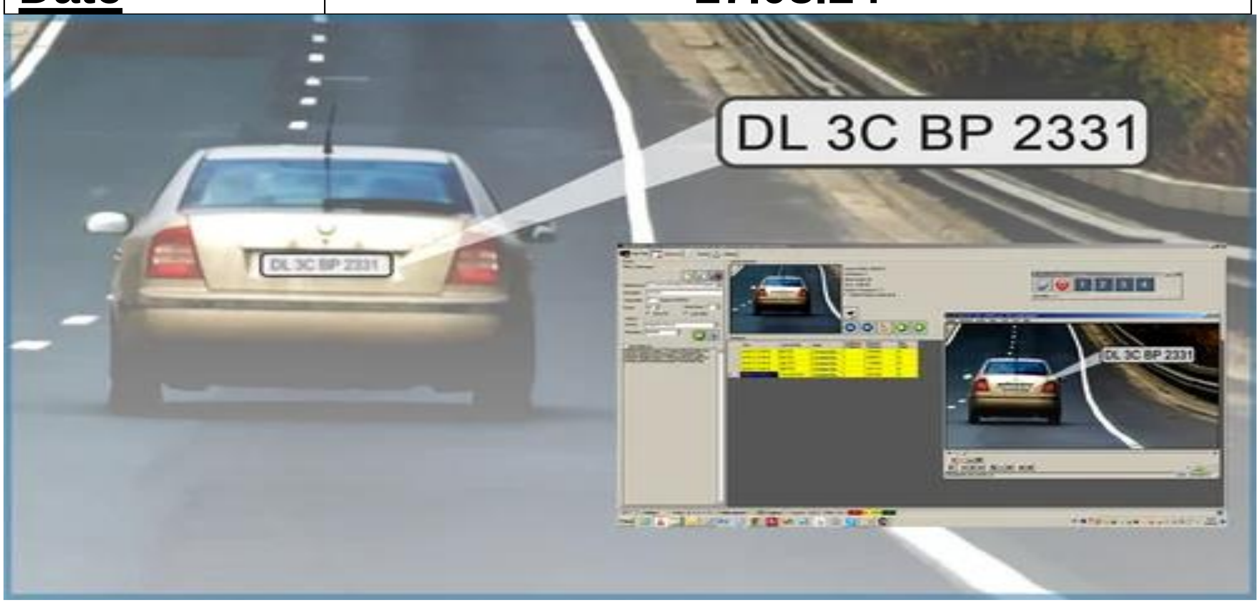
| | |
|---|---|
| **Project Title** | **Automatic Number plate recognition using computer vision and OCR** |
| **Student Name** | **HIMANSHU RAJESHKUMAR MAURYA** |
| **Course Name** | **BSC-TYDS** |
| **Instructor Name** | **PROF. MITHILESH CHAUHAN** |
| **Date** | **27.08.24** |



## INDEX

## CHAPTER 1 : Introduction /Objective

The primary objective of this project is to develop an automated system capable of detecting and recognizing vehicle number plates from video feeds. The system aims to accurately extract alphanumeric characters from the number plates to be used for various applications such as traffic monitoring, toll collection, and law enforcement.

Automatic Number Plate Recognition has become part of our lives and promises to stay in future, integrable with proposed transportation technologies. The concept of Autonomous Vehicles is offering many possibilities of changing fundamental transportation systems. ANPR (Automated number plate recognition) technology is already contributing towards intelligent transportation systems and is eliminating the need for human intervention. It is no longer just the camera on the roadside or at the barrier to the car park. It has become over the years mobile, first being deployed in vehicles, but now more recently with the advent of smart phone technology, many ANPR systems have become handheld too. Due to lower provisioning costs, ANPR is often a choice in the toll and parking lot businesses. The main reason is that the ANPR system recognizes the registered number plate with no additional transponder requirements, as compared to the Ultra High Frequency—Radio Frequency Identification (UHF-RFID) systems. The rapid urbanization of countries is a great advancement in our modern world. People migrate away from rural areas and choose to live in cities mostly. Local governments often fail to recognize the present and potential mobility needs of residents and visitors as traffic rises in these areas. ANPR is being increasingly used to examine the free flow of traffic, facilitating intelligent transportation .

Not only can modern ANPR cameras read plates, but they can provide useful additional information such as counting, direction, groups of vehicles and their speed. The ability to detect and read large volumes of fast moving vehicles has meant that ANPR technology has found its way into many aspects of today’s digital landscape. Whilst ANPR technology can come in many different packages, they all perform the same basic function which is to provide a highly accurate system of reading a vehicle without human intervention. It is utilized in very diverse applications such as access control, parking management, tolling, user billing, delivery tracking, traffic management, policing and security services, customer services and directions, the red light and lane enforcement, queue length estimation, and many other services .

Number Plate Recognition involves acquisition of number plate images from the intended scene, using a camera. Either still images or a photographic video is captured and further processed by a series of image processing based recognition algorithms to attain an alpha-numeric conversion of the captured images into a text entry. After obtaining a good quality image of the scene/vehicle, then the core dependence of any

ANPR system is on the robustness of its algorithms. These algorithms need a very careful consideration and require thousands of lines of software coding to get desired results and cover all system complexities. As a whole, a series of primary algorithms are necessary for smart vehicle technologies and ANPR to be effective.

A typical ANPR system goes through the general process of image acquisition (input to the system), number plate extraction (NPE), character segmentation (CS) and character recognition (CR) (as output from the system). After successful recognition of the vehicle the data can be accessed and used for post processing operations as required. The vehicle's data is sent to the connected back office system software which is the central repository to all data along with tools to support data analysis, queries and reporting accordingly. This data collected can be utilized for several other intelligent transportation applications since ANPR systems not just visually capture the vehicle images but also record the metadata in their central repository. This can potentially include vehicle recognition through date and time stamping as well as exact location, whilst storing a comprehensive database of traffic movement. This data can be helpful in modeling different transport systems and their analysis.

The image taken from the scene may experience some complexities depending upon the type of camera used, its resolution, lightning/illumination aids, the mounting position, area/lanes coverage capability, complex scenes, shutter speed and other environmental and system constraints.

When a vehicle is detected in the scene/image, the system uses plate localization functions to extract the license plate from the vehicle image, a process commonly termed as Number Plate Extraction. Characters on the extracted number plate are then segmented prior to the recognition process. Character segmentation is an algorithm that locates the alphanumeric characters on a number plate. The segmented characters are then translated into an alpha numeric text entry using the optical character recognition (OCR) techniques. For character recognition, algorithms such as template matching or neural network classifiers are used. The performance of an ANPR system relies on the effectiveness of each individual stage. A parameter used to quantify the whole process is the performance-rate or success-rate, which is the ratio of the number of number-plates successfully recognized to the total number of input images taken. The performance rate involves all the three stages of the recognition process, number plate extraction, segmentation and character recognition.

The ANPR system collects the primary form of the information from ANPR software including the images and its associated metadata. It provides the transport system with automation and security features. Its integration in ITS makes it possible to automate the system by providing services in toll collections, traffic analysis, improving law enforcement’s and building a comprehensive database of traffic movements. Integrating ANPR with Information Communication Technology (ICT) tools is another useful feature of the technology. The data from ANPR systems can be well utilized for modeling and implementation of various aspects of transport systems such as to model Passenger Mobility systems, traffic flow analysis and road network control strategies using Network Fundamental Diagram (NFD) models , in vehicle routing choice model to decide on Route and Path Choices of Freight Vehicles and travel demand patterns through Floating Car Data (FCD) .”

There are different terminologies used for ANPR systems:

Number plate Recognition (NPR)
Automatic License Plate Recognition (ALPR)
License Plate Recognition (LPR)
License Plate Recognition (LPR)
Automatic Vehicle Identification (AVI)
Car Plate Recognition (CPR)

## CHAPTER 2 : Problem Statement

There are a number of possible difficulties that the software must be able to cope with. These include:

- Manual monitoring of vehicle registration numbers is a time-consuming and error-prone process.
- Poor file resolution, usually because the plate is too far away but sometimes resulting from the use of a low-quality camera
- Blurry images, particularly motion blur
- Poor lighting and low contrast due to overexposure, reflection or shadows
- An object obscuring (part of) the plate, quite often a tow bar, or dirt on the plate
- Read license plates that are different at the front and the back because of towed trailers, campers, etc.
- Vehicle lane change in the camera's angle of view during license plate reading
- A different font, popular for vanity plates (some countries do not allow such plates, eliminating the problem)
- Circumvention techniques
- Lack of coordination between countries or states. Two cars from different countries or states can have the same number but different design of the plate.
- This project addresses the need for an efficient and automated system that can accurately detect and recognize vehicle number plates, thereby reducing human intervention and increasing the accuracy and speed of processing vehicle information.
- Number plate detection automates the identification of vehicles, eliminating the need for manual inspection and data entry.
- This is especially useful at toll booths, parking lots, and security checkpoints.
- The system can operate continuously without human intervention, ensuring that vehicles are monitored and recorded at all times.
- By automatically detecting and recording number plates, traffic authorities can monitor vehicle flow, identify violators, and manage traffic more efficiently without relying on human traffic officers.
- Automating tasks such as issuing tickets for violations, toll collection, and vehicle registration reduces the administrative load on staff.
- In restricted areas, number plate detection can automatically allow or deny access based on pre-set permissions, reducing the need for security personnel to manually verify each vehicle.
- Cashless and Fast Transactions: Automated number plate detection allows for seamless toll collection and parking payments, reducing the need for toll booth operators or parking attendants.

- Real-Time Data Processing: The system can instantly process and log vehicle entries and exits, reducing the need for manual record-keeping and increasing the speed of service.
- Automated Data Logging: The system can continuously collect data on vehicle movements, freeing up humans from the task of manually recording and analyzing this information.
- Minimization of Human Error: Automated systems are less prone to the mistakes that can occur with manual data entry and vehicle identification, leading to more accurate and reliable outcomes.
- Focus on Critical Tasks: By automating routine tasks, human resources can be reallocated to more critical and complex activities that require judgment, creativity, or specialized skills.

While some of these problems can be corrected within the software, it is primarily left to the hardware side of the system to work out solutions to these difficulties. Increasing the height of the camera may avoid problems with objects (such as other vehicles) obscuring the plate but introduces and increases other problems, such as adjusting for the increased skew of the plate.

On some cars, tow bars may obscure one or two characters of the license plate. Bikes on bike racks can also obscure the number plate, though in some countries and jurisdictions, such as Victoria, Australia, "bike plates" are supposed to be fitted. Some small-scale systems allow for some errors in the license plate. When used for giving specific vehicles access to a barricaded area, the decision may be made to have an acceptable error rate of one character. This is because the likelihood of an unauthorized car having such a similar license plate is seen as quite small. However, this level of inaccuracy would not be acceptable in most applications of an ANPR system.

# CHAPTER 3 : Data Description & Collection

- Video Dataset: The dataset consists of videos of vehicles captured from different angles, under various lighting conditions, and with different levels of noise.
- Annotations: The dataset includes labeled data indicating the location of the number plate within each image and the corresponding alphanumeric characters.
- Data Variability: The dataset includes images with various types of number plates, fonts, and sizes, as well as different backgrounds and obstructions.
- Gathering a diverse dataset of vehicle images and videos with annotated number plates.

## CHAPTER 4 : About Visual Studio

**Visual Studio Code**, also commonly referred to as **VS Code**, is a source-code editor developed by Microsoft for Windows, Linux, macOS and web browsers. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded version control with Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add functionality.

In the Stack Overflow 2023 Developer Survey, Visual Studio Code was ranked the most popular developer environment tool among 86,544 respondents, with 73.71% reporting that they use it.

Visual Studio Code was first announced on April 29, 2015 by Microsoft at the 2015 Build conference. A preview build was released shortly thereafter.

On November 18, 2015, the project "Visual Studio Code - Open Source" (also known as "Code - OSS"), on which Visual Studio Code is based, was released under the open-source MIT License and made available on GitHub. Extension support was also announced. On April 14, 2016, Visual Studio Code graduated from the public preview stage and was released to the web.

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including C, C#, C++, Fortran, Go, Java, JavaScript, Node.js, Python, Rust, and Julia. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called "Visual Studio Online" and "Visual Studio Team Services").

The downloadable version of Visual Studio Code is built on the Electron framework, which is used to develop Node.js web applications that run on the Blink layout engine. Visual Studio Code for the Web is a browser-based version of the editor that can be used to edit both local files and remote repositories (on GitHub and Microsoft Azure) without installing the full program. It is officially supported and hosted by Microsoft and can be accessed .

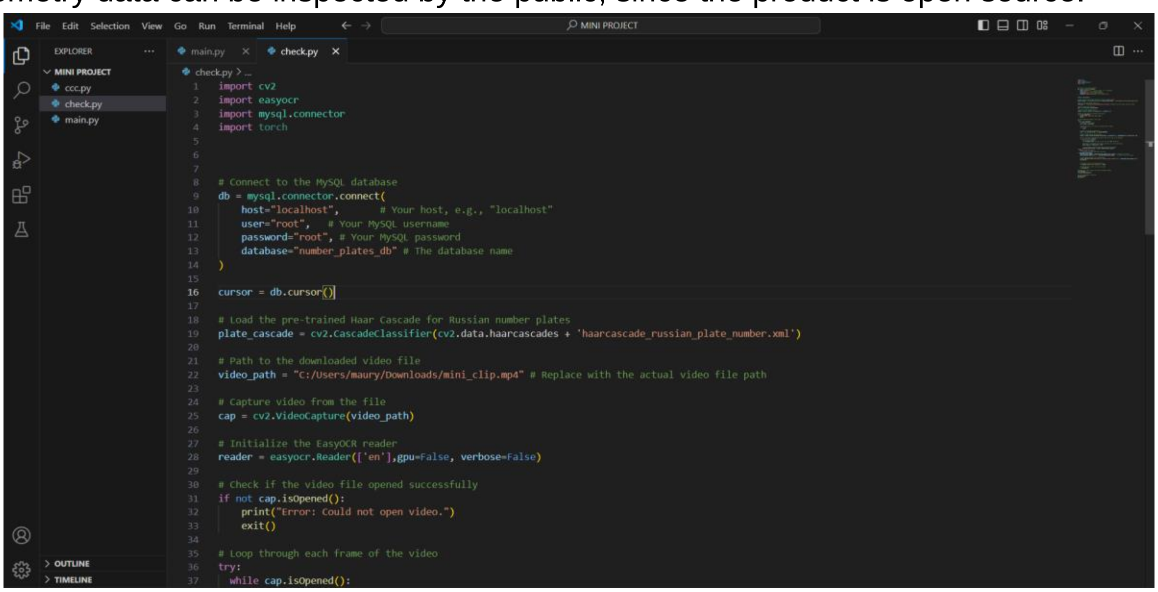
Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace.

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports many programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette.The command palette is able to execute virtually every feature the graphical interface supports, making it very keyboard-accessible.

Visual Studio Code can be extended via extensions, available through a central repository. This includes additions to the editor  and language support.A notable feature is the ability to create extensions that add support for new languages, themes, debuggers, time travel debuggers, perform static code analysis, and add code linters using the Language Server Protocol.

Source control is a built-in feature of Visual Studio Code. It has a dedicated tab inside of the menu bar where users can access version control settings and view changes made to the current project. To use the feature, Visual Studio Code must be linked to any supported version control system (Git, Apache Subversion, Perforce, etc.). This allows users to create repositories as well as to make push and pull requests directly from the Visual Studio Code program.

Visual Studio Code collects usage data and sends it to Microsoft to help improve the product. This telemetry feature can be disabled. The information contained in this telemetry data can be inspected by the public, since the product is open source.




Interface of Visual Studio

# CHAPTER 5 : Methodology

Successful extraction of the number plate from the image/video is initially the most important and critical stage for ANPR systems. The extraction rate is the rate of successfully extracted plates to the total number of input images or vehicles detected from the scene. Typically, a single camera is installed on each lane and some advanced cameras may allow multiple lane support given their high resolutions. Multiple lanes installation requires multiple readers to identify vehicle number plates and more hardware with higher costs to service and maintain. Real time scenarios may face multiple challenges. For instance, the camera installed at a fixed position may acquire images of the vehicle with tilted or skewed number plate characters. It is possible that the number plate is obscured with dirt or broken or located at a position that is out of sight to the camera (since different types of vehicles have their number plates affixed at different positions of the vehicle body). Environmental factors, light, motion blur, reflections, fog, and other similar conditions make it challenging for the system to extract the number plate efficiently. Algorithms using geometrical features for extracting the rectangular shaped number plates may have issues if there are multiple similar shapes drawn/pasted over the car body. Along with rectangular shaped features, further algorithms must be used to eliminate the unwanted regions.

The algorithms need to be robust to differentiate between the number plate and other objects in the image frame. Researchers have used various features for the extraction of number plates. A brief study of these feature extraction algorithms is presented in the following section.

NP Extraction Using Edge Information :

The number plate typically has a known aspect ratio and a rectangular shape. In , the images were first scaled to a fixed aspect ratio. The authors evaluated and put to test various algorithms from the past proposed research in and compared the results by implementing it for their own dataset. One of the number plate extraction methods they evaluated is based on the vertical edge information. It detects the vertical edges using the Sobel operator. The number plate is localized by comparing the preset minimum and maximum lengths with that of the extracted edges and removing the unwanted ones. The total extraction rate for 141 images is 65.25% which is lower than the 99.99% originally reported in. Vertical and horizontal edge histogram information is used for number plate extraction. Testing 50 images of various fonts and light conditions resulted in 90% extraction accuracy.

Edge detection algorithms are commonly used to extract number plates by finding all the rectangles in the acquired images . Mostly, the car body and number plate area have a clear color transition. The differentiation between the two is done by identifying the edges using edge detection filters or algorithms. Sobel filter, a simple algorithm, has been used for the successful edge extraction as presented in. Edges are detected by performing vertical edge detection to extract the vertical lines and horizontal edge detection to get the horizontal lines or simultaneous use of both to extract a complete rectangular shape. The number plate can be detected by using the geometric attributes by locating the rectangle lines. Various edge detection filters Sobel, Canny, Gabor & Log-Gabor filters for ANPR systems are compared in.

Gabor filters are considered the best choice for structure recognition as they show exceptionally good results for excluding clamor/noise while saving edges .

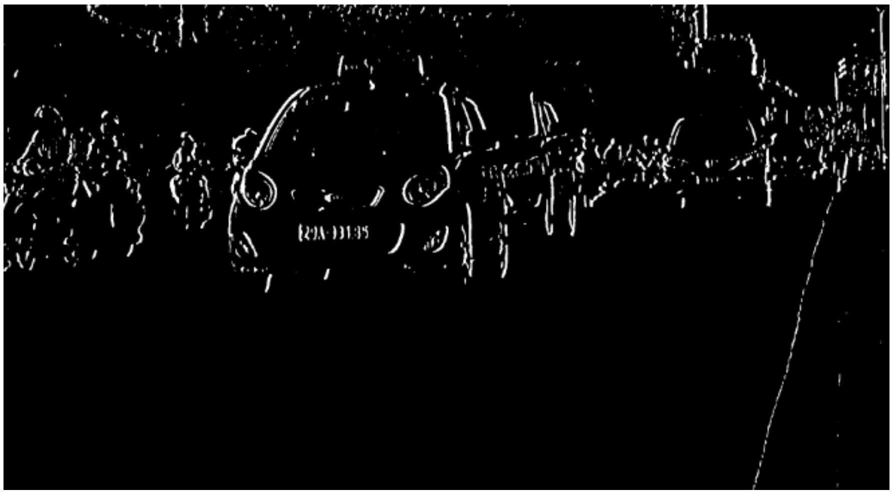
In , the indented number plate region is extracted by using the magnitude of vertical edges, and this is considered as the most robust extraction feature. The vertical edges are compared to acquire the intended rectangles which are then filtered for the one rectangle being the number plate area using the known aspect ratio. In, it is stated that if the background edges are removed and the vertical edges are obtained, the number plate can be successfully extracted from the remaining edges in the image. The total processing time for an image of size 384 × 288 is 47.9 ms and the detection rate was about 100% for 1165 test images.

Vertical Edge Detection Algorithm (VEDA) is said to be the robust algorithm for edge detection as proposed for number plate extraction in. The extraction rate for 50 images in various lighting conditions is 96%. The horizontal edges can result in possible errors. These errors occur mainly due to the car bumper . From the literature, we can also find the block-based method for this purpose. According to , the possible number plate areas are the blocks having the high edge magnitude. This method is independent of the boundary edges of the number plate and can be applied to identify it from unclear images too. An accuracy of 92.5% is obtained for character recognition, using a pair of 180 different images. Tests were carried out to check the inspection status of motorcycles by recognizing its number plate. A success rate of 95.7% for roadside and 93.7% for inspection stations test images was achieved. Hough Transform is used to extract the number plate using the boundaries. The number plate is located by detecting straight lines in the test image. This transform has the ability to detect straight lines with an inclination of up to 30 degrees. However, it is computationally expensive and requires large memory. In, the generalized symmetry transform is used. The corners from the edges in the image are detected by scanning them in selective directions. By using the generalized symmetry transform the number plate region is extracted by detecting the similarities between these corners. The continuity of the edges is important when using the edge based methods as these are considered to be simple and fast. The extraction rate can significantly improve by eliminating the unwanted edges using some morphological steps. A combination of morphology and the edge statistics was proposed in . Prior to this, the basic pre-processing techniques were also applied to enhance the image for color contrast and noise removal. 98% of successful extraction and 75–85% overall performance rate is achieved by testing on 9745 images.

1. To reduce the noise we need to blur the input Image with Gaussian Blur and then convert it to grayscale.


2. Find vertical edges in the image.

3. To reveal the plate we have to binarize the image. For this apply Otsu’s Thresholding on the vertical edge image. In other thresholding methods, we have to choose a threshold value to binarize the image but Otsu’s Thresholding determines the value automatically.

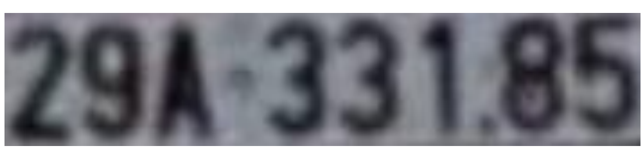


4. Apply Closing Morphological Transformation on the thresholded image. Closing is useful to fill small black regions between white regions in a thresholded image. It reveals the rectangular white box of license plates.

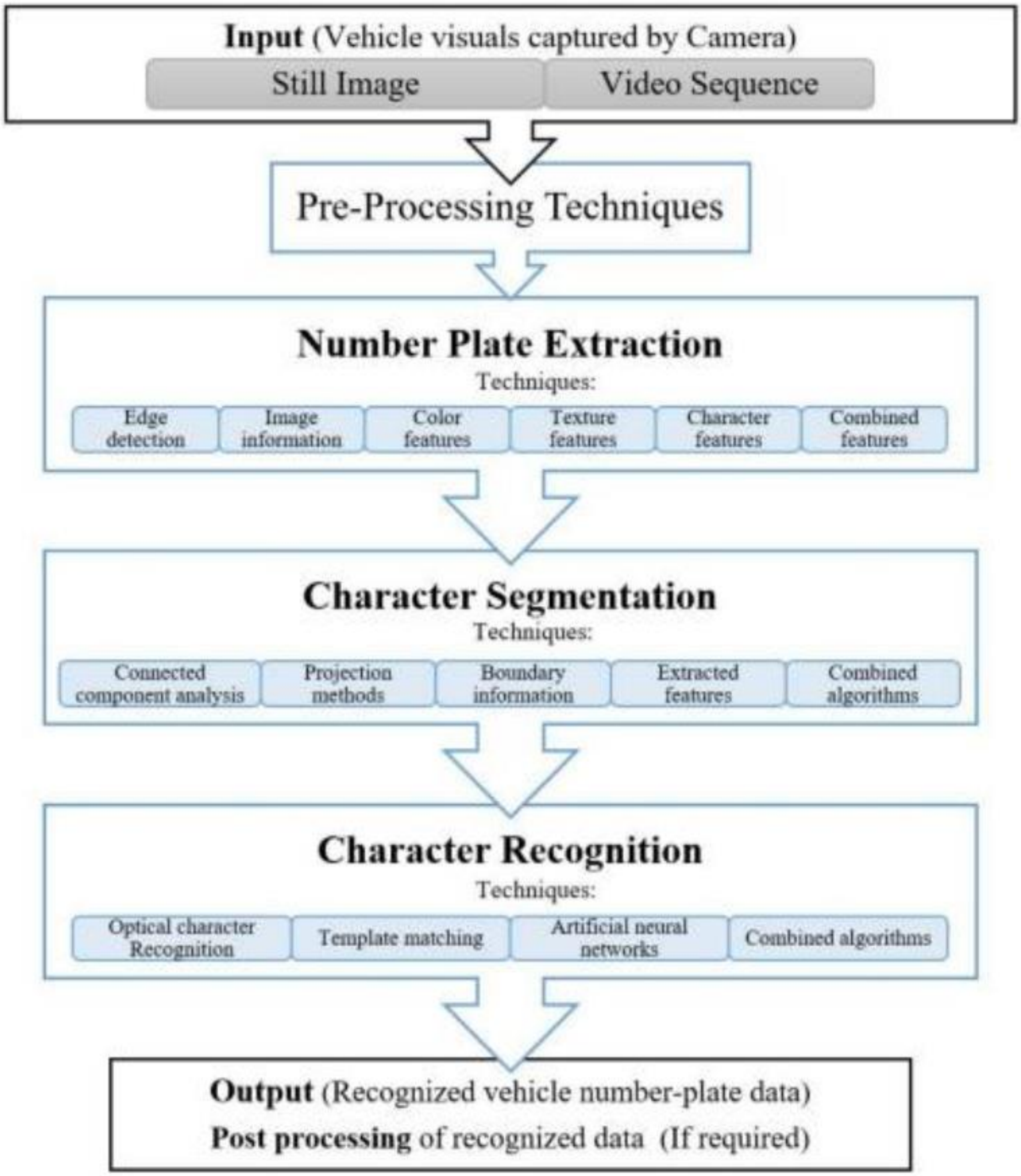
5. To detect the plate we need to find contours in the image. It is important to binarize and morph the image before finding contours so that it can find a more relevant and less number of contours in the image. If you draw all the extracted contours on the original image, it would look like this:

6. Now find the minimum area rectangle enclosed by each of the contours and validate their side ratios and area. We have defined the minimum and maximum area of the plate as 4500 and 30000 respectively.

7. Now find the contours in the validated region and validate the side ratios and area of the bounding rectangle of the largest contour in that region. After validating you will get a perfect contour of a license plate. Now extract that contour from the original image. You will get the image of the plate:



Process of ANPR is as follows :



**Packages used :**

➢ Opencv :

- The cv2 package is a Python binding for the popular OpenCV (Open Source Computer Vision Library) library, which is widely used in the field of computer vision and image processing.
- OpenCV provides a comprehensive set of tools for image and video processing, enabling developers and researchers to build applications that can capture, analyze, and manipulate visual data.
- Functions for reading, writing, resizing, rotating, cropping, and converting images between different color spaces (e.g., BGR to grayscale, RGB, HSV).
- Tools for detecting and recognizing objects, faces, and other patterns in images and videos.
- Functions to create windows and display images or videos using cv2.imshow().

➢ EasyOCR :

- EasyOCR is a Python package that provides an easy-to-use and efficient Optical Character Recognition (OCR) solution.
- EasyOCR is designed to perform text detection and recognition on images and is particularly known for its simplicity and effectiveness across various languages and scripts.
- EasyOCR uses deep learning models, specifically Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), for both text detection and recognition. This results in high accuracy, even on challenging fonts, sizes, and layouts.
- EasyOCR can accurately detect text in images, even with complex backgrounds, various fonts, orientations, and layouts (e.g., rotated or distorted text).

➢ Mysql Connector :

- The MySQL Connector is a standardized interface that allows applications to communicate with MySQL databases. Specifically, when referring to MySQL Connector/Python, it is the official Oracle-supported driver that enables Python applications to connect to and interact with MySQL databases seamlessly.
- MySQL Connector/Python is written entirely in Python, which means it doesn’t require any external libraries or components to be installed.
- connection = mysql.connector.connect(
    host='localhost',
    database='your_database',

```
        user='your_username',
        password='your_password'
    )
```

➢ plate_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_russian_plate_number.xml')

- This line of code is used to load a pre-trained Haar cascade classifier for detecting Russian license plate numbers using the OpenCV library in Python.
- Haar cascades are machine learning-based object detection methods. These are trained using a large number of positive images (i.e., images of the object) and negative images (i.e., images without the object). After training, the classifier can be used to detect objects in other images.
- The classifier works by sliding a window across the image and applying the trained Haar features to check if that region contains the object.
- If a region passes all stages of the classifier, it is considered to contain the object (e.g., a face or a license plate).
- It loads a pre-trained Haar cascade classifier for detecting Russian license plates.
- This is the name of the specific XML file that contains the pre-trained Haar cascade data for detecting Russian license plate numbers.
- The file includes the classifier data obtained from training the Haar cascade algorithm on images containing Russian license plates.

➢ Torch

- The torch package is the core library of PyTorch, an open-source machine learning framework widely used for applications such as deep learning, computer vision, and natural language processing (NLP).
- PyTorch was developed by Facebook's AI Research lab (FAIR) and has become one of the most popular deep learning frameworks due to its flexibility, dynamic computation graph, and ease of use.
- The torch package is a powerful and flexible library for building and training machine learning models, particularly deep learning models.
- Its dynamic computation graph, ease of debugging, and strong support for GPU acceleration make it a popular choice among researchers and practitioners in the field of AI and machine learning.

- ➢ Dependencies of EasyOCR on Torch

- EasyOCR is a Python library that provides an easy-to-use interface for Optical Character Recognition (OCR) using deep learning models.
- It is built on top of PyTorch, making torch a critical dependency for its operation.
- EasyOCR is heavily dependent on torch (PyTorch) for its core functionality, including model definition, tensor operations, GPU acceleration, and optimization.
- PyTorch serves as the backbone of the deep learning models used by EasyOCR, allowing it to perform efficient and accurate optical character recognition on various types of images.
- Without torch, the deep learning models in EasyOCR would not be able to function.

# CHAPTER 6 : Source code

```
import cv2
import easyocr
import mysql.connector
import torch



# Connect to the MySQL database
db = mysql.connector.connect(
    host="localhost",       # Your host, e.g., "localhost"
    user="root",   # Your MySQL username
    password="root", # Your MySQL password
    database="number_plates_db" # The database name
)

cursor = db.cursor()

# Load the pre-trained Haar Cascade for Russian number plates
plate_cascade = cv2.CascadeClassifier(cv2.data.haarcascades +
'haarcascade_russian_plate_number.xml')

# Path to the downloaded video file
video_path = "C:/Users/maury/Downloads/mini_clip.mp4" # Replace with the actual
video file path

# Capture video from the file
cap = cv2.VideoCapture(video_path)

# Initialize the EasyOCR reader
reader = easyocr.Reader(['en'],gpu=False, verbose=False)

# Check if the video file opened successfully
if not cap.isOpened():
    print("Error: Could not open video.")
    exit()

# Loop through each frame of the video
```

```
try:
  while cap.isOpened():
    # Read the current frame
    ret, frame = cap.read()

    # Break the loop if no frame is returned (end of video)
    if not ret:
        break

    # Convert the frame to grayscale
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Detect number plates in the frame
    plates = plate_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5,
minSize=(30, 30))

    # Draw rectangles around detected plates and store them in the database
    for (x, y, w, h) in plates:
        # Draw the rectangle
        cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 2)

        # Extract the detected plate region (Optional, for more processing)
        plate_region = frame[y:y+h, x:x+w]

          # Use EasyOCR to detect the text from the plate region
        detected_plate_number = reader.readtext(plate_region)
except:
    print("Video has ended.")
        # Extract the text detected by EasyOCR
if detected_plate_number:
    detected_plate_number = detected_plate_number[0][1].strip()  # Clean up the string
    detected_plate_number_str = "".join(detected_plate_number)    # Convert list to a
string

    # Insert detected plate into the database
    cursor.execute("INSERT INTO vehicles (number_plates) VALUES (%s)",
(detected_plate_number_str,))
    db.commit()
```

```
    # Display the frame with detected plates
    cv2.imshow('Number Plate Detection', frame)

    # Break the loop if the 'q' key is pressed
    cv2.waitKey(0)

# Release the video capture and close all OpenCV windows
cap.release()
cv2.destroyAllWindows()

# Close the database connection
cursor.close()
db.close()
```

# CHAPTER 7 : Result

## Installation of  cv package

```
PS C:\MINI PROJECT> pip install opencv-python
Requirement already satisfied: opencv-python in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (4.10.0.84)
Requirement already satisfied: numpy>=1.21.2 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from opencv-python) (1.26.4)
PS C:\MINI PROJECT>
```

## Installation of  easyocr package

```
PS C:\MINI PROJECT> pip install easyocr
Requirement already satisfied: easyocr in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (1.7.1)
Requirement already satisfied: torch in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (2.4.0+cu118)
Requirement already satisfied: torchvision>=0.5 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (0.19.0+cu118)
Requirement already satisfied: opencv-python-headless in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (4.10.0.84)
Requirement already satisfied: scipy in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (1.12.0)
Requirement already satisfied: numpy in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (1.26.4)
Requirement already satisfied: Pillow in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (10.2.0)
Requirement already satisfied: scikit-image in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (0.24.0)
Requirement already satisfied: python-bidi in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (0.6.0)
Requirement already satisfied: PyYAML in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (6.0.1)
Requirement already satisfied: Shapely in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (2.0.6)
Requirement already satisfied: pyclipper in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (1.3.0.post5)
Requirement already satisfied: ninja in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from easyocr) (1.11.1.1)
Requirement already satisfied: filelock in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch->easyocr) (3.14.0)
Requirement already satisfied: typing-extensions>=4.8.0 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch->easyocr) (4.12.2
Requirement already satisfied: fsspec in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch->easyocr) (2024.6.1)
Requirement already satisfied: setuptools in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch->easyocr) (70.1.1)
Requirement already satisfied: imageio>=2.33 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from scikit-image->easyocr) (2.35.1)
Requirement already satisfied: tifffile>=2022.8.12 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from scikit-image->easyocr) (2024
.8.10)
Requirement already satisfied: packaging>=21 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from scikit-image->easyocr) (23.2)
Requirement already satisfied: lazy-loader>=0.4 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from scikit-image->easyocr) (0.4)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from jinja2->torch->easyocr) (2.1.5)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from sympy->torch->easyocr) (1.3.0
)
PS C:\MINI PROJECT>
```

## Installation of torch package

```
PS C:\MINI PROJECT> pip install torch
Requirement already satisfied: torch in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (2.4.0+cu118)
Requirement already satisfied: filelock in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch) (3.14.0)
Requirement already satisfied: typing-extensions>=4.8.0 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch) (4.12.2)
Requirement already satisfied: sympy in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch) (1.13.2)
Requirement already satisfied: networkx in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch) (3.3)
Requirement already satisfied: jinja2 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch) (3.1.4)
Requirement already satisfied: fsspec in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch) (2024.6.1)
Requirement already satisfied: setuptools in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from torch) (70.1.1)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from jinja2->torch) (2.1.5)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in c:\users\maury\appdata\local\programs\python\python312\lib\site-packages (from sympy->torch) (1.3.0)
PS C:\MINI PROJECT>
```

Output generated after detecting video :



## CHAPTER 8 : Conclusion

ANPR systems are based on complicated optical, computing and digitizing capabilities that may result in a slow recognition process of plates. The ANPR solutions available in the market do not offer a standardized set for all the countries; each company has to be provided with a well optimized system for different parts/regions of the world, since the same system as developed is not sufficient and needs to be designed according to the region where deployed; keeping all the affecting factors in considerations. OCR engines often are optimized for specific countries. It needs to be made sure if the required countries are supported in the library or engine that is installed on the camera. Each ANPR solutions system provided by vendors has its own strengths and weaknesses. The best among these is the one that caters for the needs of the region in an identified system affecting conditions of that area.

Future research in ANPR still faces several challenges; For instance, there is a need to concentrate on more robust algorithms for non-standardized formats, irrespective of regions. Also, all proposed/designed algorithms need to be tested for real time scenarios rather pre-acquired images. In addition, high resolution cameras need to be integrated, allowing robust algorithms to reduce processing times and increase recognition capabilities. Yet another avenue is Obscure character recognition, since there are a lot similarities in characters like the pairs-(O,0), (P,B), (Z,2), (S,5), (3,8), (B,8), (P,R), (D,O), (1,I), (Q,O), (C,G), (A,4), (K,X), (F,E), (b,6), (q,9), (p,b), (V,W), (X,Y), (V,U), (6,8), (5,3), (5,8), (0,8), (3,9), (4,9) etc. The similarities, together with impairments, can easily deceive the optical character recognition mechanism if there is small tilt, fonts change, broken, snow or dirt on characters or if the image is acquired at different angles. Lastly, it is recommended that moving vehicles, fast speeds, low contrast, insufficient or over exposed lights and real time scenarios must be tested to check robustness of the proposed algorithms.

With recent advancements in Deep learning, Computer vision systems are enabling numerous exciting applications, ranging from safe autonomous driving to accurate object recognition, to automatic reading of images in various applications . Other real time object detectors such as YOLO, can be trained and evaluated for this system .

The Android platform has gained much importance in the technology field and numerous applications are being integrated with it. Many researchers have proposed the ANPR systems built on android platforms. However, their performance is very limited and have several constraints that can be worked on in future to develop a comparatively accurate phone-based recognition system for vehicles. Future concerns are memory resources, use of Global Positioning System coordinates for geo-tagging and online databases for respective applications .

Other than image processing based ANPR systems, RFID based vehicle verification systems are also emerging and being used in many countries for transportation applications. Radio Frequency Identification based vehicle recognition is another way to recognize vehicles identity or track them. Its purpose on road applications is similar to image processing based ANPR recognition system but the working terminology is different. RFID technology is proven to provide an effective solution to different tracking and localization problems that are more common in Image processing based systems. The most important step involved in recognition within CS/ML based ANPR systems is the extraction of the number plate from the scene, which is the most complex part in terms of performance. While using RFID for extraction/identification purposes, in case of missed vehicles this technology may come into action hence helping the ANPR. Also, the speed detection can be performed with RFIDs techniques. The vehicle may be tracked with RFID technology irrespective of its location whether it is within or without line of sight to the camera. The vehicle can be easily tracked throughout its travel on the road depending upon the types of RFID technology utilized. RFID allows toll payments facility as well. In short, RFID works on radio frequency whereas the image processing based ANPR systems are dependent on the camera. RFID does not require any camera and it can communicate with the tag on the vehicle on the go, eliminating many complexities that are associated with camera dependent technologies.

Image processing based ANPR integration with RFID technology may help in various road applications and may improve system efficiency as in . The integration of RFID and ANPR may result in a hybrid system and it can be considered for multiple applications of intelligent transport systems in present and future .

It is important to mention RFID technology here as future technology since many countries are now considering the integration of ANPR and RFID to take maximum benefit of the hybrid solution making the transportation system more accurate and secure. Both systems have their strengths and weaknesses. For ANPR, in terms of algorithms and performance, the main weakness is the successful localization of the vehicle number plate which is very much dependent on the camera and many other factors that makes it challenging to successfully recognize a vehicle number plate in some conditions mentioned earlier. Also, no additional transponders or tags are required to attach on the vehicles. The strength of this technology is that along with the recognition of vehicle its very helpful for security/surveillance applications since it captures the vehicles visuals.

The strength of RFID is the highest accuracy rates for recognition since it works on radio frequency detections by sensing the transponder attached on the vehicle, in most cases a label or tag. It can track the vehicle throughout the travel irrespective of the line of light as compared to the cameras based systems. It can effectively be used in e toll collection and the tag data can be updated accordingly. The weaknesses to this technology is limited in case of a recognition/reading vehicle however it does not store any visuals of the scene. Detection accuracy from RFID and visual security sense from ANPR together creates a hybrid system which can make the transportation system more secure and accurate.

This technology is unfortunately not a one-size-fits-all solution and needs optimization from region-to-region. To allow a uniform evaluation of different approaches, the proposed algorithms needs to be tested using complex datasets provided various factors as diversity in number plate styles, colors, fonts, sizes, orientations/tilt/skewed, occlusions, obscure characters and other physical conditions, camera resolution, shutter speed, lightning/illumination aids, coverage capability for number plate extraction from the real time complex scenes, fast moving vehicles and to maintain low processing times and increase recognition capabilities in real time scenarios. A real-time video scene is recommended for the tests rather than using pre-taken still images.

The current state of the art approaches are more inclined towards the use of OCR engines equipped with AI capabilities. Recognition algorithms based on Artificial Neural Networks are providing better recognition rates. Integration of the ANPR system with other ICT tools is also gaining popularity such as integration of ANPR engines with GPS, Online databases, Android/IOS platforms, RFID and other various tools that serve different applications in intelligent transportation systems. Future research is needed to highlight the importance and ways of incorporating this technology with other ICT tools which can be beneficial for the transport system and its policy making. The available CV algorithm’s accuracy is limited to particular regions and its standardization for the number plates. Further research is needed to make the algorithms smart enough to work in variable environments provided non-standardized diverse number-plate datasets.

Now that the accuracy of ANPR systems is improving with time and are being used in tandem with AI capabilities and IoT, it is expected that these disruptive technologies and applications will be more widely adopted and that new use cases will emerge in the coming times. It is possible with the relevant tools/software to transform the raw augmented ANPR camera data into practical knowledge and help understand the traffic flow including passenger and freight mobility. ANPR cameras, have the potential and can be augmented with vehicle category information

## CHAPTER 9 : References


https://stackoverflow.com/questions/56106040/unable-to-scrape-google-news-heading-via-their-class
https://visualstudio.microsoft.com/vs/features/cplusplus/
https://tesseract-ocr.github.io/tessdoc/Compiling-%E2%80%93-GitInstallation.html
https://youtu.be/fyJB1t0o0ms?si=MA1fpQQa7EIe6PR8
https://youtu.be/ltpnWBBT7NI?si=M-U4eQ4ar6IRg3Hi
https://www.geeksforgeeks.org/detect-and-recognize-car-license-plate-from-a-video-in-real-time/
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8123416/
https://www.mdpi.com/1424-8220/21/9/3028#B47-sensors-21-03028