

# **AUTOMATIC LICENCE PLATE RECOGNITION SYSTEM**

## **A PROJECT REPORT**

**Submitted By**

<b>E.GODWIN VIMAL KUMAR</b>	<b>-</b>	<b>510817104011</b>
<b>D.PRABHU</b>	<b>-</b>	<b>510817104028</b>
<b>H.PRAKASH</b>	<b>-</b>	<b>510817104030</b>

*In partial fulfillment for the award of the degree*

*Of*

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**GANADIPATHYTULSI'S JAIN ENGINEERING COLLEGE,**

**VELLORE**



Gateway to  
Engineering Excellence...  
SINCE 2000

**ANNA UNIVERSITY: CHENNAI 600 025**

**APRIL 2021**

# **ANNA UNIVERSITY: CHENNAI 600 025**

## **BONAFIDE CERTIFICATE**

Certified that this project report “**AUTOMATIC LICENCE PLATE RECOGNITION SYSTEM**” is the bonafide work of

**E GODWIN VIMAL KUMAR - 510817104011**

**D PRABHU - 510817104028**

**H PRAKASH - 510817104030**

who carried out the project work under my supervision.



### **SIGNATURE**

Ms.S.VENNILA. M.E.,

ASSISTANT PROFESSOR,  
Department Of Computer Science  
And Engineering

GTEC

Kaniyambadi



### **SIGNATURE**

Mr.K.SUDHAKAR.M.Tech.,

HEAD OF THE DEPARTMENT,  
Department Of Computer Science  
And Engineering

GTEC

Kaniyambadi

Submitted for the university examination held on 05-08-21



**Internal Examiner**

**External Examiner**

## ACKNOWLEDGEMENT

We first offer our deepest gratitude to our **GOD** the almighty who has given us strength and good health during the course of the project.

We are conscious about our indebtedness to our **Managing Trustee Shri.N.Sugalchand Jain**, our Chairman **Shri.P.Pyarelal Jain**, our Secretary **Shri.T.Amarchand Jain**, our **Principal Dr.M.Barathi**, who inspired us to reach greater heights in the pursuits of knowledge.

We articulate our sincere gratitude to our **Head of the Department Mr.K.Sudhakar,M.Tech.**, who had taught us the way to do a successful project and without whose constant encouragement and whole idea, the project would not have been possible.

We wish to express our sincere thanks to our **Internal Guide Ms.S.Vennila,M.E** who had helped us a lot and had given valuable suggestion, which made us to finish this project in very successful and neat way.

We wish to express our sincere thanks and honour to our **Project Coordinator Mr.K.Sudhakar, M.Tech.**,

We extend our thanks to **Mrs.S.Santhanalakshmi,M.E., Mr.R.Venkatesan, M.E.**, for their motivation. We also express our sincere thanks to the entire team of teaching and non-teaching staffs of CSE Department.

We take this opportunity to thank our parents and friends who have been the source of inspiration and an encouraging factor throughout the project period.

## **ABSTRACT**

With the explosive growth in the number of vehicles in use, automated license plate recognition (ALPR) systems are required for a wide range of tasks such as law enforcement, surveillance, and toll booth operations. The operational specifications of these systems are diverse due to the differences in the intended application. For instance, they may need to run on handheld devices or cloud servers, or operate in low light and adverse weather conditions. In order to meet these requirements, a variety of techniques have been developed for license plate recognition. Even though there has been a notable improvement in the current ALPR methods, there is a requirement to be filled in ALPR techniques for a complex environment. Thus, many approaches are sensitive to the changes in illumination and operate mostly in daylight.

This study explores the methods and techniques used in ALPR in recent literature. We present a critical and constructive analysis of related studies in the field of ALPR and identify the open challenge faced by researchers and developers. Further, we provide future research directions and recommendations to optimize the current solutions to work under extreme conditions. The character recognition network returns the bounding boxes of the predicted characters and does not provide information about the sequence of the LP number. A LP number with an incorrect sequence is considered wrong. The proposed ALPR system consumes about 42 ms per image on average for extracting LP number. Experimental results demonstrate the effectiveness of our ALPR system.

## TABLE OF CONTENT

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	iv
	LIST OF FIGURES	ix
	LIST OF ABBREVIATION	viii
1	INTRODUCTION	1
	1.1 OBJECTIVE	1
	1.2 OVERVIEW	1
	1.3 INTRODUCTION	1
2	LITERATURE SURVEY	3
3	SYSTEM ANALYSIS	8
	3.1 EXISTING SYSTEM	8
	3.1.1 DISADVANTAGES	8
	3.2 PROPOSED SYSTEM	8
	3.2.1 ADVANTAGES	8
4	REQUIREMENT SPECIFICATION	9
	4.1 HARDWARE REQUIREMENTS	9
	4.2 SOFTWARE REQUIREMENTS	9
	4.3 TECHNOLOGIES USED	10
	4.3.1 PYTHON	10
	4.3.1.1 INTRODUCTION TO PYTHON	10
	4.3.1.2 WORKING OF PYTHON	11
	4.3.1.3 THE PYTHON PROGRAMMING	12

	LANGUAGE	
	4.3.2 PYCHARM IDE	15
	4.3.3 RUSSIAN HAAR CASCADE ALGORITHM	17
	4.3.4 TESSERACT OCR	21
<b>5</b>	<b>SYSTEM DESIGN SPECIFICATION</b>	<b>23</b>
	5.1 ARCHITECTURE DIAGRAM	23
	5.2 DATA FLOW DIAGRAM	24
	5.3 UML DIAGRAM	25
	5.3.1 USE CASE DIAGRAM	25
	5.3.2 ACTIVITY DIAGRAM	26
<b>6</b>	<b>SYSTEM DESIGN IMPLEMENTATION</b>	<b>27</b>
	6.1 IMPLEMENTATION	27
	6.2 MODULE DESCRIPTION	27
	6.2.1 VEHICLE DETECTION	27
	6.2.2 LICENCE PLATE DETECTION	28
	6.2.3 CHARACTER SEGMENTATION	28
	6.2.4 STORE DATA IN CSV FILE	28
<b>7</b>	<b>SYSTEM TESTING AND MAINTENANCE</b>	<b>29</b>
	7.1 TESTING OBJECTIVES	29
	7.2 TYPES OF TESTING	30
	7.2.1 CONVENTIONAL TESTING	31
	7.2.2 UNCONVENTIONAL TESTING	31
	7.3 TEST CASE DESIGN	32
	7.3.1 UNIT TESTING	32
	7.3.2 INTEGRATION TESTING	32
	7.3.3 VALIDATION TESTING	32

7.4 TESTING STRATEGIES	33
7.4.1 INTEGRATION BOX TESTING	33
7.4.2 WHITE BOX TESTING	33
7.4.3 BLACK BOX TESTING	34
7.4.4 INTERFACE TESTING	34
7.4.5 MODULE TESTING	34
7.4.6 SMOKE TESTING	34
7.5 MAINTENANCE	35
<b>8 CONCLUSION AND FUTURE ENHANCEMENT</b>	<b>35</b>
8.1 CONCLUSION	36
8.2 FUTURE ENHANCEMENT	36
<b>APPENDIX 1</b>	<b>37</b>
<b>APPENDIX 2</b>	<b>41</b>
<b>REFERENCES</b>	<b>44</b>

## **LIST OF FIGURES**

<b>FIGURE NO</b>	<b>TITLE</b>	<b>PAGE NO</b>
5.1	ARCHITECTURE DIAGRAM	22
5.2	DATA FLOW DIAGRAM	23
5.3	USECASE DIAGRAM	24
5.4	ACTIVITY DIAGRAM	25



## **LIST OF ABBREVIATIONS**

<b>ABBREVIATIONS</b>	<b>DESCRIPTIONS</b>
<b>YOLO</b>	<b>You Only Look Once</b>
<b>OCR</b>	<b>Optical Character Recognition</b>
<b>SPP</b>	<b>Spatial Pyramid Pooling</b>

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 OBJECTIVE**

The main goal of our project is to recognize the licence plate of the vehicle.

### **1.2 OVERVIEW**

We introduced Russian haar-cascade to detect the vehicle from the input. And using haar cascade to crop the licence plate from the detected image. finally the cropped licence plate image is converted into text using Tesseract OCR(Optical Character Recognition)

### **1.3 INTRODUCTION**

Automatic License Plate Recognition (ALPR) systems are attracting an increasing interest due to their applicability in intelligent transportation systems that have been installed in many countries for tasks such as traffic law enforcement and traffic monitoring. Besides, ALPR systems are also used to manage exit and entrance in vehicle parks, collect toll payments, and to control security measures in restricted areas like military campsites, and protected sanctuaries. Often, these ALPR systems are employed to prevent fraud and to intensify security in specific areas. For instance, they can be helpful when searching for missing vehicles or vehicles related to crimes.

Unless for ALPR systems, this task requires a sizable amount of labour, time, and resources. Also, manual intervention in such tasks may leads to erroneous interpretations, and in the meantime, it is practically difficult for a human to remember or to read a license plate of a moving vehicle efficiently

Generally, an ALPR system takes an image or a video stream as the input to the system and, if the given frame contains a vehicle it outputs the content of the license plate, usually as a text. These systems consist of a camera to capture the images of the vehicles. Those images can be either colour, black and white, or infrared depending on the requirements for the system. Techniques such as object detection, image processing, and pattern recognition are used to detect and read the license plate.

## **CHAPTER 2**

### **LITERATURE SURVEY**

**[1] TITLE:** “Automatic vehicle license plate recognition using optimal K-means with convolutional neural network for intelligent transportation systems”

**AUTHORS:** Joel J P C Rodrigues ,Deepak Gupta

**JOURNAL NAME & YEAR:** IEEE access on optimal K-means with convolution neural network Jan-Feb 2015

**DESCRIPTION:**

Several researchers have begun to focus on LPR which works on LP localization, segmentation and character recognition. Therefore, effective placement of LP system is meticulous while the extensive dissection of single part requires a one to perform a task in a combined manner. This paper presents an effective DL-based VLPR model using optimal K-means (OKM) clustering-based segmentation and CNN-based recognition called OKM-CNN model.

The proposed OKM-CNN model has three main stages. During first stage, LP localization and detection process take place using Improved Bernsen Algorithm (IBA) and Connected Component Analysis (CCA) model. Subsequently, OKM clustering with Krill Herd (KH) algorithm get executed to segment the LP image. Finally, the characters in LP get recognized with the help of CNN model. An extensive experimental investigation was conducted using three datasets namely Stanford Cars, FZU Cars and HumAIn 2019 Challenge dataset.

**LIMITATIONS:**

- It works only in the daytimes
- Less efficiency.

**[2] TITLE:** "Robust Automatic Recognition of Chinese License Plates in Natural Scenes"

**AUTHORS:** Ming Xiang He ,Peng Hao

**JOURNAL NAME & YEAR:** IEEE access on Chinese License Plates in Natural Scenes" Jan-Feb 2015

**DESCRIPTION:**

The present paper proposes a new methodology for license plate (LP) recognition in the state of the art of image processing algorithms and an optimized neutrosophic set (NS) based on genetic algorithm (GA). First of all, we have performed some image processing techniques such as edge detection and morphological operations in order to utilize the (LP) localization. In addition, we have extracted the most salient features by implementing a new methodology using (GA) for optimizing the (NS) operations. The use of (NS) decreases the indeterminacy on the (LP) images. Moreover, k-means clustering algorithm has been applied to segment the (LP) characters. Finally, we have applied connected components labeling analysis (CCLA) algorithm for identifying the connected pixel regions and grouping the appropriate pixels into components to extract each character effectively. Several performance indices have been calculated in order to measure the system efficiency such as accuracy, sensitivity, specificity, dice, and jaccard coefficients. Moreover, we have created a database for all detected and recognized (LP) for testing purposes.

**LIMITATIONS:**

- The detector is not trained very well.
- It detects only the Chinese number plate.

**[3] TITLE:** “A Robust Deep Learning Approach for Automatic Iranian Vehicle License Plate Detection and Recognition for surveillance systems”

**AUTHORS:** Ali Tourani ,Sajjad soroori

**JOURNAL NAME& YEAR:** IEEE access on surveillance systems Sept 2017

**DESCRIPTION:**

The process of detecting vehicles' license plates, along with recognizing the characters inside them, has always been a challenging issue due to various conditions. These conditions include different weather and illumination, inevitable data acquisition noises, and some other challenging scenarios like the demand for real-time performance in state-of-the-art Intelligent Transportation Systems (ITS) applications. This paper proposes a method for vehicle License Plates Detection (LPD) and Character Recognition (CR) as a unified application that presents significant accuracy and real-time performance. The mentioned system is designed for Iranian vehicle license plates, which have the characteristics of different resolution and layouts, scarce digits/characters, various background colors, and different font sizes. In this regard, the system uses a separate fine-tuned You Only Look Once (YOLO) version 3 platform for each of the mentioned phases and extracts Persian characters from input images in two automatic steps.

**LIMITATIONS:**

- It uses different types of the datasets for the detection
- It gives less accuracy.
- Very difficult to detect in day times

**[4] TITLE:** “Automated License Plate Recognition: A Survey on Methods and Techniques”

**AUTHOR:** Jithmi Shashirangana; Heshan Padmasiri.

**JOURNAL NAME& YEAR:** IEEE access on methods and techniques  
Sept 2017

**DESCRIPTION:**

With the explosive growth in the number of vehicles in use, automated license plate recognition (ALPR) systems are required for a wide range of tasks such as law enforcement, surveillance, and toll booth operations. The operational specifications of these systems are diverse due to the differences in the intended application. For instance, they may need to run on handheld devices or cloud servers, or operate in low light and adverse weather conditions. In order to meet these requirements, a variety of techniques have been developed for license plate recognition. Even though there has been a notable improvement in the current ALPR methods, there is a requirement to be filled in ALPR techniques for a complex environment. Thus, many approaches are sensitive to the changes in illumination and operate mostly in daylight. This study explores the methods and techniques used in ALPR in recent literature. We present a critical and constructive analysis of related studies in the field of ALPR and identify the open challenge faced by researchers and developers. Further, we provide future research directions and recommendations to optimize the current solutions to work under extreme conditions.

**LIMITATIONS:**

- It works only in the day times.
- It requires continuous lightning

**[5] TITLE:** “Image recognition technique based on machine learning”

**AUTHOR:** Lijuan Liu, Yanping Wang.

**JOURNAL NAME & YEAR:** IEEE access on machine learning, 2016

**DESCRIPTION:**

In this era of artificial intelligence (AI), the Internet of things (IoT) with the capability of connecting a great number of heterogeneous terminals and the popularity of mobile devices, which makes more devices available as another pair of eyes for people, such as video surveillance and smart navigation. As image enhancement is assisting people to better work together and helping people live a smarter life, it is therefore becoming increasingly important. Nevertheless, many vision systems are sensitive to factors like the scattering of light or motion which may cause blur. This paper promotes an image enhancement method and is dedicated to reduce the adverse effects of blurred images on vision systems. We first researched the theory of indented frame over fusion (IFOF) and presented an automatic screening function. Then subtly combined color reversal, registration and transformation. Lastly, we tested two examples with this method and gained good results. Image processed by this method, facing strong scattering of light environment, has improved its quality and visual effects. With the development of this kind of technology, there will be more practical and intelligent applications in our lives, such as license plate numbers recognition in bad weather and important items search on fire scenes.

**LIMITATIONS:**

- It is hard to detect the complex background.
- It is difficult to recognize the licence plate in fog.



## **CHAPTER 3**

### **SYSTEM ANALYSIS**

#### **3.1 EXISTING SYSTEM**

In previous licence plate recognition system detects the numbers from the licence plate. it using canny edge and morphological to detect the number it gives the less accuracy. And the frame rate also very less in this system.

##### **3.1.1 DISADVANTAGE**

- It detect only single line licence plate layout of the vehicle
- In previous system canny edge detection algorithm is used to detect the car it gives less accuracy.
- It is tough to detect the rotated licence plate of the vehicles.

#### **3.2 PROPOSED SYSTEM**

In proposed system it uses to detect the vehicle from the input(video) using Russian haar cascade. After detecting the vehicle number plate is cropped by using haar cascade and finally the cropped licence plate is converted into text using Tesseract(OCR)

##### **3.2.2 ADVANTAGES**

- It extract the licence plate in the correct sequences.
- This system recognize the double line licence plates of the vehicle.
- The licence plate recognition in incorrect sequence it consider as wrong

## **CHAPTER 4**

### **SYSTEM REQUIREMENTS**

#### **4.1 HARDWARE CONFIGURATION**

- RAM 4 GB
- DISK SSD drive with at least 5 GB of free space
- MONITER RESOLUTION 1920×1080
- PROCESSOR Pentium –IV
- SPEED 2.3 GHZ

#### **4.2 SOFTWARE CONFIGURATION**

OPERATING SYSTEM Windows 7/8/10,LINUX.

APPLICATION Pycharm IDE 2020.3.1

- Russian Haar Cascade
- Tesseract(OCR)

## **4.3 TECHNOLOGIES USED**

### **4.3.1 PYTHON**

Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

#### **4.3.1.1 INTRODUCTION OF PYTHON**

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's Benevolent Dictator For Life, a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. He now shares his leadership as a member of a five-person steering council. In January 2019, active Python core developers elected Brett Cannon, Nick Coghlan, Barry Warsaw, Carol Willing and Van Rossum to a five-member "Steering Council" to lead the project. Guido van Rossum has since then withdrawn his nomination for the 2020 Steering council.

Python 2.0 was released on 16 October 2000 with many major new features, including a cycle-detecting garbage collector and support for Unicode.

Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely backward-compatible. Many of its major features were backported to Python 2.6.x and 2.7.x version series. Releases of Python 3 include the `2to3` utility, which automates (at least partially) the translation of Python 2 code to Python 3.

Python 2.7's end-of-life date was initially set at 2015 then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. No more security patches or other improvements will be released for it. With Python 2's end-of-life, only Python 3.6.x and later are supported.

Python 3.9.2 and 3.8.8 were expedited as all versions of Python (including 2.7) had security issues, leading to possible remote code execution and web cache poisoning.

#### **4.3.1.2 WORKING OF PYTHON**

Python is an object oriented programming language like Java. Python is called an interpreted language. Python uses code modules that are interchangeable instead of a single long list of instructions that was standard for functional programming languages. The standard implementation of python is called “cpython”. It is the default and widely used implementation of the Python. Python doesn't convert its code into machine code, something that hardware can understand. It actually converts it into something called byte code. So within python, compilation happens, but it's just not into a machine language. It is into byte code and this byte code can't be understood by CPU.

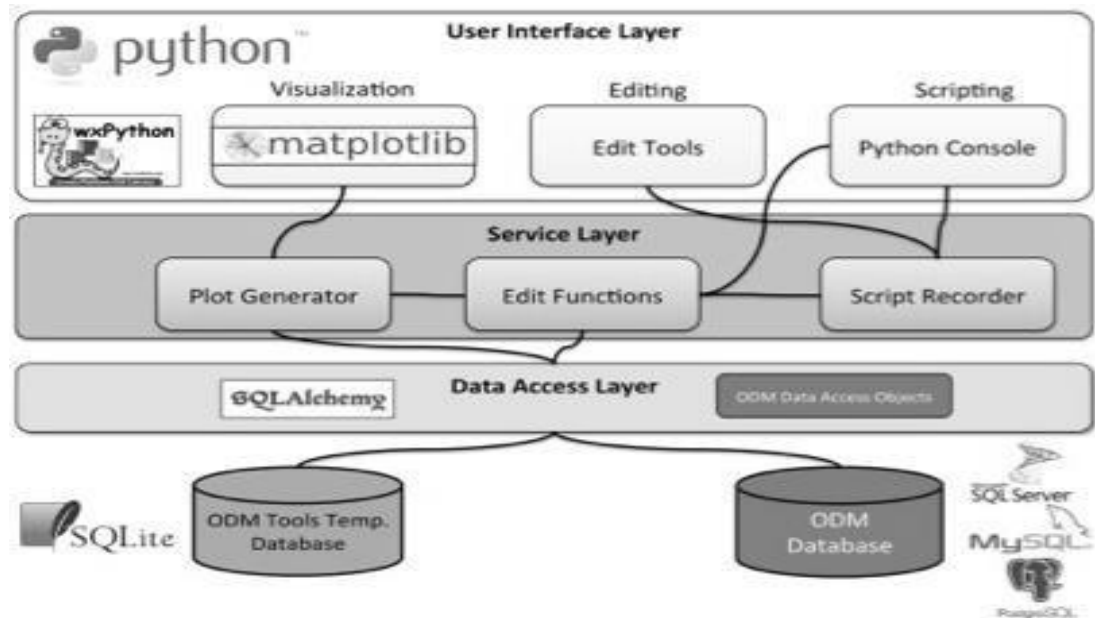
#### **4.3.1.3 THE PYTHON PROGRAMMING LANGUAGE**

Python is a popular programming language. It was created by Guido van Rossum, and released in 1991.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.
- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).

Python has a simple syntax similar to the English language. Python has syntax that allows developers to write programs with fewer lines than some other programming languages. Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick. Python can be treated in a procedural way, an object-oriented way or a functional way.

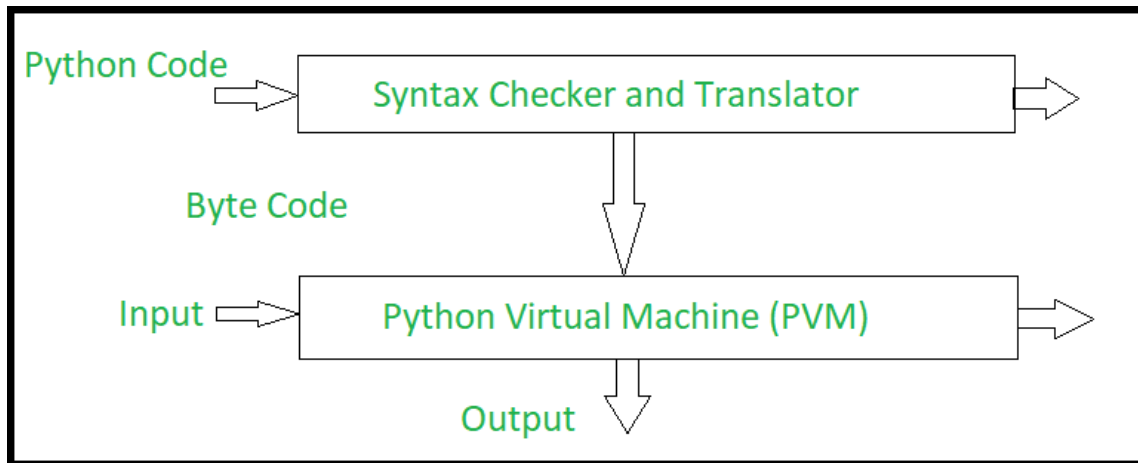


**Fig 4.1 Python Architecture**

### **Execute a Python program :**

- Step 1 : The interpreter reads a python code or instruction. Then it verifies that the instruction is well formatted, i.e. it checks the syntax of each line. If it encounters any error, it immediately halts the translation and shows an error message.
- Step 2 : If there is no error, i.e. if the python instruction or code is well formatted then the interpreter translates it into its equivalent form in intermediate language called “Byte code”. Thus, after successful execution of Python script or code, it is completely translated into Byte code.

- Step 3 : Byte code is sent to the Python Virtual Machine(PVM).Here again the byte code is executed on PVM.If an error occurs during this execution then the execution is halted with an error message.



**Fig 4.1 Working of python**

### **4.3.2 PYCHARM IDE 2020.3.1**

PyCharm is the most popular IDE used for Python scripting language. This chapter will give you an introduction to PyCharm and explains its features.

PyCharm offers some of the best features to its users and developers in the following aspects –

- Code completion and inspection
- Advanced debugging
- Support for web programming and frameworks such as Django and Flask

#### **Features of PyCharm**

Besides, a developer will find PyCharm comfortable to work with because of the features mentioned below –

##### **Code Completion**

PyCharm enables smoother code completion whether it is for built in or for an external package.

##### **SQLAlchemy as Debugger**

You can set a breakpoint, pause in the debugger and can see the SQL representation of the user expression for SQL Language code.

##### **Git Visualization in Editor**

When coding in Python, queries are normal for a developer. You can check the last commit easily in PyCharm as it has the blue sections that can define the difference between the last commit and the current one.



## **Code Coverage in Editor**

You can run **.py** files outside PyCharm Editor as well marking it as code coverage details elsewhere in the project tree, in the summary section etc.

## **Package Management**

All the installed packages are displayed with proper visual representation. This includes list of installed packages and the ability to search and add new packages.

## **Local History**

Local History is always keeping track of the changes in a way that complements like Git. Local history in PyCharm gives complete details of what is needed to rollback and what is to be added.

## **Refactoring**

Refactoring is the process of renaming one or more files at a time and PyCharm includes various shortcuts for a smooth refactoring process.

## **User Interface of PyCharm Editor**

The user interface of PyCharm editor is shown in the screenshot given below. Observe that the editor includes various features to create a new project or import from an existing project.

.

### 4.3.3 RUSSIAN HAAR CASCADE ALGORITHM

The algorithm can be explained in four stages:

- Calculating Haar Features
- Creating Integral Images
- Using Adaboost
- Implementing Cascading Classifiers

It's important to remember that this algorithm requires a lot of positive images of faces and negative images of non-faces to train the classifier, similar to other machine learning models.

#### Calculating Haar Features

The first step is to collect the Haar features. A Haar feature is essentially calculations that are performed on adjacent rectangular regions at a specific location in a detection window. The calculation involves summing the pixel intensities in each region and calculating the differences between the sums. Here are some examples of Haar features below.

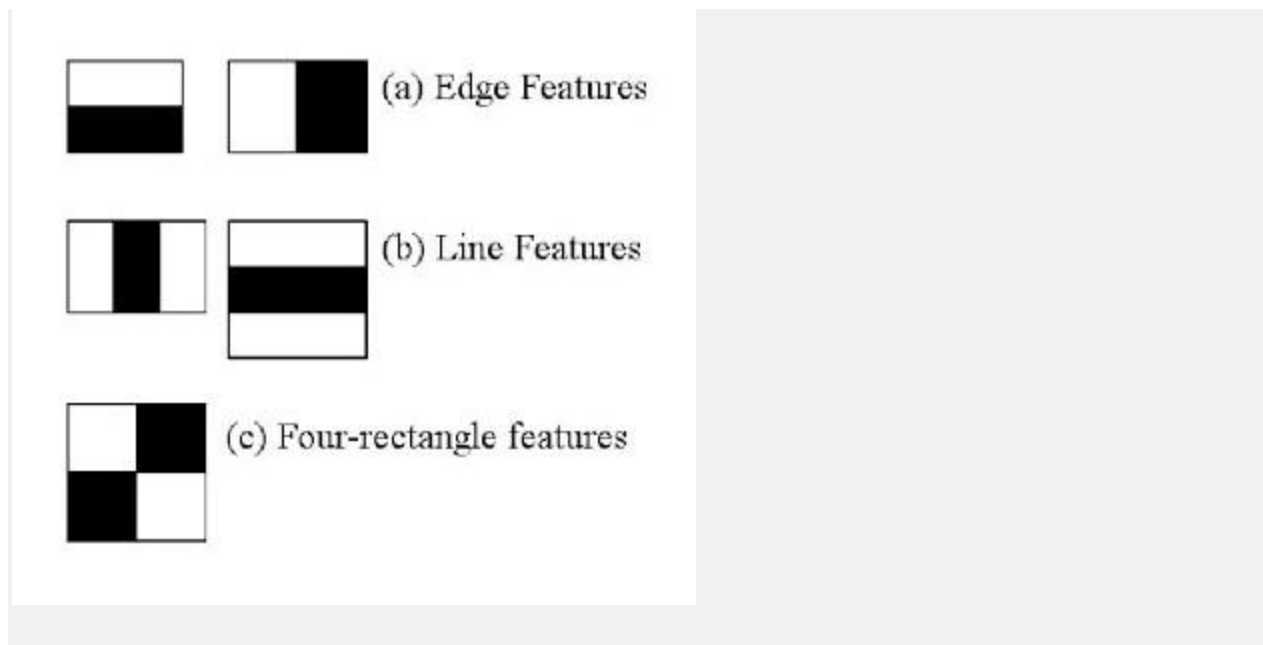


Fig 4.3.4 Edge Detection

These features can be difficult to determine for a large image. This is where integral images come into play.

### Creating Integral Images

Without going into too much of the mathematics behind it (check out the paper if you're interested in that), integral images essentially speed up the calculation of these Haar features. Instead of computing at every pixel, it instead creates sub-rectangles and creates array references for each of those sub-rectangles. These are then used to compute the Haar features.

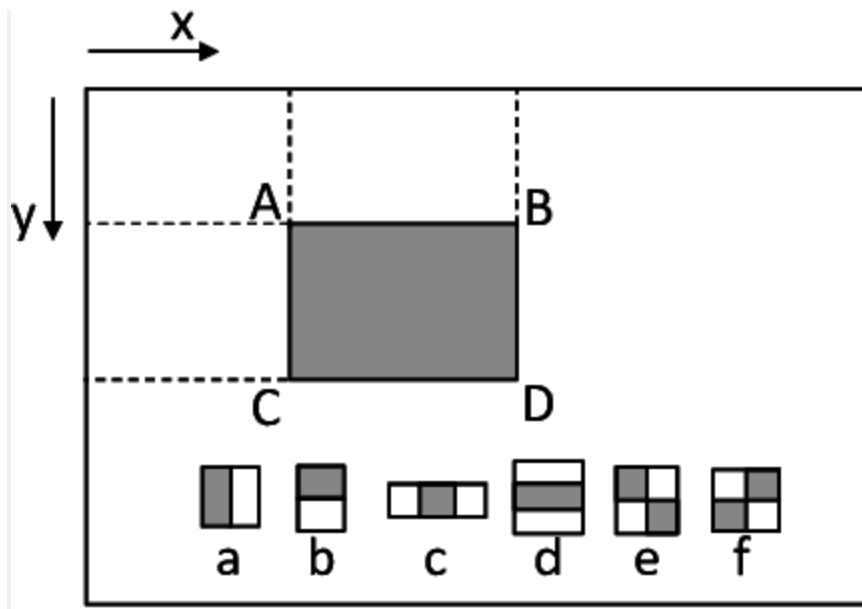


Fig 4.3.5 Graph of Edge

Illustration for how an integral image works.

It's important to note that nearly all of the Haar features will be irrelevant when doing object detection, because the only features that are important are those of the object. However, how do we determine the best features that represent an object from the hundreds of thousands of Haar features? This is where Adaboost comes into play.

### Adaboost Training

Adaboost essentially chooses the best features and trains the classifiers to use them. It uses a combination of “weak classifiers” to create a “strong classifier” that the algorithm can use to detect objects.

Weak learners are created by moving a window over the input image, and computing Haar features for each subsection of the image. This difference is compared to a learned threshold that separates non-objects from objects. Because these are “weak classifiers,” a large number of Haar features is needed for accuracy to form a strong classifier.

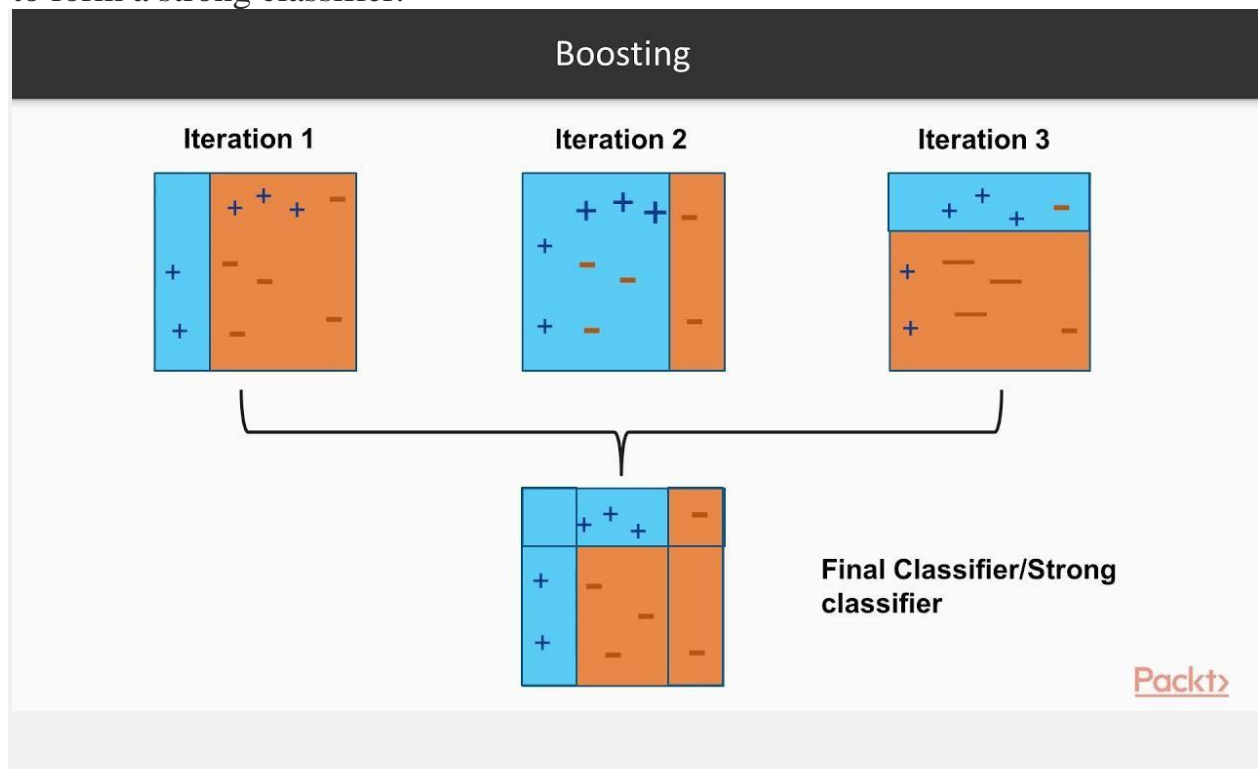


Fig 4.3.6 Iteration Diagram

The last step combines these weak learners into a strong learner using cascading classifiers.

## Implementing Cascading Classifiers

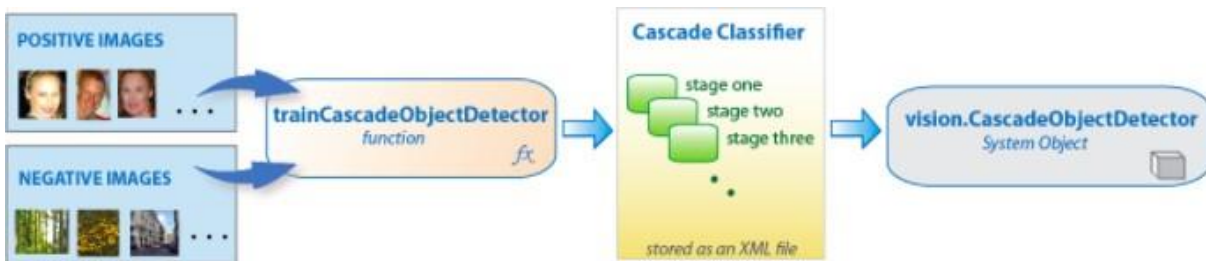


Fig 4.3.4 Haar cascade architecture

### 4.3.4 TESSERACT OCR (Optical Character Recognition)

**Optical Character Recognition (OCR)** is a process by which text characters can be input to a computer by providing the computer with an image. The computer uses an **OCR Engine**--a computer program with the specific function of making a guess which letter (recognizable to a computer) an image (recognizable to a human) represents.

Paperless includes an OCR Engine, which it uses to recognize text and numerical values. In order to understand how the OCR Engine in Paperless produces OCR results, it is useful also to understand how OCR Engines make these guesses.

#### PROCESS OF OCR

- An image of the document is acquired by the computer.
- The image is submitted as input to an OCR engine.
- The OCR engine matches portions of the image to shapes it is instructed to recognize.

- Given logic parameters that the OCR engine has been instructed to use, the OCR engine will make its best guess as to which letter a shape represents.

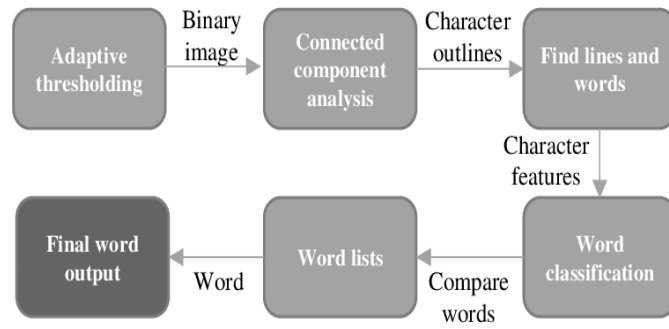


Fig 4.3.5 OCR

## CHAPTER 5

### SYSTEM DESIGN AND SPECIFICATION

#### 5.1 ARCHITECTURE DIAGRAM

Fig 4.1 In this system the image of the car getting from the video. after the image is converted into grayscale and the edges of the image is detected. the licence plate localization is done by haar cascade classifier it crop the licence plate and finally the cropped licence plate is converted into text using OCR(Optical Character Recognition). display the output on the screen.

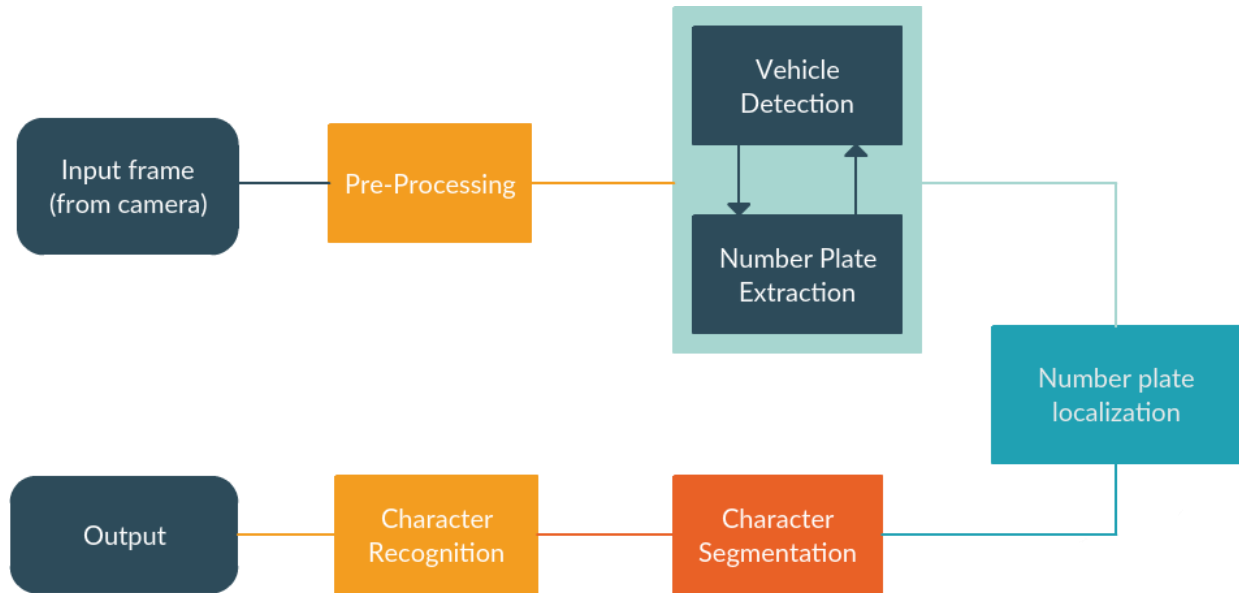


Fig 4.1-Architecture Diagram



## 5.2 DATA FLOW DIAGRAM

From the data flow diagram the capturing and recognition of the licence plate is described.

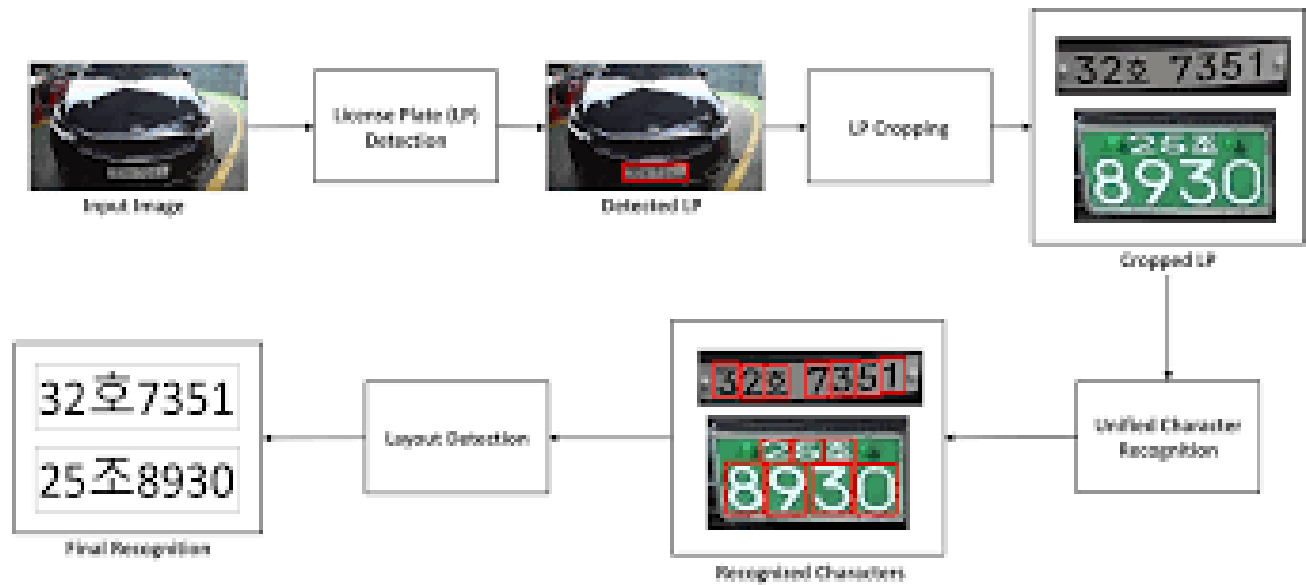


Fig 5.2- Dataflow Diagram

## 5.3 USE CASE DIAGRAM

A use case is a set of scenarios that describing an interaction between a user and a system. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use case and actors. An actor is representing a user or another system that will interact with the system modeled. A use case is an external view of the system that represents some action the user might perform in order to complete a task.

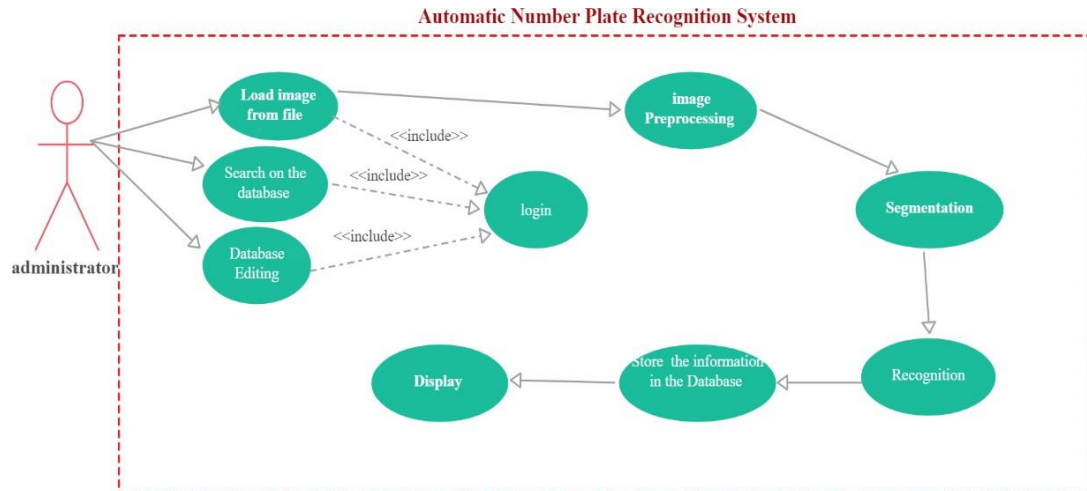


Fig 5.3-Use case diagram

## 5.4 ACTIVITY DIAGRAM

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. Activity diagram is basically a flow chart to represent the flow from one activity to another activity. The activity is described as an operation of the system.

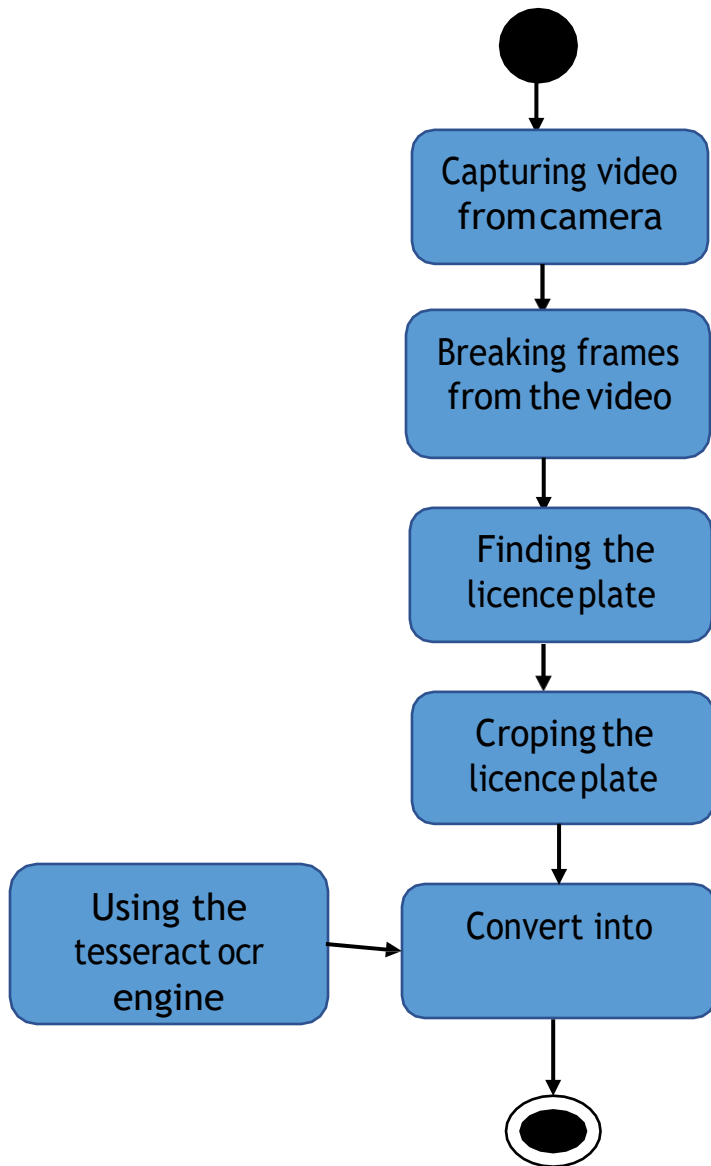


Fig 5.4- Activity Diagram

## **CHAPTER 6**

### **SYSTEM DESIGN AND IMPLEMENTATION**

#### **6.1 IMPLEMENTATION**

In this system we started with vehicle detection it capture from the video.after the detection is done the image is converted into the gray scale image.and haar cascade classifier is used to crop the licence plate from the gray scale image.the cropped licence plate image is converted into the text using the OCR -optical character recognition

#### **6.2 MODULE DISCRPTION**

- Vehicle detection
- Licence plate detection
- Unified character recognition
- Storing the data into CSV file

##### **6.2.1 VEHICLE DETECTION**

In this module capturing the video from the camera and detecting the vehicle image from the video using Russian haar cascade xml algorithm.and the original video is covertred into gray scale image.

### **6.2.2 LICENCE PLATE DETECTION**

In this module we detect the number plate from the given input. It is done by using the Haar cascade algorithm which is used to detect and crop the number plate of the detected vehicle.

### **6.2.3 UNIFIED CHARACTER RECOGNITION**

In this module the cropped licence plate is taken as an input and converted to a gray scale image and using the Tesseract(OCR) convert the cropped licence plate image into text.

### **6.2.4 STORING THE DATA INTO CSV FILE**

In this module the captured data is stored into the data.csv. It is one of the excel file formats to store the data, it is used to store a large amount of data and is easy to access the data. It is useful to keep the data for future purposes.

## **CHAPTER 7**

### **SYSTEM TESTING AND MAINTANANCE**

#### **7.1 TESTING OBJECTIVES**

Testing is a set of activities that can be planned in advance and conducted systematically. For this reason a template for software testing, a set of steps into which we can place specific test case design techniques and testing methods should be defined for software process. Testing often accounts for more effort than any other software engineering activity. If it is conducted haphazardly, time is wasted, unnecessary effort is expended, and even worse, errors sneak through undetected. It would therefore seem reasonable to establish a systematic strategy for testing software.

In the testing process we test the actual system in an organization and gather errors from the new system and take initiatives to correct the same. All the front-end and back-end connectivity are tested to be sure that the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently.

The main objective of testing is to uncover errors from the system. For the uncovering process we have to give proper input data to the system. So we should have more conscious to give input data. It is important to give correct inputs to efficient testing.

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus the system testing is a confirmation that all is correct and an

opportunity to show the user that the system works. Inadequate testing or non-testing leads to errors that may appear few months later.

This will create two problems, Time delay between the cause and appearance of the problem. The effect of the system errors on files and records within the system. The purpose of the system testing is to consider all the likely variations to which it will be suggested and push the system to its limits.

The testing process focuses on logical intervals of the software ensuring that all the statements have been tested and on the function intervals (i.e.,) conducting tests to uncover errors and ensure that defined inputs will produce actual results that agree with the required results. Testing has to be done using the two common steps Unit testing and Integration testing. In the project system testing is made as follows:

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion. Here system maintenance is done every months or year based on company policies, and is checked for errors like runtime errors, long run errors and other maintenances like table verification and reports.

## **7.2. TYPE OF TESTING**

There are two type of testing according their behaviours,

1. Unconventional Testing
2. Conventional Testing

### **7.2.1. UNCONVENTIONAL TESTING**

Unconventional testing is a process of verification which is doing by SQA (Software Quality Assurance) team. It is a prevention technique which is performing from begging to ending of the project development. In this process SQA team verifies project development activities and insuring that developing project is fulfilling the requirement of the client or not. In this testing the SQA team follows these methods:

1. Peer review
2. Code walk and throw
3. Inspection
4. Document Verification

### **7.2.2. CONVENTIONAL TESTING**

Conventional Testing is a process of finding the bugs and validating the project. Testing team involves in this testing process and validating that developed project is according to client requirement or not. This process is a correction technique where testing team find bugs and reporting to the development team for correction on developed project built.



## **7.3 TEST CASE DESIGN**

### **7.3.1 UNIT TESTING**

The primary goal of unit testing is to take the smallest piece of testable software in the application, isolate it from the remainder of the code, and determine whether it behaves exactly as you expect. Each unit is tested separately before integrating them into modules to test the interfaces between modules. Unit testing has proven its value in that a large percentage of defects are identified during its use. In the company as well as seeker registration form, the zero length username and password are given and checked. Also the duplicate username is given and checked. In the job and question entry, the button will send data to the server only if the client side validations are made. The dates are entered in wrong manner and checked. Wrong email-id and web site URL (Universal Resource Locator) is given and checked.

### **7.3.2. INTEGRATION TESTING**

Testing is done for each module. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system works.

### **7.3.3. VAIDATION TESTING**

The final step involves Validation testing, which determines whether the software function as the user expected. The end-user rather than the system developer conduct this test most software developers as a process called “Alpha and Beta test” to uncover that only the end user seems able to find. The compilation of the entire project is based on the full satisfaction of the end users. In the project,

validation testing is made in various forms. In question entry form, the correct answer only will be accepted in the answer box. The answers other than the four given choices will not be accepted.

## **7.4. TESTING STRATEGIES**

A number of software testing strategies have been proposed in the literature. All provide the software developer with a template for testing and all have the following generic characteristics:

Testing begins at the component level and works “outward” toward the integration of the entire computer-based system.

1. Different testing techniques are appropriate at different points in time.
2. The developer of the s/w conducts testing and for large projects, independent test group.

### **7.4.1 INTEGRATION BOX TESTING**

The strategies for integrating software components into a functioning product include the bottom-up strategy, the top-down strategy and to ensure that modules will be available for integration into the evolving software product when needed. The integration strategy dictates the order in which modules must be available and thus exerts a strong influence on the order in which modules are written, debugged and unit tested.

### **7.4.2 WHITE BOX TESTING**

It is just the vice versa of the Black Box testing. There we do not watch the internal variables during testing. This gives clear idea about what is going on during execution of the system. The point at which the bug occurs were all clear and were removed.

### **7.4.3 BLACK BOX TESTING**

In this testing we give input to the system and test the output. Here we do not go for watching the internal file in the system and what are the changes made on them for the required output.

### **7.4.4 INTERFACE TESTING**

The Interface Testing is performed to verify the interfaces between sub modules while performing integration of sub modules aiding master module recursively.

### **7.4.5 MODULE TESTING**

Module Testing is a process of testing the system, module by module. It includes the various inputs given, outputs produced and their correctness. By testing in this method we would be very clear of all the bugs that have occurred.

### **7.4.6 SMOKE TESTING**

Smoke testing refers to physical tests made to closed systems of pipes to test for leaks. By metaphorical extension, the term is also used for the first test made after assembly or repairs to a system, to provide some assurance that the system under test will not catastrophically fail. After smoke test proves that "the pipes will not leak, the keys seal properly, the circuit will not burn, or the software will not crash outright", the system is ready for more stressful testing.

## 7.5 MAINTENANCE

The objectives of this maintenance work are to make sure that the system gets into work all time without any bug. Provision must be for environmental changes which may affect the computer or software system. This is called the maintenance of the system. Nowadays there is the rapid change in the software world. Due to this rapid change, the system should be capable of adapting these changes. In our project the process can be added without affecting other parts of the system. Maintenance plays a vital role. The system will be able to accept any modification after its implementation. This system has been designed to favor all new changes. Doing this will not affect the system's performance or its accuracy. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion.

As a rule, system testing takes, as its input, all of the "integrated" software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

## **CHAPTER 8**

### **CONCLUSION AND FUTURE ENHANCEMENT**

#### **8.1 CONCLUSION**

This technique is very efficient in detecting the vehicle number plate from videos of the vehicle using the Russian haar cascade algorithm. It is highly effective in security areas like border crossing, red lights, toll plaza and petrol stations etc., it shows number plate extracted from all images one by one automatically.

#### **8.2 FUTURE ENHANCEMENT**

Future scope of this system to detect the multinational licence plate from the different countries. Because the vehicles are belonging from the different countries it is a chance to escape from the security to avoid this to enhance the system to detect the multinational licence plate is a challenging work.

# APPENDIX 1

## SOURCE CODE

### Main.py

```
import re

import cv2

import csv

import pytesseract

from datetime import datetime

# capturing video

video = cv2.VideoCapture('sample1.webm')

# Tesseract file location

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

# Getting csv count

file = open("output/data.csv")

reader = csv.reader(file)
```

```
lines = len(list(reader))

count = lines

# Trained Haarcascade

nPlateCascade_file = 'Haarcascade/haarcascade_russian_plate_number.xml'

nPlateCascade_tracker = cv2.CascadeClassifier(nPlateCascade_file)

# Breaking video into frame

while True:

    (read_successful, frame) = video.read()

    # cropping the video exact location

    video_crop = frame[1700:2160, 0:600]

    # converting grayscaled

    if read_successful:

        grayscaled_frame = cv2.cvtColor(video_crop, cv2.COLOR_BGR2GRAY)

    else:

        break
```

```
# Detecting the plate
```

```
nPlateCascade = nPlateCascade_tracker.detectMultiScale(grayscaled_frame)
```

```
for (x, y, w, h) in nPlateCascade:
```

```
    cv2.rectangle(video_crop, (x, y), (x + w, y + h), (0, 255, 255), 1)
```

```
    crop = video_crop[y + 1:y + h, x + 1:x + w]
```

```
    resize = cv2.resize(crop, None, fx=3, fy=3, interpolation=cv2.INTER_CUBIC)
```

```
    blur = cv2.GaussianBlur(resize, (5, 5), 0)
```

```
    gray = cv2.medianBlur(blur, 3)
```

```
    # converting image to text
```

```
    data = (pytesseract.image_to_string(gray))
```

```
    result = re.sub('[\W_]+', '', data)
```

```
    if not result:
```

```
        continue
```

```
    elif len(result) != 7:
```

```
        continue
```



```

print(result)

cv2.putText(video_crop,result,(x,y+h+25),
cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (0, 255, 255), 1)

now = datetime.now()

date = now.strftime("%d-%m-%Y")

time = now.strftime("%H:%M:%S")

# storing the data in csv

with open('output/data.csv', 'a') as csvfile:

    writer = csv.writer(csvfile, lineterminator='\n')

    writer.writerow([str(count), result, time, date])

    count += 1

# showing the sample video

cv2.imshow('Output', video_crop)

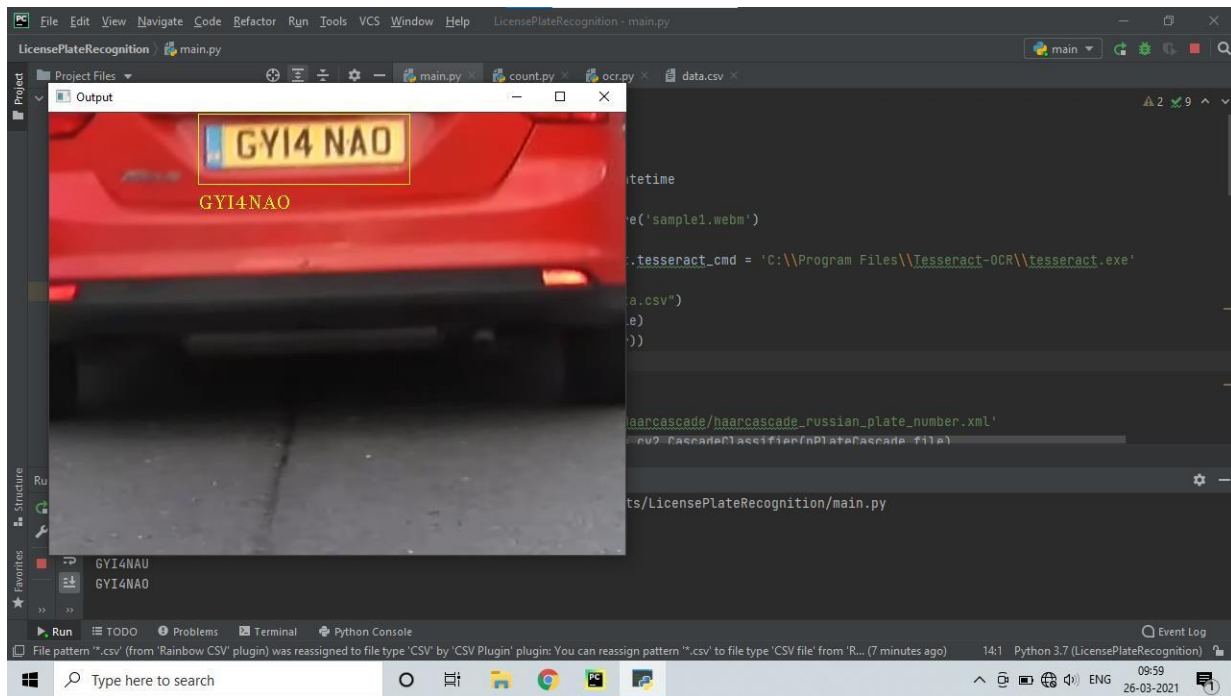
cv2.waitKey(1)

```

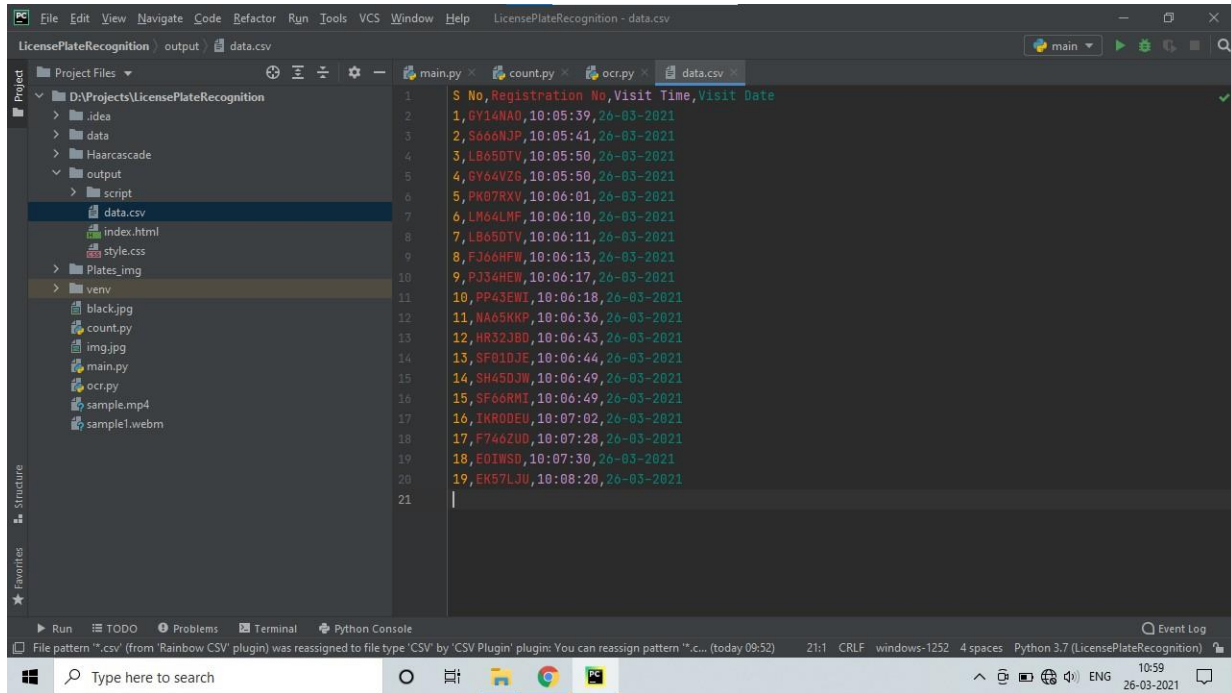
## APPENDIX 2

### OUTPUT

#### LICENCE PLATE DETECTION



## STORING THE DATA IN CSV FILE



The screenshot shows an IDE window titled "LicensePlateRecognition - data.csv". The left sidebar displays a project structure for "D:\Projects\LicensePlateRecognition" with folders like "data", "Haarcascade", "output", "script", and "Plates\_img", and files like "data.csv", "index.html", "style.css", "black.jpg", "count.py", "img.jpg", "main.py", "ocr.py", "sample.mp4", and "sample1.webm". The main editor area shows the content of "data.csv" with the following data:

S	No	Registration No	Visit Time	Visit Date
1	1	6Y14NAD	10:05:39	26-03-2021
2	2	S666NJP	10:05:41	26-03-2021
3	3	LB65DTV	10:05:50	26-03-2021
4	4	6Y64VZ6	10:05:50	26-03-2021
5	5	PK07RXV	10:06:01	26-03-2021
6	6	LM64LNF	10:06:10	26-03-2021
7	7	LB65DTV	10:06:11	26-03-2021
8	8	FJ66HFW	10:06:13	26-03-2021
9	9	PJ34HEW	10:06:17	26-03-2021
10	10	PP43EWI	10:06:18	26-03-2021
11	11	NA65KKP	10:06:36	26-03-2021
12	12	HR32JBD	10:06:43	26-03-2021
13	13	SF01DJF	10:06:44	26-03-2021
14	14	SH45DJW	10:06:49	26-03-2021
15	15	SF66RMI	10:06:49	26-03-2021
16	16	IKR0DEU	10:07:02	26-03-2021
17	17	F746ZUD	10:07:28	26-03-2021
18	18	E01WSD	10:07:30	26-03-2021
19	19	EK57LJU	10:08:20	26-03-2021

The bottom status bar indicates the file pattern ".csv" was reassigned to file type "CSV" by "CSV Plugin" plugins. The system tray shows the time as 10:59 on 26-03-2021.

## DISPLAY THE STORED DATA IN WEB

**Licence Plate Recognition**

Choose File data.csv

S No	Registration No	Visit Time	Visit Date
1	GY14NAO	10:05:39	26-03-2021
2	S666NJP	10:05:41	26-03-2021
3	LB65DTV	10:05:50	26-03-2021
4	GY64VZG	10:05:50	26-03-2021
5	PK07RXV	10:06:01	26-03-2021
6	LM64LMF	10:06:10	26-03-2021
7	LB65DTV	10:06:11	26-03-2021
8	FJ66HFW	10:06:13	26-03-2021
9	PJ34HEW	10:06:17	26-03-2021
10	PP43EWI	10:06:18	26-03-2021
11	NA65KKP	10:06:36	26-03-2021
12	HR32JBD	10:06:43	26-03-2021
13	SF01DJE	10:06:44	26-03-2021

## REFERENCES

- [1] Deepak Gupta,Joel JPC Rodrigues (2020)-“Automatic vehicle license plate recognition using optimal K-means with convolutional neural network for intelingent transportation sustems” IEEE access,vol 8,pp 92907-92917,16.
- [2] Ming Xiang He,Peng Hao (2020)-“Robust Automatic Recognition of Chinese License Plates in Natural Scenes” IEEE access,vol 8,pp 201317-201330,10.
- [3] Ali Tourani,Sajjad Soroori (2020)- “A Robust Deep Learning Approach for Automatic Iranian Vehicle License Plate Detection and Recognition for surveillance systems” IEEE access,vol 8,pp 201223-271343.
- [4] Heshan Padmasiri,Jithmi Shashirangana (2018)- “Automated License Plate Recognition: A Survey on Methods and Techniques” IEEE access,vol 5,pp 2169-3536,12.
- [5] Bedir Bedir Yousif,Mohamed Maher Ata (2017)- “Toward an Optimized Neutrosophic k-Means With Genetic Algorithm for Automatic Vehicle License Plate Recognition (ONKM-AVLPR)” IEEE access ,vol 8,pp 49285-49312,17.
- [6] Lijuan Liu,Yanping Wang (2019)- “Image recognition technique based on machine learning” IEEE access,vol 7,pp 2169-3539,06.