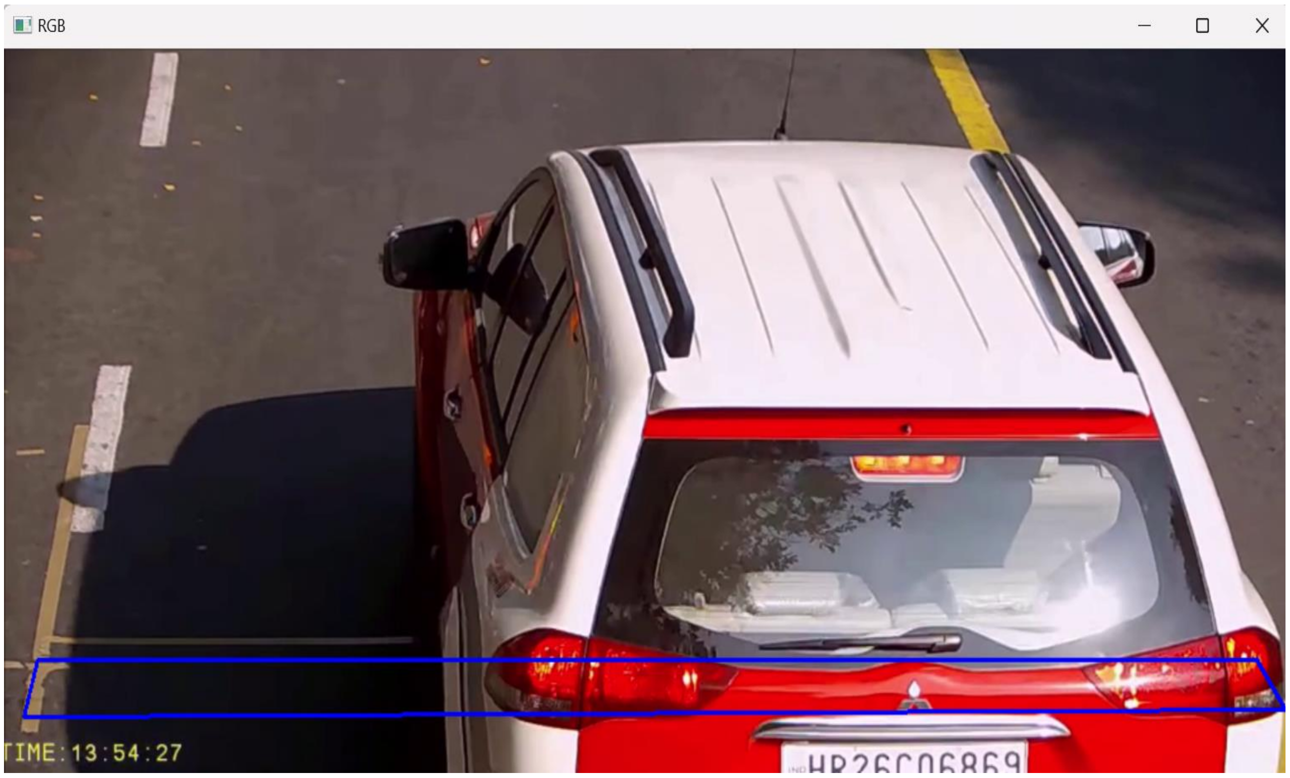


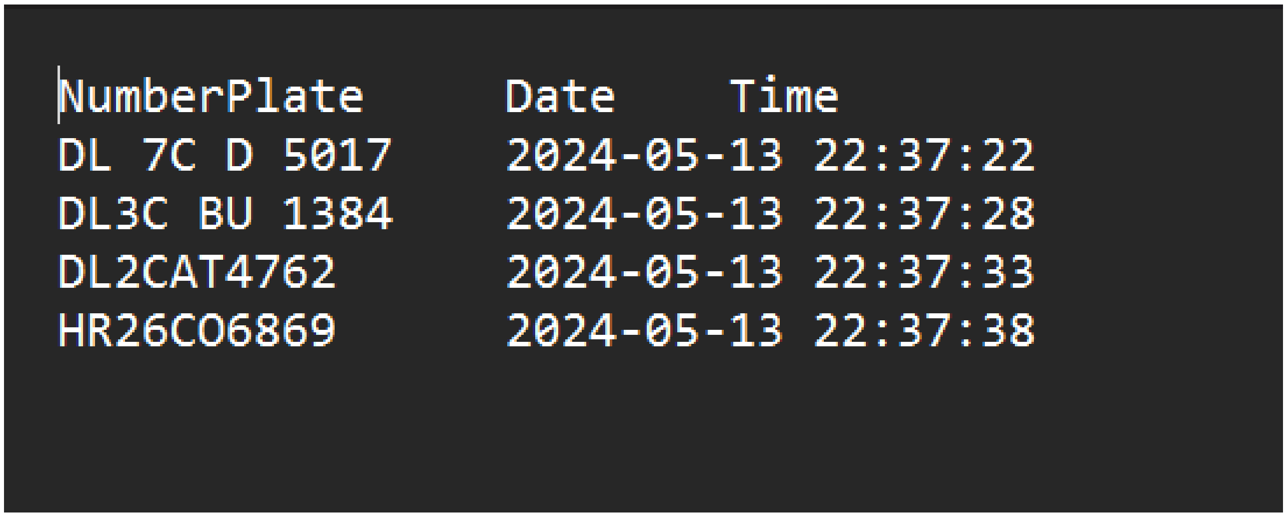
Figure 22 : Showing image of entering number plate in marker area

Pytesseract for Text Recognition:

Utilize the YOLOv8 model to identify number plate-containing areas in fresh photos once it has been trained. These areas should be extracted and preprocessed (e.g., resized, contrast enhanced). Employ Pytesseract, a Python wrapper for Tesseract-OCR, an engine developed by Google, to carry out optical character recognition (OCR) on the number plate areas that were retrieved. The text (license plate numbers) may be extracted and recognized from the number plate areas.

Take note of the license plate and timestamp.

After processing each image, obtain the timestamp by using the cv2 and datetime libraries. Compare the associated timestamp with the identified license plate. Save this data as a text document or any other format you'd want (CSV, JSON, etc.). The timestamp, the identified license plate number, and maybe the path to the associated picture file should be included with each item in the file.



```
NumberPlate       Date     Time
DL 7C D 5017    2024-05-13 22:37:22
DL3C BU 1384    2024-05-13 22:37:28
DL2CAT4762      2024-05-13 22:37:33
HR26CO6869      2024-05-13 22:37:38
```

Figure 23 : Showing dataset of vehicles including number plate and datetime

CHAPTER 6: CONCLUSION & FUTURE SCOPE

5.1 CONCLUSION

The creation and testing of the live video number plate recognition system have yielded insightful information on the prospective uses, constraints, and capabilities of this technology. After extensive testing and assessment, a number of important conclusions have been drawn that highlight the system's advantages as well as its short comings.

KEY FINDINGS:

Accuracy and Reliability: The system shows excellent accuracy when it comes to identifying and detecting number plates from live video feeds, even in well-lit environments with clearly visible plates.

Robustness: Although the system is accurate, there are times when it becomes vulnerable to problems like obscured license plates or bad weather. Additional improvements are required to strengthen the system's resilience in managing these kinds of situations.

Performance: Under moderate traffic density conditions, the system performs satisfactorily in processing live video feeds and real-time number plate identification. Under heavy traffic loads, however, scalability and efficiency could become issues.

Data Logging and Time stamping: The extracted number plates are successfully recorded by the data logging system that has been put into place, together with the appropriate timestamps. This feature improves the system's usefulness for programs that precise recording of vehicle movements.

CONSTRAINTS:

Occlusion and Unfavorable Conditions: When number plates are partially obscured or in unfavorable weather, the system's performance suffers. More investigation and development work is necessary to address these constraints.

Resource Intensiveness: The system's scalability may be limited by the computing resources needed for real-time processing of live video feeds, especially in locations with limited resources.

Reliance on Outside Sources: The quality of the recorded video streams, camera angles, and illumination all have a significant impact on the system's accuracy and dependability. Changes in these variables might affect how well the system performs.

CONTRIBUTIONS TO THE FIELD:

Development of Automated Surveillance Systems: By facilitating real-time number plate detection and recognition, the created system advances the field of automated surveillance systems and strengthens law enforcement and security capacities.

Possibilities for Traffic Management: The system's capacity to record vehicle movements and timestamps makes it possible to use it for parking enforcement, traffic management, and toll collecting systems, all of which improve resource allocation and traffic flow.

Investigation and Originality: The problems and prospects in the field of computer vision and machine learning for intelligent transportation systems have been better understood because to this study. These discoveries open up new avenues for investigation and creativity in the future with the goal of surpassing current constraints and elevating the bar for number plate recognition technology.

In conclusion, even if the created number plate recognition system has promise in a variety of applications, more development and optimization are required to resolve current issues and reach the full potential of the system in practical applications.

5.2 FUTURE SCOPE

The live video-based number plate recognition system offers a solid platform for additional research and development. In order to overcome current constraints and open up new possibilities in the field of automated surveillance and intelligent transportation systems, a number of potential development and growth paths might be explored.

Improved Robustness: Subsequent versions of the system might concentrate on making it more resilient to things like obscured license plates, bad weather, and different lighting conditions. Adding machine learning algorithms and sophisticated image processing techniques can improve the system's capacity to recognize and detect number plates in difficult situations.

Real-time Analytics: Adding real-time analytics functionalities to the system can help provide more in-depth understanding of traffic patterns and vehicle movements. Through the examination of information obtained from the number plate recognition system, decision-makers may make well-informed choices about traffic control, law enforcement, and urban development.

Scalability and Efficiency: To support high traffic volumes and extensive deployments, efforts might be focused on maximizing the system's computing efficiency and scalability. Performance may be increased by utilizing parallel processing methods and distributed computing systems.

Integration with Smart Cities Initiatives: The number plate recognition system can be easily integrated with the current infrastructure for public safety, parking enforcement, and traffic management by integrating it with smart cities initiatives. Cities may optimize parking allocation, develop dynamic traffic routing, and improve overall urban mobility by utilizing data from the system.

Multimodal Integration: In order to provide complete vehicle tracking and identification, future versions of the system may investigate integration with additional sensor modalities including lidar, radar, and GPS. Integrating information from several sensors can improve the system's accuracy and dependability while offering a comprehensive picture of traffic movement.

Adoption of Edge Computing: Distributed processing and analysis of video streams may be facilitated nearer to the source by utilizing edge computing technology, which lowers latency and bandwidth needs. The number plate recognition system can operate autonomously in remote or resource-constrained contexts and improve real-time responsiveness through edge-based deployment.

Privacy and Security Considerations: It is critical to address privacy and security issues related to the gathering and storing of car registration data. Strong encryption methods, access restrictions, and anonymization tools should be incorporated into future advancements to protect private data and guarantee legal compliance.

To sum up, the number plate recognition system will have a wide range of applications in the future, involving developments in infrastructure, technology, and regulatory frameworks. The system has the potential to transform automated surveillance, improve urban mobility, and help realize safer, smarter cities if it keeps innovating and working together across multidisciplinary disciplines.

REFERENCES

[1] H. Shi, D. Zhao,"License Plate Recognition System Based on Improved YOLOv5 and GRU,"IEEEAccess, pp. 1783–1791, Jan.2023.

[2] Shan Luo, Jihong Liu,"Research on Car License Plate Recognition Based on Improved YOLOv5m and LPRNet,"IEEEAccess, pp. 93692 - 93700, Sep.2022.

[3] Parneet Kaur1,Yogesh Kumar1,Shakeel Ahmed,"Automatic License Plate Recognition System for Vehicles Using a CNN,"TechPress, pp. 2105-2345, Feb.2021.

[4] Yadavendra Atul Sakharkar,Kakelli Anil Kumar,"A Reinforcement Learning-based OffloadDecision Model (RL-OLD) for Vehicle Number Plate Detection,"I. J. Engineering and Manufacturing, pp. 11-18, Dec.2021.

[5] Shahnaj Parvin, LitonJude Rozario,"Vehicle Number Plate Detectionand Recognition Techniques,"Advances in Science, Technology and Engineering Systems Journal, pp.423- 438, March.2021.

[6] A Roopa Devi, Ch Yamini Saraswath,"Recognition of Vehicle Number Plate Using Matlab",JournalofUniversityofShanghai for Science and Technology, pp .363-370, Feb- 2021.

[7] V Gnanaprakash,"Automatic number plate recognition using deep learning",ICCSSS 2020 ,pp .757-899, Sep.2021.

[8] J M S V Ravi Kuma,"Automatic Vehicle Number Plate Recognition System Using Machine Learning",CHSN 2020 , pp .366-401.March-2021

[9] YONGJIEZOU,YONGJUNZHANG,"ARobustLicensePlateRecognitionModel Based on Bi-LSTM,"IEEEAccess, pp. 211631–211641, Dec.2020

[10] JITHMISHASHIRANGANA,HESHANPADMASIRI,"AutomatedLicensePlate Recognition,"IEEEAccess,pp.11203–11225,Dec.2020.

[11] R.Laroca,E.Severo,L.A.Zanlorensi,L.S.Oliveira,G.R.Goncalves,W .R. Schwartz, D. Menotti, "A Robust Real-Time Automatic License Plate RecognitionBasedontheYOLODetector,"ProceedingsoftheInternational ,"IEEEAccess,pp.113–115,Dec.2021

[12] NayerehZagharietal,"Theimprovementinobstacledetectioninau-tonomousvehiclesusingYOLOnon-maximumsuppressionfuzzyalgorithm,"TheJournal . ofSupercomputing,2021,77(11):1-26.

[13] Y.Cai,T. Luanetal,"YOLOv4-5D:AnEffectiveandEfficientObject Detector for Autonomous Driving," IEEE Transactions on Instrumentation an Measurement,2021, 70, 4503613: 1-13.

[14] MadasamyKaliappanetal,"OSDDY:embeddedsystem-basedobjectsurveillance detectionsystemwithsmalldrone usingdeepYOLO,"EURASIPJournalonImage and Video Processing, 2021, 2021(1).

[15] Tumas Paulius and Serackis Art¯uras and Nowosielski Adam, "Augmenta-tion of Severe Weather Impact to Far-Infrared Sensor Images to Improve Pedestrian Detection System," Electronics, 2021, 10(8) : 934-934.

[16] Li G, Ji Z, Qu X, et al,"Cross-Domain Object Detection for Autonomous Driving: A Stepwise Domain Adaptative YOLO Approach," IEEE Trans-actions on Intelligent Vehicles, 2022.

[17] Zaremba W,Sutskever I,Vinyals O, "Recurrent Neural Network Regular-ization," Eprint Arxiv, 2014.

[18] Palaiahnakote Shivakumara et al, “CNN-RNN based method for license plate recognition,” CAAI Transactions on Intelligence Technology, 2018,3(3) : 169-175.

[19] Huiyu Wang,Y ukun Zhu,Bradley Green,Hartwig Adam,Alan Yuille,Liang-Chieh Chen, “Axial-deeplab: Stand-alone axial-attention for panoptic segmentation,” rXiv preprint arXiv:2003, 2020.

[20] TangYushun,ZhangShengguo,et al, “Color-based license plate recognition research,” modern computer, 2020, 32, 63-66, 71.