AUTOMATIC NUMBER-PLATE RECOGNITION USING EASYOCR

A MINIPROJECT REPORT

Submitted by

Aditya Antony Lambert (311018104003)

Amalraj P (311018104007)

Charan A R (311018104016)

Gowtham J K (311018104023)

Ramkumar (311018104304)

in partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

KCG COLLEGE OF TECHNOLOGY, KARAPAKKAM

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2021

ANNA UNIVERSITY : CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled **" AUTOMATIC NUMBER-PLATE RECOGNITION USING EASYOCR "** is the bonafide work of **" Charan A R (311018104016) , Amalraj (311018104007) , Gowtham J K (311018104023) , Aditya Antony Lambert (311018104002) And Ramkumar (3110181040304) "** who carried out the project work under my supervision.

Dr. Dhanalakshmi R
HEAD OF THE DEPARMENT
Professor
Dept. of CSE
KCG College of Technology
Karapakkam.

Mr. S Bairavel
SUPERVISOR
Assistant Professor
Dept. of CSE
KCG College of Technology
Karapakkam.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We thank the Almighty GOD for the abundant blessings showered on us. We extend our deepest love and gratitude who to our dear parents built up our career and backed us up in life.

We thank our management and our beloved Principal **Dr. Deiva Sundari P** for the opportunities given to us for our career development.

We feel indebted to the Head of the Department **Dr. Dhanalakshmi R,** Professor, Department of Computer Science and Engineering, KCG College of Technology, for all his encouragement, which has sustained our labor and efforts.

We evince our wholehearted gratitude to our project guide **Mr. S Bairavel,** Assistant Professor, Department of Computer Science and Engineering, KCG College of Technology, for her valuable guidance, ideas and support.

We extend our sincere thanks to the project coordinator **Dr. I R Praveen Joe,** Associate Professor, Department of Computer Science and Engineering, KCG College of Technology, for his guidance and support.

We would like to thank all the faculty members of Department of Computer Science, KCG College of Technology, for their inestimable teachings and encouragement during our study that has guided us towards a bright future.

Finally, we are thankful to all our friends and all others who encouraged us and helped us in doing this project.

ABSTRACT

Automatic Number-Plate Recognition is a technology that uses optical character recognition on images to read vehicle registration plates. Automatic number-plate recognition can be used to store the images captured by the cameras as well as the text from the license plate. Significant research and development of algorithms in intelligent transportation has grabbed more attention in recent years. An automated, fast, accurate and robust vehicle plate recognition system has become need for traffic control and law enforcement of traffic regulations; and the solution is ANPR (Automatic Number-Plate Recognition).

Automatic Number Plate Recognition system is used in various areas nowadays such as automatic toll collection, Border crossings, parking system, Traffic control, stolen cars tracking, maintaining traffic activities and law enforcement etc. This paper is dedicated on an improved technique of EASY-OCR based license plate recognition using neural network trained dataset of object features. A blended algorithm for recognition of license plate is proposed and is compared with existing methods for improve accuracy.

The whole system can be categorized under three major modules, namely License Plate Localization, Plate Character Segmentation, and Plate Character Recognition. This system is designed with a neural network which is trained to recognize all the characters that can be found in a Number Plate and is implemented using Python.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NUMBER
	ABSTRACT	iii
	LIST OF FIGURES	vii
1.	**INTRODUCTION**	**1**
	1.1 Overview	1
	1.2 Problem Statement	2
	1.3 Existing System	3
	1.4 Proposed System	4
	1.4.1 Advantages	4
2.	**LITERATURE SURVEY**	**5**
3.	**SYSTEM DESIGN**	**10**
	3.1 Unified Modelling Language	10
	3.2.1 Use Case Diagram of Automatic Number-Plate Recognition Using EasyOCR	11
	3.2.2 Class Diagram of Automatic Number-Plate Recognition Using EasyOCR	12
	3.2.3 Sequence Diagram of Automatic Number-Plate Recognition Using EasyOCR	13
	3.2.4 Activity Diagram of Automatic Number-Plate Recognition Using EasyOCR	14
	3.2.5 Collaboration Diagram of Automatic Number-Plate Recognition Using EasyOCR	15

4. SYSTEM ARCHITECTURE 16

4 System Architecture 16

4.1 List of Modules 16

4.1.1 Pre-Processing 16

4.1.2 Segmentation 17

4.1.3 Region Extraction 17

4.1.4 Character Recognition Using EasyOCR 18

5. SYSTEM IMPLEMENTATION 19

5.1 System Description 19

List of Modules 19

5.1.1 Collecting Dataset 19
5.1.2 Pre-Processing 20
5.1.3 Training Model 20
5.1.4 Testing Model 21

6. RESULTS AND CODING 22

6.1 Sample Code 22

6.2 Sample Screenshots 28

6.2.1 List of Components 28

6.2.2 Command For Model Training 28
6.2.3 Input Image Path 29
6.2.4 License Detection 29
6.2.5 Text Extraction 30
6.2.6 Real-Time Detection 30
6.2.7 Realtime Result Stored 31

7. CONCLUSION AND FUTURE WORK 31

7.1 Conclusion 31

7.2 Future work 32

REFERENCES 33

LIST OF FIGURES

LIST OF FIGURES	NAMES OF THE FIGURE	PAGE NO
1.	EasyOCR	02
3.1	Use Case Diagram of Automatic Number-Plate Recognition	13
3.2	Class Diagram of Automatic Number-Plate Recognition	15
3.3	Sequence Diagram of Automatic Number-Plate Recognition	16
3.4	Activity Diagram of Automatic Number-Plate Recognition	17
3.5	Collaboration Diagram of Automatic Number-Plate Recognition	18
4.	System architecture	20
6.2.1	List of Components	28
6.2.2	Command for Model Training	28
6.2.3	Input Image Path	29
6.2.4	License Detection	29
6.2.5	Text Extraction	30
6.2.6	Real-Time Detection	30
6.2.7	Realtime Results Stored	31

CHAPTER 1

INTRODUCTION

Automatic Number-Plate Recognition systems compromises integration of Artificial Intelligence along with computer vision and pattern recognition. Optical character recognition (EASYOCR) which plays chief role in automatic number plate recognition is among the main aspect of research in artificial intelligence and computer vision and have evolved greatly since its inception.

In Automatic number-plate recognition, we extract the license number i.e., the characters from a given image of a vehicle by using various dependencies and python programming.

1.1 Overview

Automatic Number-Plate Recognition system is used for many purposes like toll way authorities uses this system for allowing the vehicle to enter the toll road by detecting their number plate automatically and use it for useful purpose. Due to the mass integration of information technology in all aspects of modern life, there is a demand for information systems for data processing in respect of vehicles.

In most cases, vehicles are identified by their license plate numbers, which are easily readable by humans but not machines. For machines, a registration number plate is just a dark spot that is within a region of an image. So, by using Optical character recognition (OCR) we can easily identify the characters of the number plate to be recognized.

In the final stage EASY-OCR transforms character into encoded text information. Therefore, any authorities can get hold of the number plate of any vehicle registered. ANPR provides the best solution for providing parking management. Vehicles with registered plates can automatically enter into parking areas while non-registered vehicles will be charged by time of check in and check out.

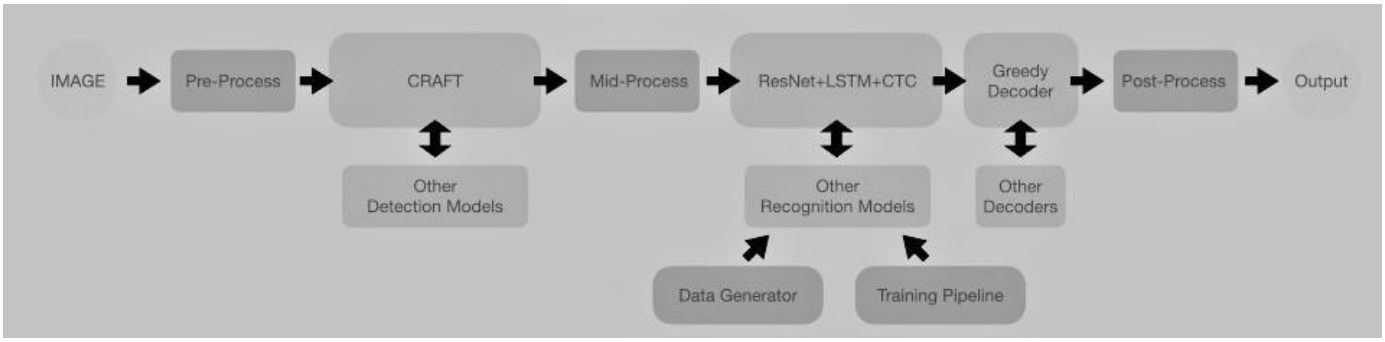

Figure 1: EasyOCR

The foremost function of our project is to recognize the characters of a given number plate by using the easy-ocr. This will provide the accurate characters of various Number-plates upon which the program depends. Overall, our project idea is feasible, which can be easily be implemented and has a wide scope in terms of its application as long as transportation in road happens.

1.2 PROBLEM STATEMENT:

One of the foremost sectors over the past 50 years that may have caused various discomforts and risks is the Automotive sector. Therefore, there is the need of increase of knowledge in surveillance.

To design and develop a real-time detection, tracking and license plate recognition system that will work efficiently under the conditions of slow-moving objects and the objects that are merged into the background due to a temporary stop and becoming foreground again, adaptive to different traffic environment conditions, robustness against progressive or sudden illumination changes, Occlusions, identification time of the system should be as short as possible.

The system should detect all the types of vehicles, recognize all the license plates of the country and should also be resistant to any kinds of disturbances, which may occur in images and mechanical plate damages which may appear in reality.

The attributes of the License plates play important role in the recognition process. The size, color of the license plate, its font face i.e. size, color of each character, spacing between characters, the number of lines in the license plate, script, characters' height and width are maintained very strictly in developed countries. There are several countries in the world who have adapted this very method of standardizing the License plate.

But in India, the license plate is purely localized and people don't follow the standard pattern assigned by Indian government, so the recognition process is quite difficult. Apart from these conditions, the algorithms applied for the recognition also plays a vital role.

If the quality of the algorithm is good, then more varieties of images can begiven as input to the system, and this will also reduce the computation speed of the process. The most basic issues in the real-time ANPR system are the accuracy and the recognition speed.

1.3 Existing System

The existing system of Automatic Number-Plate Recognition has few setbacks or drawbacks which makes the recognition hard in real time. Therefore, the major cause that affects the recognition of the number plates are:

- High quality imaging –the quality of the captured image is high, so that it will be easy to evaluate the attributes of the image.
- Minimal skewing and rotation –the camera is fit such that the captured image or video will not suffer much decent angle of skewing and rotation.
- Risk of mismanagement and of data when the project is in development.

- Moreover, cameras deployed in India tend to be of lower quality, thereby compounding the ability for ANPR engines to accurately decode the license plate.

1.4 Proposed System

To debug the existing system, remove procedures those cause data redundancy, make the algorithm proper. In addition to enhancing the underlying ANPR engine for India, we have made some concrete suggestions to help our customers achieve high accuracy ANPR with Plate Recognizer.

Advantages:

- The Recognition process can be quicker and easier.
- As the model is trained well with more no of images of vehicle number plate, the recognition has a good success rate.
- This technique is simpler and faster than most others.
- This Technique has only 3 steps.

The three main stages of this technique are: Number Plate Localization (NPL), character segmentation, OCR matching. In the first stage Number Plate (NPs) are identified and localized in the scene and improve plate visual using pre-processing techniques.

In the second stage characters are segmented from the detected Number-Plate so only useful information is retained for recognition. In the final stage OCR transforms character into encoded text information

CHAPTER 2

LITERATURE SURVEY

Christos-Nikolaos E. Anagnostopoulos [1] presented various methods used for number plate extraction. Shan Du [2] presented a survey on existing ANPR methods and categorizing them according to the features used in each stage and compares them in terms pros, cons, accuracy, and processing speed. Sahil Shaikh [3] proposed method for number plate recognition. For plate localization, several traditional images processing techniques such as image enhancement, edge detection, filtering and component analysis are used. Norizam Sulaiman [4] presented the development of automatic vehicle plate detection system in which after pre-processing the candidate plate is detected by means of feature extraction method, character segmentation is done by boundary box and character recognition is done by template matching. Reza Azad [5] proposed a fast and real time method in which has an appropriate application to find tilt and poor-quality plates. In the proposed method, the image is converted into binary mode using adaptive threshold.

Ronak P Patel [6] proposed new algorithm for recognition number plate using Thresholding operation, Morphological operation, Edge detection, boundary box analysis for number plate extraction. Najeem Owamoyo [7] proposed Automatic Number recognition for Nigerian vehicles. Number plate extraction is done using Sobel edge detection filter, morphological operations and connected component analysis. Character segmentation is done by connected component and vertical projection analysis. Sourav Roy [8] proposed algorithm for localization of number plate for the vehicles in West Bengal (India) and segmented the numbers as to identify each number separately.

This approach is based on morphological operation and sobel edge detection. After reducing noise from the input image the enhancement of image is done using histogram equalization.

Divya Gilly [9] proposed an efficient method for LPR. LPR system mainly consists of three main phases 1) plate detection 2) character segmentation 3) character recognition. This method utilizes a template matching technique for character recognition.

This method is suitable for both Indian number plates and foreign license plates. Isack Bulugu [10] has proposed an algorithm that is designed to recognize the license plate from the front end and rear end of the vehicle. The implementation of the program is developed on MATLAB (R2010b). Rupali Kate [11] has proposed an algorithm based on morphological operation with number of area criteria tests for number plate localization. Character segmentation was achieved region props toolbox function in MATLAB and character recognition was done by the Template matching. P.Mohan Kumar [12] proposed method for real time vehicle license plate identification.

Hadi Sharifi [13] presented the study and evaluation of some important license plate detection algorithms and compared them in terms of performance, accuracy, complexity, and their usefulness in different environmental condition. The dynamic programming algorithm is the fastest and the Gabor transform is the most accuracy algorithm compared to other license plate detection algorithms.

Kumar Parasuraman [14] has proposed an algorithm consist of 3 parts. Edge detection algorithm and vertical projection method are used for extracting the Plate region. In segmentation part, filtering and vertical and horizontal projection are used. Chain code concept is used for character recognition. In raspberry pi is used for the recognition the Number plate automatically. This system is used opencv Platform. For character detection Optical Character Recognition (OCR) method is used. For distance measurement ultrasonic sensor is used.

In [15] Connected-component Analysis (CCA) method identify letters. The proposed algorithm is compared with the existing methods and found that it is performed better than the existing methods. In [16] presents a system which is capable of identifying the texts written in any style

and at any angular position. In [17] template matching algorithm is used with normalized cross correlation and phase correlation method. The system takes 90 patterns for test under several conditions. This paper presents comparison between normalized cross correlation method and phase correlation method. For identification of number plate normalized cross correlation method used. For identification of number plate accuracy was 67.98 of normalized cross correlation method.

S. Hamidreza Kasaei [18] presented a real time and robust method of license plate detection based on the morphology and template matching. Ronak P Patel [19] proposed new algorithm for recognition number plate using Morphological operation and bounding box analysis for number plate extraction. Shan Du presented a comprehensive survey on existing ANPR techniques by categorizing them according to the features used in each stage and compare them in terms accuracy, and processing speed.

Shan Du, Mahmoud Ibrahim, Mohamed Shehata and Wael presented a review by categorizing different techniques used in Automatic License Plate Recognition (ALPR) according to the features these techniques used in each stage. The author suggests recognition of plates with different styles, multiple plates and recognizing characters with ambiguity as the future area of research in ALPR [20]. G. T. Shrivakshan and Dr. C. Chandrasekar presents a comparison of various edge detection techniques used in image processing [14].

This explains the behavior of real ants that can be used to solve several combinatorial problems. Ant System (AS) is proposed to be a valid approach to stochastic combinatorial optimization in . The paper implements AS to solve various problems like Travelling Salesman Problem, Quadratic Assignment and Job- Shop Scheduling. Proposes an ACO based edge detection approach by exploiting Ant Colony System. The performance of this approach is proved to be better than that discussed in [21] which is obtained by exploiting Ant System. Proposes a weighted heuristic ACO algorithm. The proposed approach provides better accuracy

by setting weights to the intensities of neighboring pixels. This is very well exploited by the ants which move continuously over the image.

Namrata Dave discussed various methodologies to segment a text-based image at various levels of segmentation. The author concludes that the pixel counting approach is the most suitable to segment printed text whereas histogram approach is flexible for both handwritten and printed text but is computationally slow.

Sunitha Beevi K. and Sajeena A. proposes an algorithm which focuses on segmenting the characters of two rows license plate image. [22] addresses the issue of inaccurate license plate location. The paper further provides a solution of extracting characters in such a situation. However, presence of noise is seen to introduce errors in the extraction procedure. Lihong Zheng, Xiangjian He, Qiang Wu, Wenjing Jia, Bijan Samali and Marimuthu Palaniswami designed a two-stage classifier based on Inductive Learning and SVM-based classification which can be used to recognize characters in real time applications. L. Zheng X. He, Q. Wu and T. Hintz proposed a number recognition algorithm on Spiral Architecture which is a hexagonal image structure.

In the concept of manipulating images is used to recognize number plates. The method proposed yielded a success rate of 98.73%. However, the research was only limited to the recognition of Indian license plates with a minimum resolution of 640X480, Najeem Owamoyo, A.

Alaba Fadele and Abimbola Abudu presented Automatic Number Plate recognition for the Nigerian vehicular plates in which character extraction and segmentation was done using Sobel edge mask and Connected Component Analysis (CCA) with accuracy of 80%. A SVM based character recognition system is implemented which yielded 79.84% success rate. Low success rate is mainly due to the impact of deep shadows and reflections on the characters.

The author suggests designing of two SVMs for digits and characters to improve recognition accuracy [23]. In, an Artificial Neural Network based OCR algorithm for Automatic Number

Plate Recognition (ANPR) application is proposed. The proposed approach meets the requirements of real time ANPR system by recognizing character image in 8.4 ms with a 97.3% success rate.

Najeem Owamoyo proposed method for number plate extraction using Sobel filter and morphological operations. Divya Gilly [24] presented an efficient method for license plate detection by connected component analysis. Isack Bulugu has proposed edge finding method to find the location of the plate. Rupali Kate proposed algorithm based on a combination of morphological operation with area criteria tests for number plate localization.

Hadi Sharifi [24] has study and evaluates some most important license plate detection algorithms and compared them in terms of accuracy, performance, complexity, and their usefulness in different environmental condition. This evaluation gives views to the developers or end-users to choose the most appropriate technique for their applications.

The study and investigation show that the dynamic programming algorithm is the fastest and the Gabor transform is the most accuracy algorithm compared to other algorithms. Kumar Parasuraman and P. Vasantha Kumar [25] proposed algorithm for extracting the Plate region using edge detection algorithm and vertical projection method.

This paper presents an efficient approach for the extraction of number plate from the vehicle image based on morphological operations (opening, closing, dilation, and erosion), image subtraction, thresholding, sobel edge detection and the connected component analysis. Firstly, the input image is pre-processed by iterative bilateral filter and adaptive histogram equalization.

CHAPTER 3

SYSTEM DESIGN

In this chapter, the various UML diagram for Automatic Number-Plate Recognition (ANPR) System is represented and various functionalities are explained.

3.1 UNIFIED MODELLING LANGUAGE

Unified Modelling language (UML) is a standardized modelling language enabling developers to specify, visualize, construct and document artifacts of a software system. Thus, UML makes these artifacts scalable, secure and robust in execution. UML is an important aspect involved in object-oriented software development. It uses graphic notation to create visual models of software systems.

The UML architecture is based on the meta object facility, which defines the foundation for creating modelling language. They are precise enough to generate the entire application. A fully executable UML can be deployed to multiple platforms using different technologies and can be used with all processes throughout the software development cycle.

UML is designed to enable users to develop an expressive, ready to use visual modelling language. In addition, it supports high level development concepts such as frameworks, patterns and collaborations.

Some of the UML diagrams are discussed below.

3.1.1 USE CASE DIAGRAM OF AUTOMATIC NUMBER-PLATE RECOGNITION

Use case diagrams are considered for high level requirement analysis of a system. So, when the requirements of a system are analysed, the functionalities are captured in use-cases. So, it can be said that uses cases are nothing but the system functionalities written in an organized manner. Now the second things which are relevant to the use cases are the actors.

The actors can be human user, some internal applications or may be some external applications. These requirements are mostly design requirements. Hence, when a system is analysed to gather its functionalities, use-cases are prepared and actors are identified.

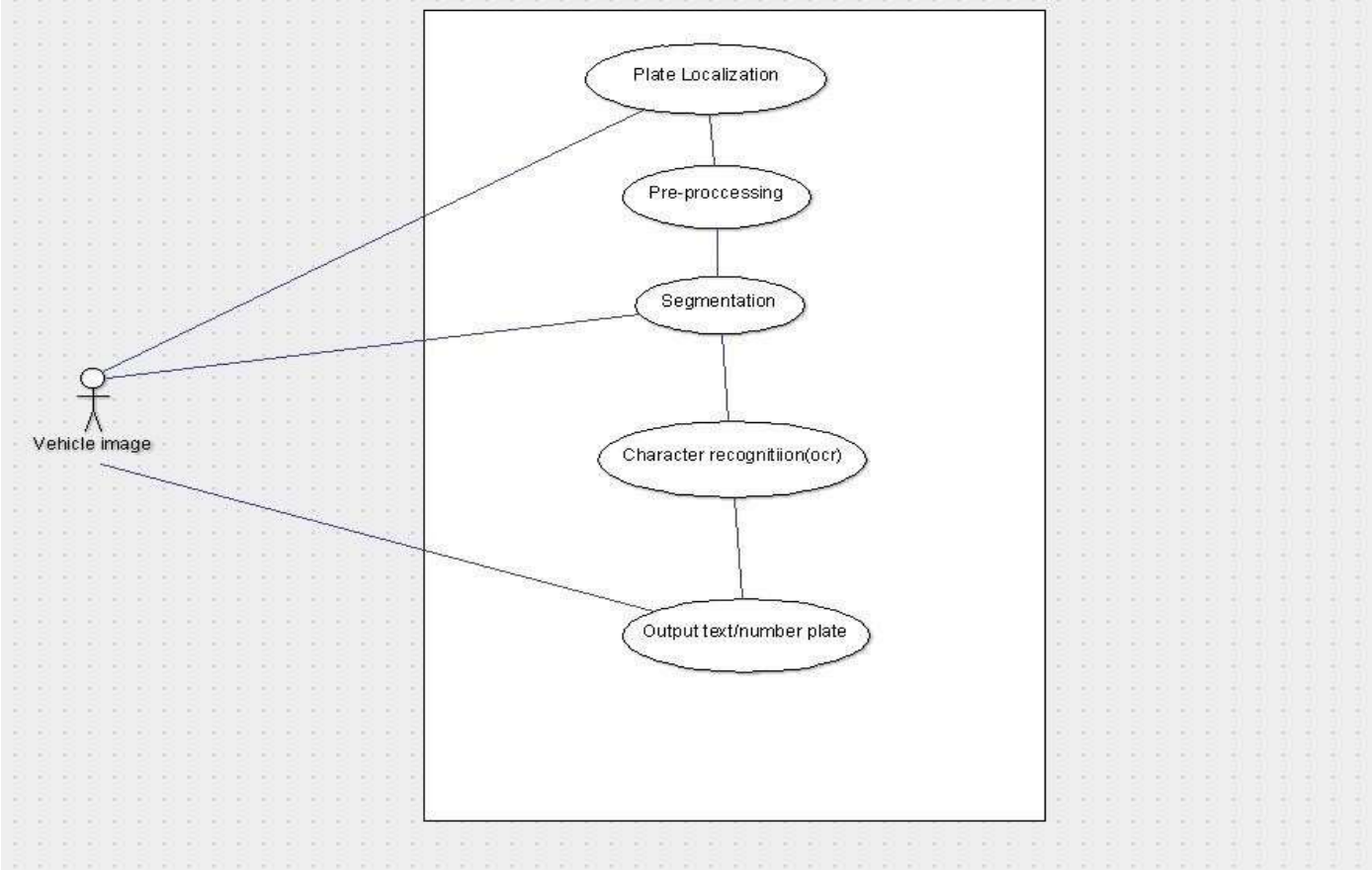

Figure 3.1 USE CASE DIAGRAM FOR ANPR

The Functionalities are to be represented as a use case in the representation. Each and every use case is a function in which the user or the server can have the access on it. The names of the use cases are given in such a way that the functionalities are preformed, because the main purpose of the functionalities is to identify the requirements. To add some extra notes that should be clarified to the user, the notes kind of structure is added to the use case diagram. Only the main relationships between the actors and the functionalities are shown because all the representation may collapse the diagram. The use case diagram as shown in Figure 3.1 provides details based on the OCR Tool.

3.1.2 CLASS DIAGRAM OF AUTOMATIC NUMBER-PLATE RECOGNITION

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. So, a collection of class diagrams represent the whole system.

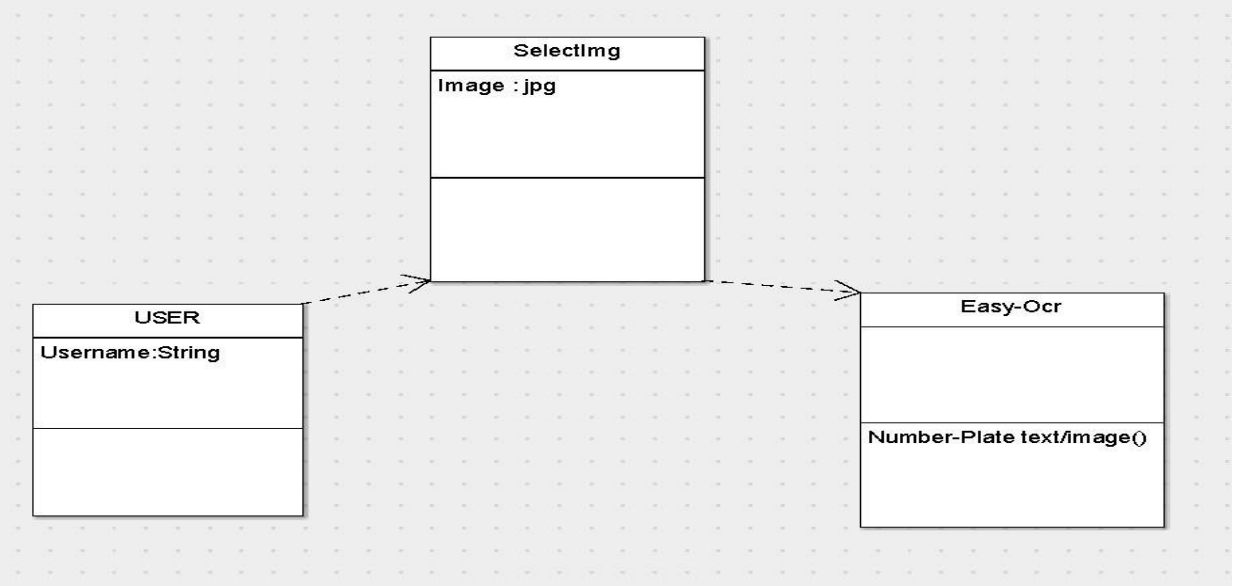

Figure 3.2 CLASS DIAGRAM OF ANPR

The name of the class diagram should be meaningful to describe the aspect of the

system. Each element and their relationships should be identified in advance of the responsibility (attributes and methods) of each class should be clearly identified for each class minimum number of properties should be specified. All of these each specifications for the system are displayed as a class diagram in Figure 3.2.

3.1.3 SEQUENCE DIAGRAM OF AUTOMATIC NUMBER-PLATE RECOGNITION

UML sequence diagrams model the flow of logic within the system in visual manner, enabling to both document and validate the logic, and are commonly used for both analysis and design purposes.

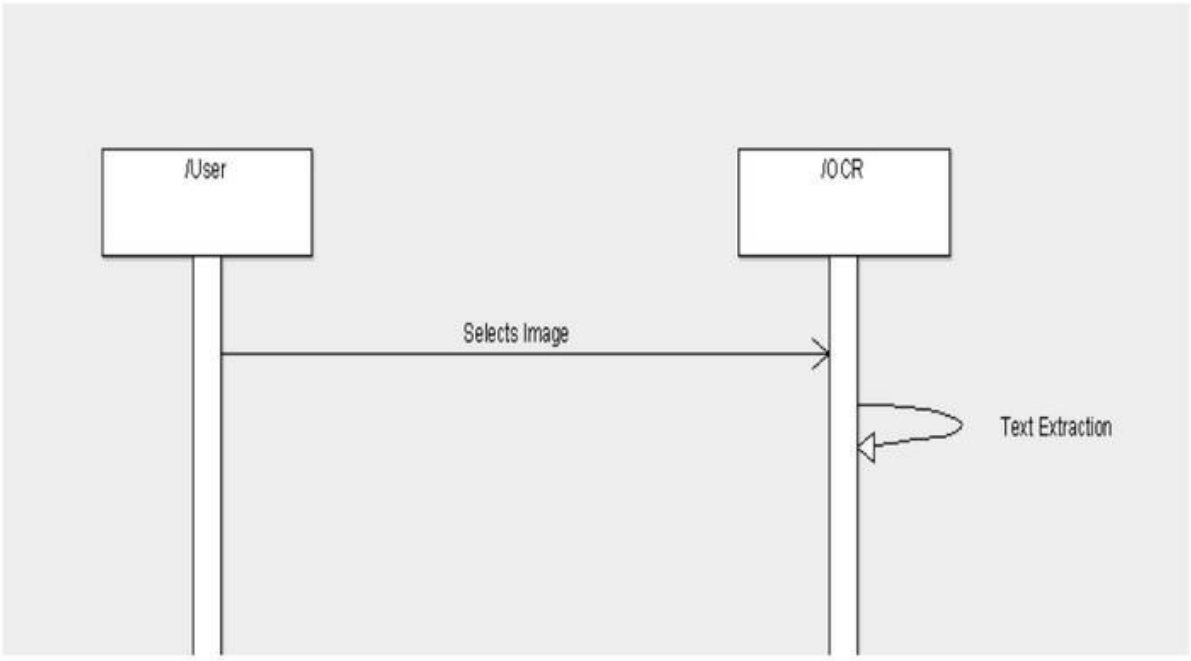

Figure 3.3 SEQUENCE DIAGRAM OF ANPR

The various actions that take place in the application in the correct sequence are shown in Figure 3.3 Sequence diagrams are the most popular UML for dynamic modelling.

3.1.4 ACTIVITY DIAGRAM OF AUTOMATIC NUMBER-PLATE RECOGNITION

Activity is a particular operation of the system. Activity diagram is suitable for modelling the activity flow of the system. Activity diagrams are not only used for visualizing dynamic nature of a system but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in activity diagram is the message part.

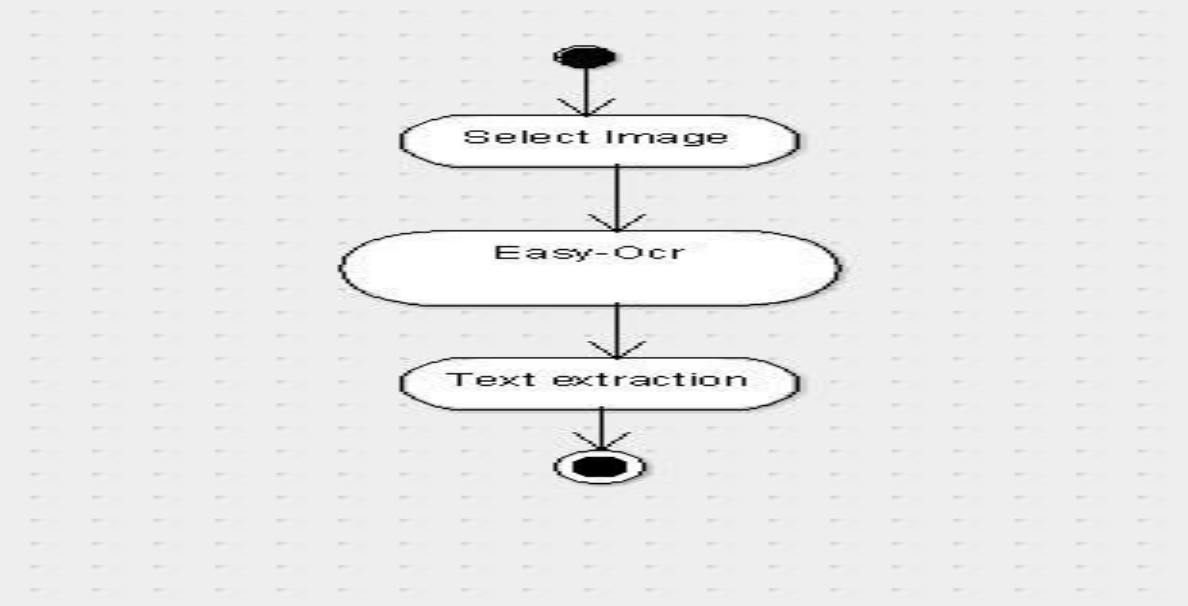

Figure 3.4 ACTIVITY DIAGRAM OF ANPR

An application can have multiple systems. Activity diagram also captures these systems and describes the flow from one system to another. This specific usage is not available in other diagrams. These systems can be database, external queues, or any other system. Activity diagram is suitable for modelling the activity flow of the system. It does not show any message flow from one activity to another. Activity diagram is sometime considered as the flow chart. Although the diagrams look like a flow chart but it is not.

It shows different flow like parallel, branched, concurrent and single. The Figure 3.4 shows the activity diagram of the developed application.

3.1.5 COLLABORATION DIAGRAM OF AUTOMATIC NUMBER-PLATE RECOGNITION

The next interaction diagram is collaboration diagram. It shows all of the object organization. Here in collaboration diagram the method call sequence is indicated by some numbering technique. The number indicates how the methods are called one after another.

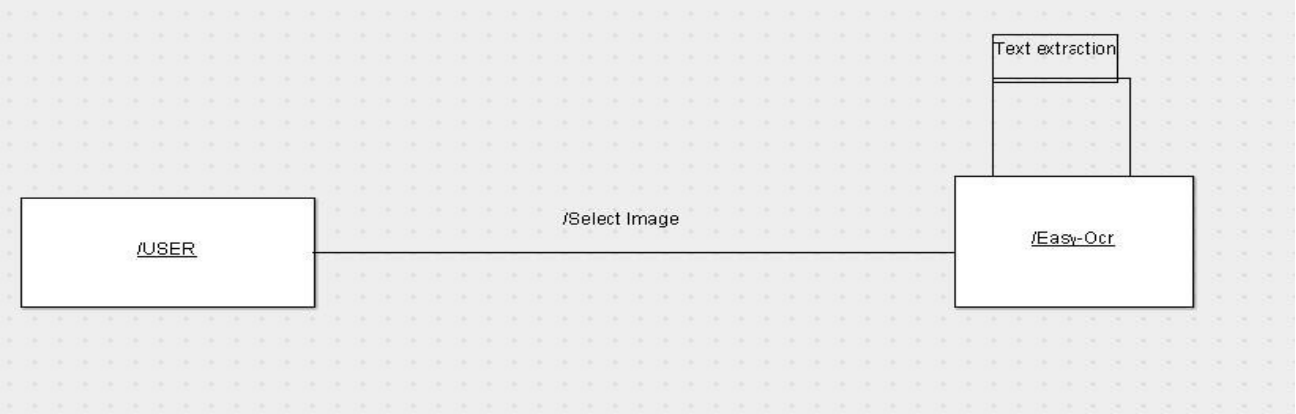

Figure 3.5 COLLABORATION DIAGRAM OF ANPR

The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization but whereas the collaboration diagram shows the object organization. The various objects involved and their collaboration is shown in Figure 3.5. Now to choose between these two diagrams the main emphasis is given on the type of requirement. If the time sequence is important then sequence diagram is used and if organization is required then collaboration diagram is used.

CHAPTER 4

SYSTEM ARCHITECTURE

In this chapter, the System Architecture for the Automatic Number-Plate Recognition Using Easy-Optical Character Recognition (OCR) is represented and the modules are explained.

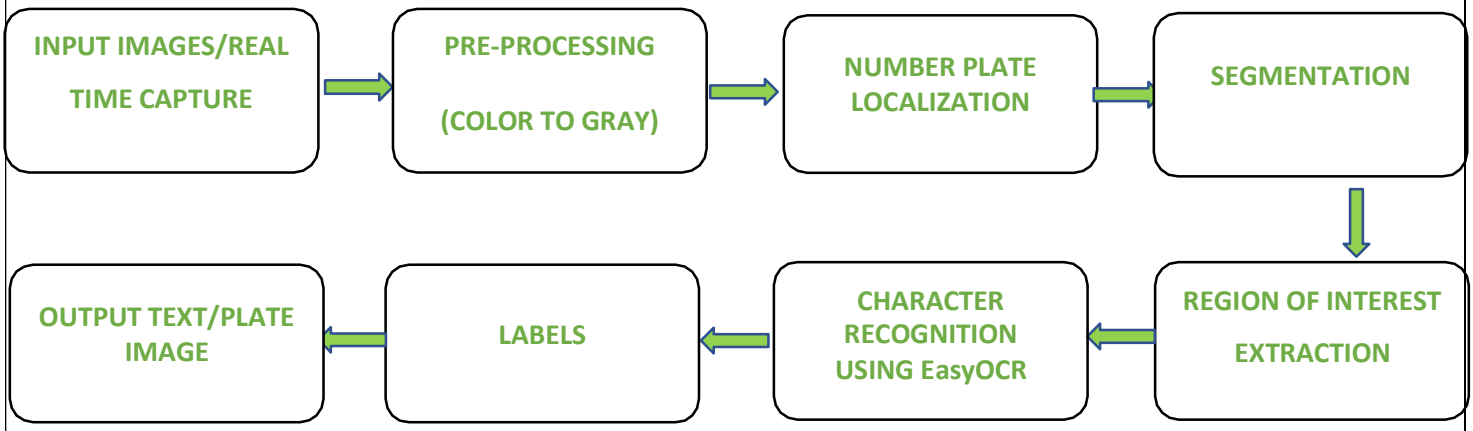

Figure 4: SYSTEM ARCHITECTURE

4.1 LIST OF MODULES

1. **Pre-Processing**
2. **Segmentation**
3. **Region Extraction**
4. **Character Recognition Using EasyOCR**

4.1.1 Pre-Processing

In The pre-processing phase, there is a series of operations performed on the scanned input image. It enhances the image rendering it suitable for segmentation the Gray-level

character image is normalized into a window sized. After noise reduction, we produced a bitmap image. Then, the bitmap image was transformed into a thinned image.

4.1.2 Segmentation

In the segmentation of number plate characters, numberplate is segmented into its constituent parts obtaining the characters individually. Firstly, image is filtered for enhancing the image and removing the noises and unwanted spots. Then dilation operation is applied to the image for separating the characters from each other if the characters are close to each other.

After performing the operations here, we have to options foe segmentation of characters either we use region props () function for getting the bounding box of each character and then crop it or we just get the highest row and highest column of each character and then crop it. Each cropped character is then resized and stored in row matrix respectively. This dataset is used as the input of the trained neural network for testing either the characters matched or not.

4.1.3 Region Extraction

After that we need to extract the plate region from the scene or image. Plate region extraction is the first stage in this algorithm. Image captured from the camera is first converted to the binary image consisting of only 1's and 0's (only black and white) by thresholding the pixel values of 0 (black) for all pixels in the input image with luminance less than threshold value and 1 (white) for all other pixels. Thus, converting color image to Black-Grey color for better extraction.

4.1.4 Character Recognition Using EasyOCR

After the segmentation process the last step is the character recognition. For this step the output of the segmentation process is used as the input. Means the segmented characters' output matrix is feed to the neural network and neural network make some processing on it and give the results as text. Before recognition algorithm, the characters are normalized.

Normalization is to refine the characters into a block containing no extra white spaces (pixels) in all the four sides of the characters. For matching the characters with the database, input images must be equal-sized with
the database characters. Here the characters are fit to 20 * 20. The extracted characters cut from plate and the characters on database are now equal-sized. Because, of the similarities of some characters, there may be some errors during recognition. The confused characters mainly are B and 8, E and F, D and O, S and 5, Z and 2. To increase the recognition rate, some criteria tests are used in the system for the confused characters defining the special features of the characters. With these features of characters and applied tests during recognition algorithm, recognition rate is increased with the minimum error.

CHAPTER 5

SYSTEM IMPLEMENTATION

In this chapter, the System Implementation of Automatic Number-Plate Recognition is explained in detail.

5.1 System Description

LIST OF MODULES

- Collecting Dataset
- Pre-processing
- Feature Extraction
- Training the Model
- Testing the Model

5.1.1 Collecting Dataset

There are basically a greater number of vehicles travelling around for each and every minute and for recognizing the number plate of each and every image of vehicles in the real-world, the model has to be trained well. So, for training the model we need datasets of vehicles with number plate in it. These datasets must consist all kinds of images of vehicle with good lighting, poor lighting, good quality image and bad ones. Thus, this helps the model to be pre-trained well. In total there are 433 images of vehicle as datasets.

5.1.2 Pre-Processing

The data preparation stage is when data is profiled, formatted and structured as needed to make it ready for training the model. This is the stage where the appropriate characteristics and attributes of data are selected. This stage is likely to have a direct impact on the execution time and results. This is also at the stage where data is categorized into two groups – one for training the ML model and the other for evaluating the model. Pre-processing of data by normalizing, eliminating duplicates and making error corrections is also carried out at this stage. Pre-processing is an improvement of the image data that suppresses unwilling distortions or enhances some image features important for further processing, although geometric transformations of images (e.g. rotation, scaling, translation). Transformations on the raw data before it is fed to the machine learning or deep learning algorithm. For instance, training a convolutional neural network on raw images will probably lead to bad classification performances.

5.1.3 Training Model

Models can be trained to benefit manufacturing processes in several ways. The ability of models to process large volumes of data can help manufacturers identify anomalies and test correlations while searching for patterns across the data feed. It can equip with predictive maintenance capabilities and minimize planned and unplanned downtime. A training model is a dataset that is used to train an algorithm. It consists of the sample output data and the corresponding sets of input data that have an influence on the output. The training model is used to run the input data through the algorithm to correlate the processed output against the sample output. The result from this correlation is used to modify the model. This iterative process is called “model fitting”.

The accuracy of the training dataset or the validation dataset is critical for the precision of the model. Model training in machine language is the process of feeding an algorithm with data to help identify and learn good values for all attributes involved. There are several types of machine learning models, of which the most common ones are

supervised and unsupervised learning. Supervised learning is possible when the training data contains both the input and output values. Each set of data that has the inputs and the expected output is called a supervisory signal. The training is done based on the deviation of the processed result from the documented result when the inputs are fed into the model.

Unsupervised learning involves determining patterns in the data. Additional data is then used to fit patterns or clusters. This is also an iterative process that improves the accuracy based on the correlation to the expected patterns or clusters. There is no reference output dataset in this method.

5.1.4 Testing Model

Pre-train tests: This type of test is performed early on and allows you to catch bugs before running the model. They do not need training parameters to be run. An example of a pre-train test is a program that checks whether there are any labels missing in your training and validation datasets.

Post-train tests: These tests are performed on a trained model and check whether it performs correctly. They allow us to investigate the logic behind the algorithm and see whether there are any bugs there.

Therefore, by testing the model we can know that the model has trained well for its application and if not, let the model train for extra times more than before. The more times you use the same data to make decisions about hyperparameter settings or other model improvements, the less confident you are that the model will generalize well on new, unseen data. So, it is a good idea to collect more data to ‘freshen up’ the test set and validation set.

CHAPTER 6

RESULT AND CODING

6.1 Sample Code

```
import os

CUSTOM_MODEL_NAME = 'my_ssd_mobnet'

PRETRAINED_MODEL_NAME ='ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8'

PRETRAINED_MODEL_URL='http://download.tensorflow.org/models/object_detection/tf2/20200711/ssd_mobilenet_v2_fpnlite_320x320_coco17_tpu-8.tar.gz'

TF_RECORD_SCRIPT_NAME = 'generate_tfrecord.py'

LABEL_MAP_NAME = 'label_map.pbtxt

paths = {

   'WORKSPACE_PATH': os.path.join('Tensorflow', 'workspace'),

   'SCRIPTS_PATH': os.path.join('Tensorflow','scripts'),

   'APIMODEL_PATH': os.path.join('Tensorflow','models'),

   'ANNOTATION_PATH': os.path.join('Tensorflow', 'workspace','annotations'),

   'IMAGE_PATH': os.path.join('Tensorflow', 'workspace','images'),

   'MODEL_PATH': os.path.join('Tensorflow', 'workspace','models'),

   'PRETRAINED_MODEL_PATH': os.path.join('Tensorflow', 'workspace','pre-trained-models'),

   'CHECKPOINT_PATH': os.path.join('Tensorflow', 'workspace','models',CUSTOM_MODEL_NAME),
```

```
    'OUTPUT_PATH': os.path.join('Tensorflow',
'workspace','models',CUSTOM_MODEL_NAME, 'export'),

    'TFJS_PATH':os.path.join('Tensorflow',
'workspace','models',CUSTOM_MODEL_NAME, 'tfjsexport'),

    'TFLITE_PATH':os.path.join('Tensorflow',
'workspace','models',CUSTOM_MODEL_NAME, 'tfliteexport'),

    'PROTOC_PATH':os.path.join('Tensorflow','protoc')

 }

files = {

    'PIPELINE_CONFIG':os.path.join('Tensorflow', 'workspace','models',
CUSTOM_MODEL_NAME, 'pipeline.config'),

    'TF_RECORD_SCRIPT': os.path.join(paths['SCRIPTS_PATH'],
TF_RECORD_SCRIPT_NAME),

    'LABELMAP': os.path.join(paths['ANNOTATION_PATH'],
LABEL_MAP_NAME) }

import os

import tensorflow as tf

from object_detection.utils import label_map_util

from object_detection.utils import visualization_utils as viz_utils

from object_detection.builders import model_builder

from object_detection.utils import config_util

gpus = tf.config.list_physical_devices('GPU')

if gpus:
```

```
    try:
        tf.config.experimental.set_virtual_device_configuration(gpus[0],
[tf.config.experimental.VirtualDeviceConfiguration(memory_limit=5120)])
    except RunTimeError as e:
        print(e)

configs = config_util.get_configs_from_pipeline_file(files['PIPELINE_CONFIG'])
detection_model = model_builder.build(model_config=configs['model'],
is_training=False)

ckpt = tf.compat.v2.train.Checkpoint(model=detection_model)
ckpt.restore(os.path.join(paths['CHECKPOINT_PATH'], 'ckpt-23')).expect_partial()

@tf.function
def detect_fn(image):
    image, shapes = detection_model.preprocess(image)
    prediction_dict = detection_model.predict(image, shapes)
    detections = detection_model.postprocess(prediction_dict, shapes)
    return detections
IMAGE_PATH = os.path.join(paths['IMAGE_PATH'], 'test','Cars376.png')
img = cv2.imread(IMAGE_PATH)
image_np = np.array(img)
```

```
input_tensor = tf.convert_to_tensor(np.expand_dims(image_np, 0), dtype=tf.float32)
detections = detect_fn(input_tensor)

num_detections = int(detections.pop('num_detections'))
detections = {key: value[0, :num_detections].numpy()
               for key, value in detections.items()}
detections['num_detections'] = num_detections

detections['detection_classes'] = detections['detection_classes'].astype(np.int64)

label_id_offset = 1
image_np_with_detections = image_np.copy()

viz_utils.visualize_boxes_and_labels_on_image_array(
          image_np_with_detections,
          detections['detection_boxes'],
          detections['detection_classes']+label_id_offset,
          detections['detection_scores'],
          category_index,
          use_normalized_coordinates=True,
          max_boxes_to_draw=5,
```

```
            min_score_thresh=.8,
            agnostic_mode=False)

plt.imshow(cv2.cvtColor(image_np_with_detections, cv2.COLOR_BGR2RGB))
plt.show()
image = image_np_with_detections
scores = list(filter(lambda x: x> detection_threshold, detections['detection_scores']))
boxes = detections['detection_boxes'][:len(scores)]
classes = detections['detection_classes'][:len(scores)]
def filter_text(region, ocr_result, region_threshold):
    rectangle_size = region.shape[0]*region.shape[1]

    plate = []
    for result in ocr_result:
        length = np.sum(np.subtract(result[0][1], result[0][0]))
        height = np.sum(np.subtract(result[0][2], result[0][1]))

        if length*height / rectangle_size > region_threshold:
            plate.append(result[1])
    return plate
def ocr_it(image, detections, detection_threshold, region_threshold):
```

```
    scores = list(filter(lambda x: x> detection_threshold, detections['detection_scores']))
    boxes = detections['detection_boxes'][:len(scores)]
    classes = detections['detection_classes'][:len(scores)]

    width = image.shape[1]
    height = image.shape[0]

    for idx, box in enumerate(boxes):
        roi = box*[height, width, height, width]
        region = image[int(roi[0]):int(roi[2]),int(roi[1]):int(roi[3])]
        reader = easyocr.Reader(['en'])
        ocr_result = reader.readtext(region)
        text = filter_text(region, ocr_result, region_threshold)
        plt.imshow(cv2.cvtColor(region, cv2.COLOR_BGR2RGB))
        plt.show()
        print(text)
        return text, region
text, region = ocr_it(image_np_with_detections, detections, detection_threshold,
region_threshold)
```

6.2 SAMPLE SCREENSHOTS

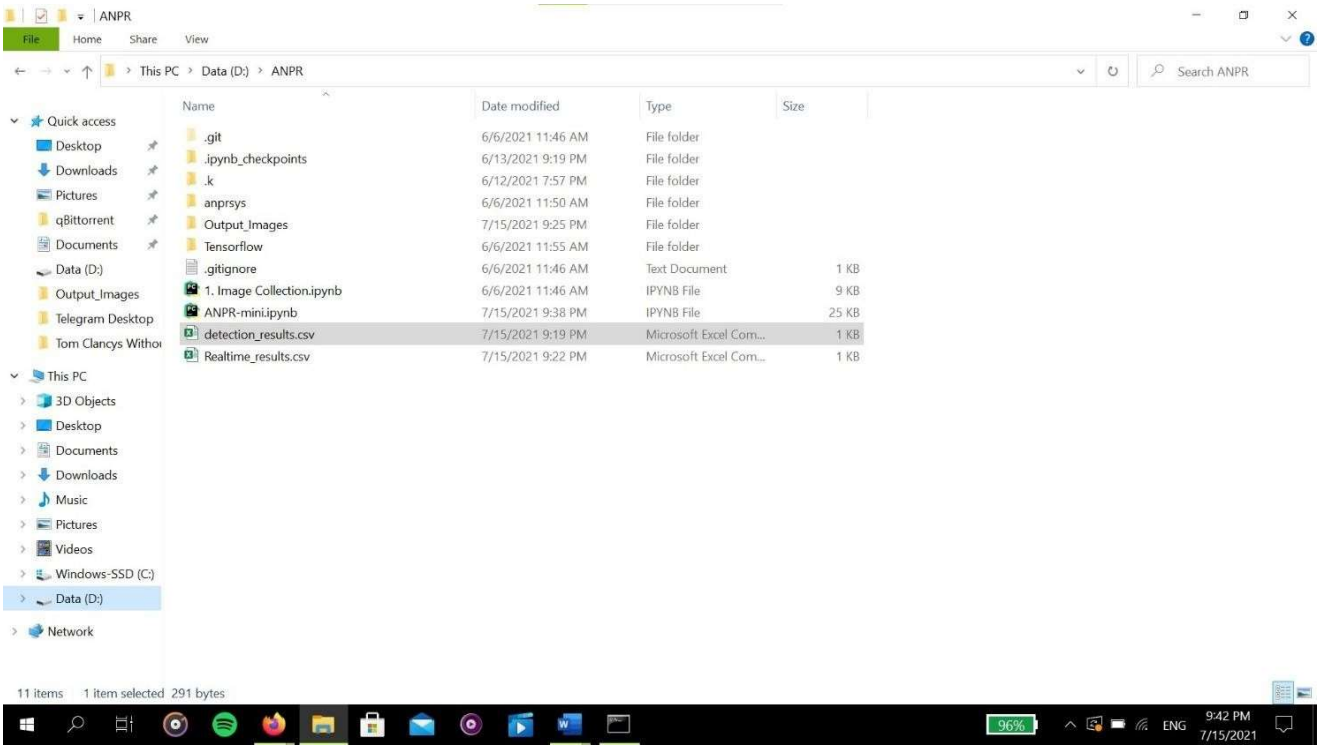

Figure 6.2.1 List of Components

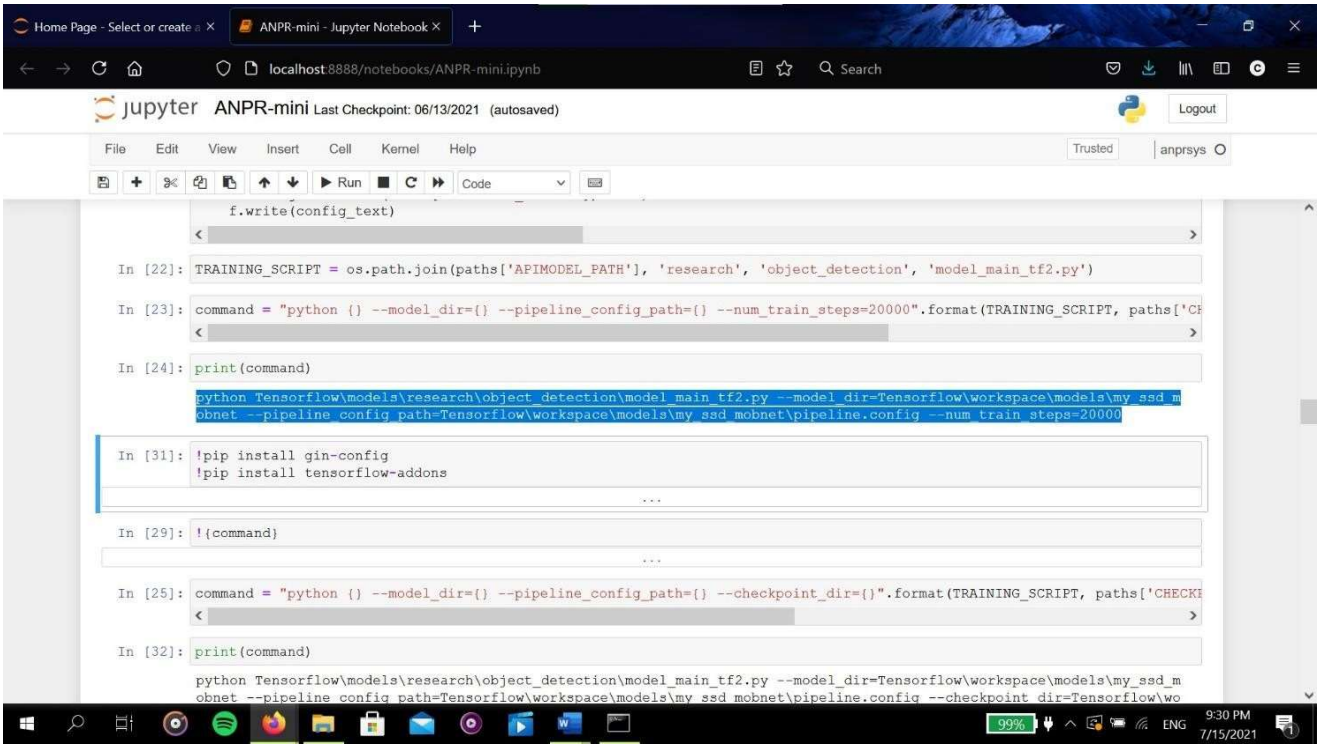

Figure 6.2.2 Command for model training

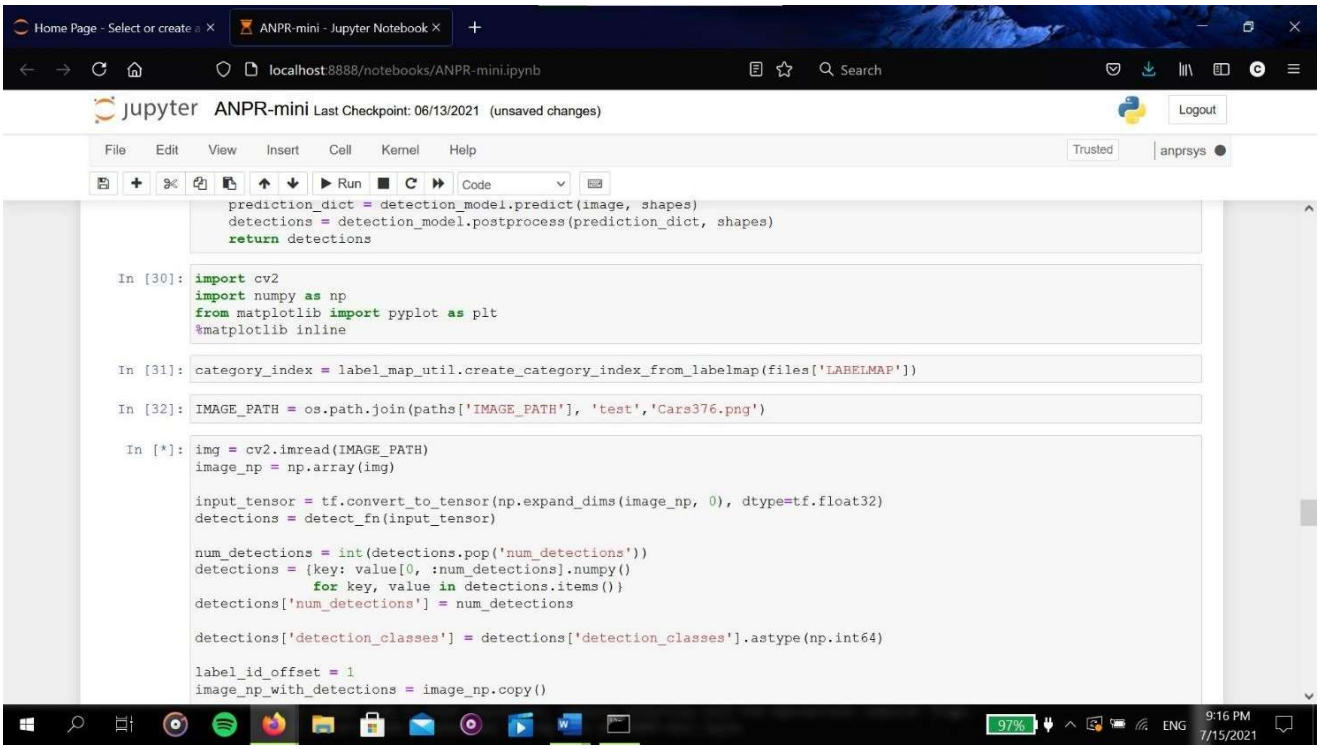

Figure 6.2.3 Input Image Path

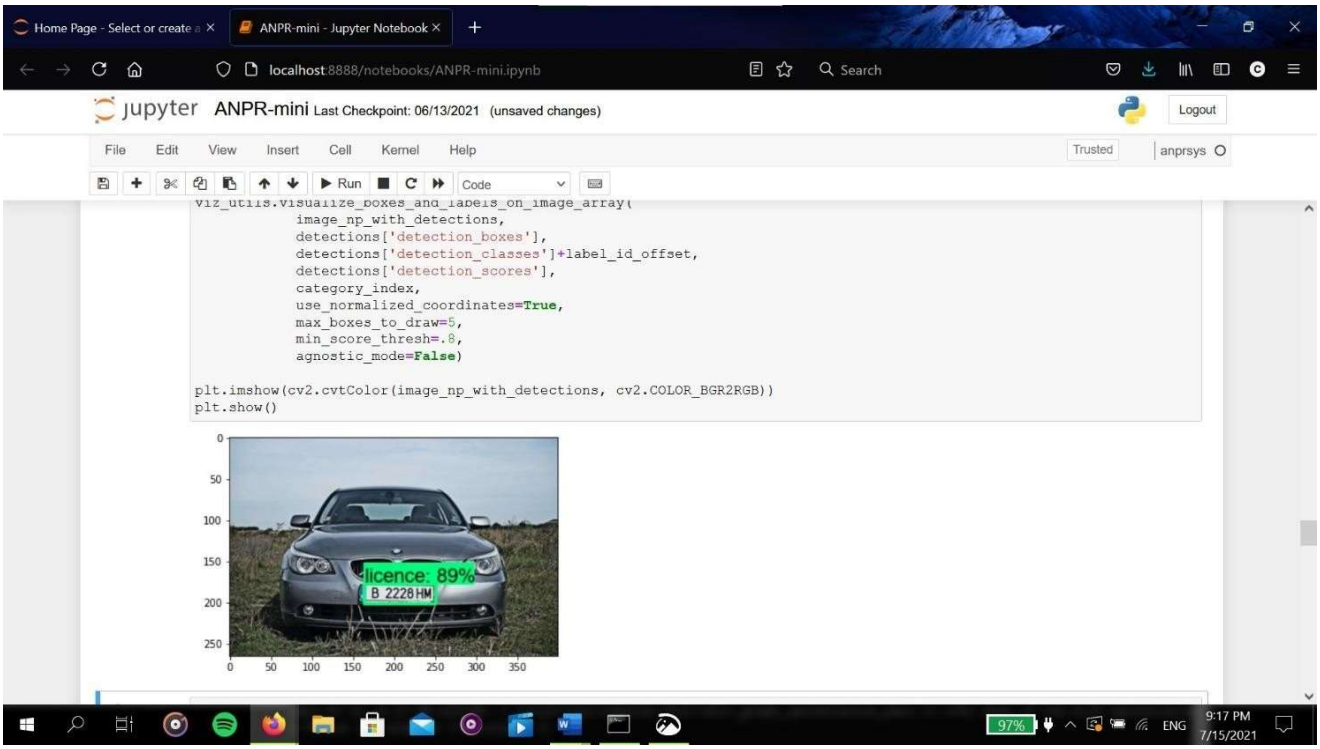

Figure 6.2.4 License Detection

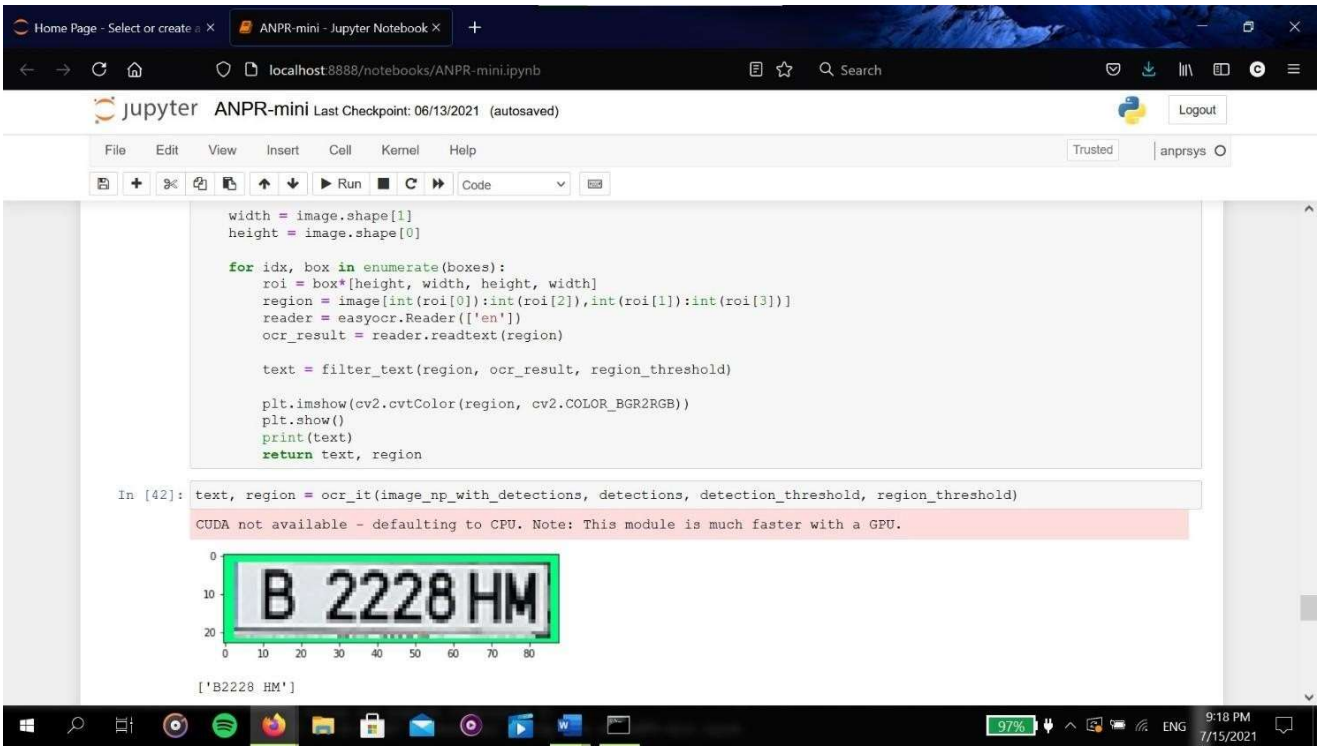

Figure 6.2.5 Text extraction

Figure 6.2.6 Real-Time Detection

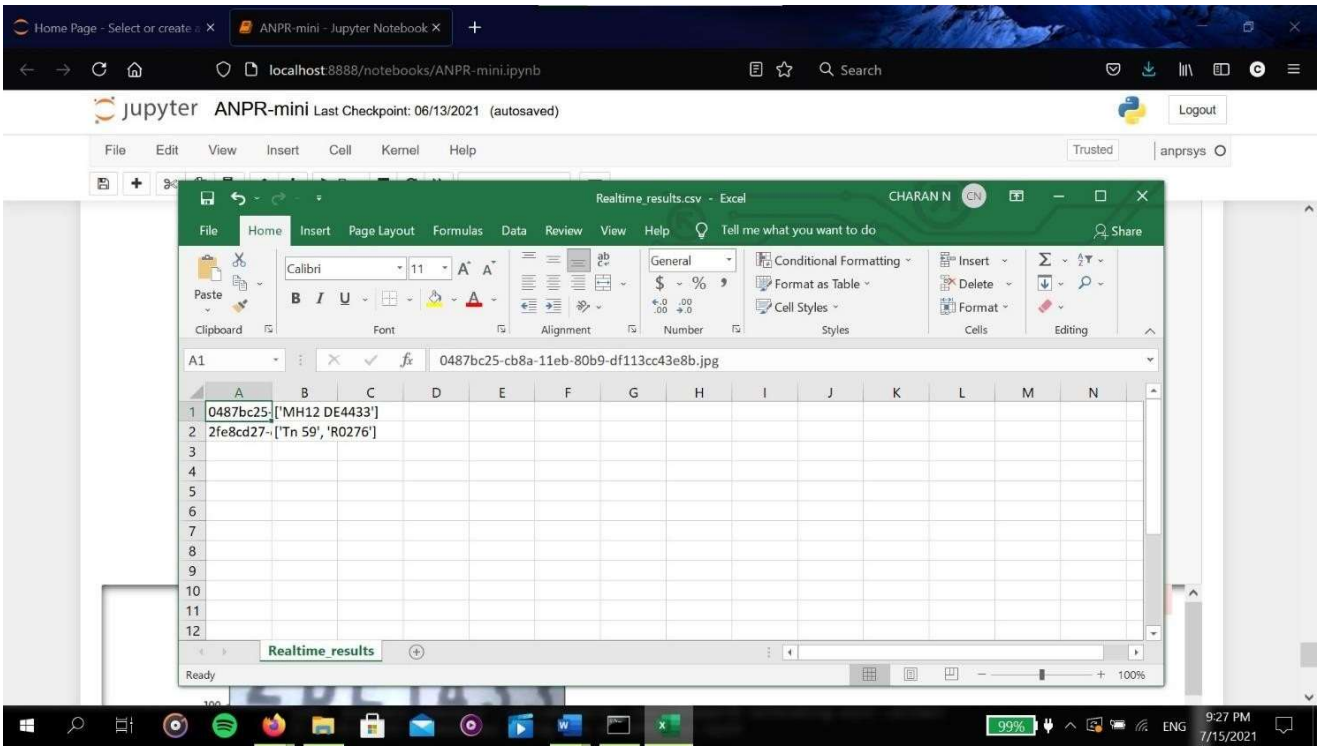

Figure 6.2.7 Realtime Result Stored

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 CONCLUSION

In this paper, the Automatic Number Plate Recognition system using EasyOCR is presented. And by performing this model, we are able to achieve around 85% accuracy. This system uses image processing techniques for recognition of the vehicle from the database stored in the computer. The system works satisfactorily for wide variation of conditions and different types of number plates. The system is implemented and executed in Jupyter Notebook and performance is tested on genuine images. This ANPR system works quite well however, there is still room for improvement. This ANPR system speed can be increase with high resolution Images and Cameras. Which can be able to capture clear images of the vehicle. The OCR method is sensitive to misalignment and to different sizes, so we have to create different kind of templets for

different RTO specifications. The statistical analysis can also be used to define the probability of detection and recognition of the vehicle number plate. At present there are certain limits on parameters like speed of the vehicle, script on the vehicle number plate, skew in the image which can be removed by enhancing the algorithms further.

7.2 FUTURE WORK

Today advances technology took Automatic Number Plate Recognition (ANPR) systems from hard to set up, limited expensive, fixed based applications to simple mobile ones in which “point to shoot” method can be used. This is possible because of the creation of software which ran on cheaper PC based and also non specialist hardware in which their no need to give pre-defined direction, angels, speed and size in which the plate would be passing the camera field of view. Also Smaller cameras which can read license plates at high speed, along with smaller, more durable processors that can fit in police vehicles, allowed law enforcement officers to patrol daily with the benefit of license plate recognition in real time.

REFERENCES

1. www.kaggle.com/andrewmvd/car-plate (Dataset).

2. C. Anagnostopoulos, T. Alexandropoulos, V. Loumos and E.Kayafas, " Intelligent traffic management through MPEG-7vehicle flow surveillance," in Proceedings of IEEE International Symposium on Modern Computing, 2006, pp. 202-207.

3. N. Mani, and B. Srinivasan, "Application of artificial neural network model for optical character recognition," presented at the International Conference on Systems, Man, and Cybernetics, 1997.

4. R. P. van Heerden, E. C. Botha, Optimization of Vehicle License Plate Segmentation and Symbol Recognition, Department of Electrical, Electronic and Computer engineering, University of Pretoria, South Africa, 2010.

5. F. M. Rodrguez, X. F. Hermida, New Advances in Automatic Reading of V.L.P.s (Vehicle License Plates), Proceedings in SPC-2000 (Signal Processing and Communications), Marbella, September 2000.

6. F.M. Rodrguez, M. G. Saburido, J. L. A. Castro, New Methods forAutomatic Reading of V.L.P.s (Vehicle License Plates), SPPRA- 2002 (Signal Processing Pattern Recognition and Applications), June 2002.

7. A. E. Savakis, Adaptive Document Image Thresholding Using foreground and Background Clustering, International Conference on Image Processing, October 1998.

8. Ondrej Martinsky, Algorithmic and mathematical Principles of automatic number plate Recognition systems, Brno University of technology, 2007.

9. Sushruth Shastry, Gunasheela G, Thejus Dutt, Vinay D S andSudhir Rao Rupanagudi,i - A novel algorithm for OpticalCharacter Recognition (OCR), IEEE