

**“VEHICLE TYPE DETECTION &  
LICENSE PLATE EFFECTS TESTBENCH”**

*Project Report submitted for the degree of*  
**Bachelor of Technology**

*In*  
**Computer Science and Engineering**

*By*  
**Anurag Jain (131272)**

*Under the supervision of*  
**Mr. Anoop Prabhu**  
(CTO)

**Md. Danish Ansari**  
(Software Developer)

And

**Dr. Milind Padalkar**  
(Sr. Research Engineer)

*To*



Department of Computer Science & Engineering and Information  
Technology

**Jaypee University of Information Technology Waknaghat,  
Solan-173234, Himachal Pradesh**

# DECLARATION

I, Anurag Jain hereby declare that the work presented in this report entitled **“VEHICLE TYPE DETECTION & LICENSE PLATE EFFECTS TESTBENCH”** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology, **Jaypee University of Information Technology, Wagnaghat** is an authentic record of our original work carried out over a period from February 2017 to June 2017 under the supervision of **Mr. Anoop Prabhu, Md. Danish Ansari and Dr. Milind Padalkar**. The matter embodied in the report has not been submitted to any other Institute for the award of any other degree or diploma.

Anurag Jain, 131272

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Mr. Anoop Prabhu

CTO

Vehant Technologies Pvt. Ltd.

Dr. Milind Padalkar

Sr. Research Engineer

Vehant Technologies Pvt. Ltd.

Md. Danish Ansari

Software Developer

Vehant Technologies Pvt. Ltd.

# CERTIFICATE

This is to certify that the project entitled “**VEHICLE TYPE DETECTION & LICENSE PLATE EFFECTS TESTBENCH**” submitted by Anurag Jain to **Jaypee University of Information Technology, Wagnaghat** is a record of bonafide research work under my supervision and consider it worthy for the award of the degree of Bachelor of Technology of the Institute.

Mr. Anoop Prabhu

CTO

Vehant Technologies Pvt. Ltd.

Dr. Milind Padalkar

Sr. Research Engineer

Vehant Technologies Pvt. Ltd.

Md. Danish Ansari

Software Developer

Vehant Technologies Pvt. Ltd.

# ACKNOWLEDGEMENT

It is my pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced my thinking , behavior and acts during the course of study.

I am thankful to Mr. Anoop Prabhu, Md. Danish Ansari and Dr. Milind Padalkar for their support, cooperation, and motivation provided to me during the period for constant inspiration, presence and blessings. I am privileged to experience a sustained enthusiasm and involved interest from their side. This fueled my interest even further and encouraged me to boldly step into what was a totally dark and unexplored expanse before. I would also like to thank my team-mate and friends who were ready with their inputs at all times, whether it was an offhand comment to encourage me or a constructive piece of criticism. Last but not least, I would like to thank Vehant Technologies and JUIT staff members and the institute, in general, for extending a helping hand at every juncture of need.

Anurag Jain, 131272

# Table of Contents

Chapter	Title	Page No.
	List of Abbreviations	vi
	List of Figures	vii
	List of Tables	vii
	About Company	ix
1	Introduction	
	1.1 Introduction	1
	1.2 Motivation	2
	1.3 Problem Statement	
	1.3.1 Vehicle Type Detection TestBench	3
	1.3.2 License Plate Effects TestBendh	4
	1.3.3 What we want?	5
	1.3.4 Why LiveView of Camera & recording is required	6
2	Literature Survey	
	2.1 Previous works on Type Detection	7
	2.2 Super Resolution of Images	8
	2.3 Conventional ANPR System	8
3	System Development	
	3.1 Software Requirements	
	3.1.1 Language Used	10
	3.1.2 Libraries and Software Used	10
	3.1.3 Tools Used	12
	3.2 Hardware Requirements	14
	3.3 Implementation	
	3.3.1 Classification of Vehicles	14
	3.3.2 Different Effects Used & their Parameters	14
	3.3.3 Proposed Approach	18

4	Performance Analysis	
	4.1 Performance Measures	20
	4.2 Result Analysis	20
	4.3 Output Screenshots	
	4.3.1 Vehicle Type Detection TestBench	22
	4.3.2 License Plate Effects TestBench	23
5	Conclusion	
	5.1 Conclusion	25
	5.2 Future Work	25
	References	26

## List of Abbreviations

<b>Acronym</b>	<b>Definition</b>
TB	TestBech
LP	License Plate
ANPR	Automatic Licence Plate Recognition
LPR	License Plate Recognition
OCR	Optical Character Reader
VTD	Vehicle Type Detection
AUTOL	AUTO Low
AUTOH	AUTO High
LMV	Low Motor Vehicle
HMV	High Motor Vehicle
HMVH	HMV High
TWOWH	Two Wheeler
TWOWHH	Two Wheeler High
TWOWHL	Two Wheeler Low

# List of Figures

<b>Title</b>	<b>Page No.</b>
Fig 2.1 Conventional ANPR System	9
Fig 3.1 Approach for License Plate TestBench	18
Fig 3.2 Approach for Vehicle Type Detection TestBench	19
Fig 4.1 Vehicle Type Detect Screen	22
Fig 4.2 Result Screen	22
Fig 4.3 LP Effects Screen	23
Fig 4.4 Super Resolution Screen	23



## List of Tables

<b>Title</b>	<b>Page No.</b>
Table 3.1 Parameters of Random Blur	15
Table 3.2 Parameters of Add Shadows	15
Table 3.3 Parameters of Random Scratch	16
Table 3.4 Parameters of Motion Blur	16
Table 3.5 Parameters of Tail Light	16
Table 3.6 Parameters of Erode Numbers	17
Table 3.7 Parameters of Dilate Numbers	17
Table 3.8 Parameters of Suppress Contrast	17
Table 3.9 Parameters of Join Characters	17
Table 3.10 Parameters of Salt & Pepper Noise	18
Table 4.1 Result of VTD Site1	20
Table 4.2 Wrong Detection of VTD Site1	21
Table 4.3 Result of VTD Site2	21
Table 4.4 Wrong Detection of VTD Site2	21
Table 4.5 Output of different Effects	24

## **ABOUT THE COMPANY**

### **(VEHANT TECHNOLOGIES)**

Vehant Technologies formerly known as KritiKal SecureScan, pioneer in indigenously developed Physical Security, Surveillance and Traffic Monitoring Systems, designed and developed its state of the art products and solutions to meet the demands of global standards, features, quality and continuously changing technology. Vehant Technologies is the KritiKal Group company which was incubated from Indian Institute of Technology (IIT)-Delhi in the year 2002.

Vehant Technologies is the leading manufacturer of Under Vehicle Scanning Systems (UVSS) in India and catering the requirements of all verticals. Vehant aims to provide comprehensive solution to its customers, carry out research and explore technologies in the area of security, surveillance and monitoring space.

Vehant manufactures its cutting-edge products and solutions for the real world, based on its deep understanding of global security issues and challenges. Vehant has designed & developed a range of increasingly integrated security and monitoring products to create complete solution stack tailored to individual requirements.

# CHAPTER 1

## INTRODUCTION

### 1.1 Introduction

Traffic control and vehicle owner identification has become major problem in every country. Sometimes it becomes difficult to identify vehicle owner who violates traffic rules and drives too fast. Therefore, it is not possible to catch and punish those kinds of people because the traffic personal might not be able to retrieve vehicle number from the moving vehicle because of the speed of the vehicle. Due to the unlimited increase of cars and transportation systems which make it impossible to be fully managed and monitored by humans, examples are so many like traffic monitoring, tracking stolen cars, managing parking toll, red-light violation enforcement, border and customs checkpoints.

Therefore, there is a need to develop Automatic Number Plate Recognition (ANPR) system as a one of the solutions to this problem. There are numerous ANPR systems available today. These systems are based on different methodologies but still it is really challenging task as some of the factors like high speed of vehicle, non-uniform vehicle number plate, language of vehicle number and different lighting conditions can affect a lot in the overall recognition rate. Most of the systems work under these limitations.

Video surveillance plays an increasing role in public life. More and more highways, intersections or whole cities use video surveillance in order to regulate the huge traffic volume. In most cases, an ordinary traffic census system is used which counts passing vehicles by using infrared or simple computer vision approaches. Apart from the number of vehicles, it would be interesting to know the type of vehicle which passed the observation point. This information could greatly influence decisions made for the traffic system. Highways could be build according

to the type of vehicles that will use them. New intersections or traffic lights can be installed which react intelligently to the currently dominating vehicle types.

My project is to generate a data set which can be used for training purposes and we can then improve the efficiency of our ANPR modules and Vehicle Type Detection modules.

## **1.2 Motivation**

For traffic monitoring video surveillance is the best option to use in real time. In traffic monitoring there are many things to consider like speed violation, red light crossing or violating any traffic rules. Magnetic loop detectors are used to count the vehicles that pass over them, but our proposed systems provide much more different information that will help in smooth traffic control. We need a system which can detect which vehicle has violated the traffic rules that magnetic loop detectors can't find. Other useful functionalities like cars, bikes etc can be differentiated, more traffic components can be observed for violations like crossing red light, parking in no parking area zone etc. Cameras are less disruptive to install than loop detectors. To know the detail of vehicle which violated the traffic rule there is only one element that needs to be known i.e. LICENSE PLATE.

We have an application which detects the license plate. The algorithm that we have can detect 80% of vehicle license plate. So, how to know that how many are missing and how to add them? The License plate number detected is correctly interpreted in only 80% of the cases, so how to improve this? One thing that comes in mind that one has to manually review the video frame by frame and add the data for those vehicles in database. It is obvious that it will take a lot of time. So, these are the reasons that motivated me to do this project. To save manual labour and time we can save those frames for which license plate was not detected. Now, one can know with just one click that which vehicle was not detected by license plate detector.

Despite the large amount of literature on vehicle detection and tracking, there has been relatively little work done in the field of vehicle classification. This is

because vehicle classification is an inherently hard problem. Moreover, detection and tracking are simply preliminary steps in the task of vehicle classification. Given the wide variety of shapes and sizes of vehicles within a single category alone, it is difficult to categorize vehicles using simple parameters. This task is made even more difficult when multiple categories are desired. In real-world traffic scenes, occlusions, shadows, camera noise, changes in lighting and weather conditions, etc. are a fact of life. In addition, stereo cameras are rarely used for traffic monitoring. This makes the recovery of vehicle parameters – such as length, width, height etc, even more difficult given a single camera view. The inherent complexity of stereo algorithms and the need to solve the correspondence problem makes them unfeasible for real-time applications.

For the type detection of the vehicles we have a module written, but this module is not tested. So for the testing purposes and increase the efficiency, either we manually feed the system with images or a traffic recording and the result for each transaction is tested and accuracy is computed manually or we can just make use of a GUI application which can classify the results in different segments on one click and accuracy can then be checked and results can be interpreted for improvement.

## **1.3 Problem Statement**

### **1.3.1 Vehicle Type Detection TestBench**

To classify the vehicles is very important for the traffic monitoring systems. For example, at the time of speed violation there is different speed limit for different classes of the vehicles and if we are trying to combine the speed violation detection and automatic challan generation system, then this would create a lot of difficulties. If this happens and a vehicle passes by with a speed of 60 kmph which is a violation for a HMTV vehicle like truck or bus and normal for a LMV vehicle like Car or Jeep. So, what to do now? Generate the challan or not. To handle such problem we can make a person sit and let him validate for all such cases whether to generate the challan or not. But this will increase human labour and it will be time consuming. If

our systems are intelligent enough to detect the type of vehicle we can directly generate the challan for such cases.

Our build module for Vehicle Type Detection has not been tested yet. So to know about the efficiency of this module and do other useful work with the data related to this we require a TestBench application.

### **1.3.2 License Plate Effects TestBench**

In traffic monitoring there are a lot of violations to monitor like red light crossing, speed violation and other traffic violations. To know the detail of vehicle which violated the traffic rule there is only one element that needs to be known i.e. LICENSE PLATE. So License Plate Number becomes the most important parameter in traffic monitoring. So we cannot afford any kind of errors in reading the license plate number of the vehicles.

Suppose a vehicle with license number RJ14CQX252 make a violation and our ANPR wrongly interprets it as RJ14COX252 and we generate a challan corresponding to the detected license number, then this can cause a problem for an innocent person and the defaulter will not get punished. Let's take another case in which the detected license plate number is so wrong that it does not exist. So in all such cases what to do?

Our already written module of ANPR works with an accuracy of 80%. We are aiming to improve this accuracy. For doing this we are trying to impose different effects on the license plate images available to us and make them diverse enough i.e. a variety of bad kind of images which we can come across in real world. Then these images will be used to improve our modules.

Also we have some Low Resolution images of the License Plate and ANPR make faults with them, for this we are trying to convert the Low Resolution images to High resolution images and improve the accuracy. For generating this data set we require a TestBench application.

### 1.3.3 What we want?

For Vehicle Type Detection TestBench:

**Input:** A camera device or a video.

**Output:** A TestBench application with following features:

1. To run the Vehicle Type Detection module on the recording and create the transactions and save in a directory. The application should be compatible with all types of camera like backfly camera, syn camera and ip camera.
2. To load the generated transactions one by one on the screen with the transaction image and details. Also provide the user to correct the Vehicle Type if it is wrongly detected.
3. To classify all the transactions according to the type detected and modified by the user i.e. save records on each Vehicle type in their corresponding directories.
4. Show the total accuracy of the module and accuracy of each type for analysis that what are the areas we need to improve.
5. Result Screen showing the pie chart of each Vehicle type and its segments depicting what percent of total records of that type are detected as which type.
6. Export the table showing the detailed analysis of accuracy.

For License Plate Effects TestBench:

**Input:** A directory with images of different License Plate.

**Output:** A TestBench application with following features:

1. Give user an option to choose one of the all effects to modify the image.
2. Give user the option to set the required parameters for the effect chosen.
3. Show the Original image of the License plate and the after effect image, so that comparison can be made.
4. Save option to save the after effect image.

#### **1.3.4 Why LiveView of Camera and recording is required?**

Earlier video were not analyzed onsite, video stored on local storage attached with camera are carried to company. Videos were preprocessed with different libraries like FFMPEG but there was no user friendly application to carry all these processes. After the video was properly made then it was analyzed by us not the users i.e. Traffic policeman. All the data and information were provided by us. Also with this recording we can generate required License Plate Images for our analysis.



## Chapter 2

# LITERATURE SURVEY

### 2.1 Previous works on Type Detection

Peijin Ji, Lianwen Jin and Xutao Li used a partial Gabor filter bank for vision based vehicle type classification [3]. They extracted features from both edge images and grey images from the side view of the car by using a partial Gabor filter bank and different sampling methods. The dimension of the feature space was reduced by using Principal Component Analysis. A minimum distance classifier derived from the Bayes' decision theory was used to classify the vehicle into five different categories: Sedan, van, hatchback sedan, bus and van truck. They ran experiments with a testing database of total 1196 vehicle side views and achieved a maximum recognition accuracy of 95.17%.

Peter Shin, Hector Jasso, Sameer Tilat, Neil Cotofana and Tony Fountain from the Department of Structural Engineering at the University of California used strain histories from installed bridge-deck panels and tried to classify the passing vehicles using Naive Bayesian, a Neural Network and a Support Vector Machine at a time [4]. They collected strain histories from 2100 vehicles, normalized them and trained the different analytical methods by using about 400 vehicles from each of the five categories: Small vehicle, medium truck, bus, 3-axle truck and combination truck. In their experiments, they achieved a recognition accuracy of 94.8% by using the Support Vector Machine technique.

Thiang, Resmana Lim and Andre Teguh Guntoro described car recognition of various car types during daylight and at night based on Gabor wavelets [5]. They extracted Gabor features and created a database of four template images for each vehicle category (sedan, van, pickup). They matched an input image by computing the similarity value to each template. The class of the template with the highest

similarity value was chosen. In their experiment, they used the side view of 44 unknown vehicles. Their system achieved an average recognition rate of 93.88%.

## 2.2 Super Resolution of Images

With the advent of modern electronic gadgets like PDAs, cellular phones, and digital cameras, the scope of document imaging has increased. Document image analysis systems are becoming increasingly visible in everyday life. For instance, one may be interested in processing, storing, understanding a class of document images obtained by cellular phones [9]. Processing challenges in this class of documents are considerably different from the conventional scanned document images. Many of this new class of documents are characterized by low resolution and poor quality making the immediate recognition practically impossible. Super resolution provides an algorithmic solution to the resolution enhancement problem by exploiting the image specific *a priori* information.

Super-resolution of low resolution document images is becoming an important pre-requisite for design and development of robust document analysis systems. Large scale camera based book scanners employed in digital libraries could get benefited from resolution enhancement to obtain high OCR accuracies. It is also true with the text embedded in natural scenes, which could be used for indexing their images. Digital video compression algorithms can also benefit from the successful text resolution expansion techniques. Videos are often indexed and retrieved based on embedded text information. The text observed in broadcast videos is often low in resolution. Without enhancement, a simple binarization could completely remove many strokes. In these conditions, it is virtually impossible to do character recognition as most of the OCRs are designed to work at reasonably high resolutions. Resolution enhancement algorithm increase spatial resolution, while maintaining the difference between text and background. It can further assist the recognition in low-resolution text images.

## 2.3 Conventional ANPR System

In last few years, ANPR or license plate recognition (LPR) has been one of the useful approaches for vehicle surveillance. It is can be applied at number of public

places for fulfilling some of the purposes like traffic safety enforcement, automatic toll text collection, car park system and Automatic vehicle parking system. ANPR algorithms are generally divided in four steps as shown in figure.

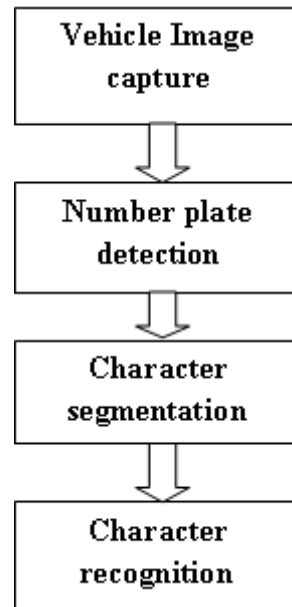


Fig 2.1 Conventional ANPR System

**Vehicle Image Capture:** In this stage, using motion module we identify that a vehicle is there and capture its full image.

**Number Plate Detection:** In this stage, the location of the license plate is identified and the output of this stage will be a sub-image that contains only the license plate.

**Character Segmentation:** This stage is meant for segmentation of the characters from the plate. The output of this stage is a set of monochrome images for each candidate character in plate.

**Character recognition:** The goal of this stage is to recognize and classify the binary images that contain characters received from the previous one. After this stage every character must have a label and an error factor, and this error factor if greater than a predefined value will be used to reject false characters accidentally passed from the previous steps. For the sake of classification, some features must be collected from the characters.

## Chapter 3

# SYSTEM DEVELOPMENT

### 3.1 Software Requirements

#### 3.1.1 Language Used

The module is purely written in C++ programming language, following the Organization's proper coding conventions and object oriented paradigm.

#### 3.1.2 Libraries and Software Used

##### ➤ OpenCV 2.24

OpenCV stands for Open Source Computer Vision Library. It is an open-source BSD-licensed library that includes several hundreds of computer vision algorithms. The document describes the so-called OpenCV 2.x API, which is essentially a C++ API, as opposite to the C-based OpenCV 1.x API. It does automatic memory management. It automatically allocates memory to output data. OpenCV uses saturation arithmetic to deal with processing of image pixels that are often encoded in a compact, 8- or 16-bit per channel. It's true that templates make working easy but sometimes it dramatically increases the compilation time and size. It has its own error handling to signal critical errors.

##### ➤ Glade

Glade is a Research & Development tool to enable quick & easy development of user interfaces for the GTK+ toolkit and the GNOME desktop environment. The Glade application allows us to layout widgets on screen and then save an XML description of the arrangement. Our application can then use the Gtk::Builder API to load that XML file at runtime and obtain a pointer to specifically named widget instances. We can use C++ code to instantiate and arrange widgets. The user interfaces

designed by glade are saved in XML file and by using Gtkbuilder GTK+ objects can be loaded by applications dynamically as needed.

Glade has the following advantages:

1. Less C++ code is required.
2. UI changes can be seen more quickly, so UIs are able to improve.
3. Designers without programming skills can create and edit UIs.

## ➤ **FFMPEG**

FFmpeg is the leading multimedia framework which is able to transcode, decode, encode, demux, mux, stream, filter and play pretty much anything that humans and machines have created. It supports the most obscure ancient formats up to the cutting edge. No matter if they were designed by some standards committee, the community or a corporation. It is also highly portable: FFmpeg runs, compiles, and passes our testing infrastructure FATE across Linux, Microsoft Windows, Mac OS X, the BSDs, Solaris, etc. under a wide variety of build environments, machine architectures, and configurations. The different libraries that it contains are libavcodec, libavutil, libavformat, libavfilter, libavdevice, libswscale and libswresample which can be used by many applications.

- Libavcodec is a library containing decoders and encoders for audio/video codec's.
- Libavformat is a library containing demuxers and muxers for multimedia container formats.
- Libswscale is a library performing highly optimized image scaling and color space/pixel format conversion operations.

## ➤ **Linux CentOS**

CentOS Linux is a Linux distribution derived from the Red Hat Enterprise Linux. Because it is free, CentOS Linux is widely popular with Linux users, web hosts and small businesses. Linux is a very hands-on operating system. We use linux because it is very secure and has a very good directory structure. Also while programming in linux environment we can

set the environment variables at any time from outside the program code, which becomes very useful.

### ➤ **PostgreSQL**

It is an object-relational database management system (ORDBMS) based on POSTGRES, Version 4.2, developed at the University of California at Berkeley Computer Science Department.

It supports a large part of the SQL standard and offers many modern features: complex queries, foreign keys, triggers, updatable views, transactional integrity, and multiversion concurrency control. PostgreSQL can be extended by the user in many ways, for example by adding new data types, functions, operators, aggregate functions, index methods and procedural languages.

And because of the liberal license, PostgreSQL can be used, modified, and distributed by anyone free of charge for any purpose, be it private, commercial, or academic.

## **3.1.3 Tools Used**

### ➤ **Vim Editor**

Vim editor is used for programming on CentOS. Vim is stable and is continuously being developed to become even better.

Among its features are:

- persistent, multi-level undo tree
- extensive plug-in system
- support for hundreds of programming languages and file formats
- powerful search and replace
- integrates with many tools

### ➤ **GDB**

GDB is used for debugging. GDB stands for GNU Debugger. It allows you to know what going inside your program while it executes or what other programs are doing when it crashed.

GDB can do four main kinds of things (plus other things in support of these) to help you catch bugs in the act:

- Start your program, specifying anything that might affect its behavior.
- Add conditional break.
- Examine what has happened, when your program has stopped.
- Alter variables in your program, so you can experiment with correcting the effects of one bug and go on to learn about another.

### ➤ **Doxygen**

Doxygen is the de facto standard tool for generating documentation from annotated C++ sources. It is a tool for writing software reference documentation. The documentation is written within code, and is thus relatively easy to keep up to date. Doxygen can cross reference documentation and code, so that the reader of a document can easily refer to the actual code.

Doxygen can help us in three ways:

1. It can generate an on-line documentation browser (in HTML) and/or an off-line reference manual from a set of documented source files. There is also support for generating output in RTF (MS-Word), PostScript, hyperlinked PDF, compressed HTML, and UNIX man pages. The documentation is extracted directly from the sources, which makes it much easier to keep the documentation consistent with the source code.
2. We can configure doxygen to extract the code structure from undocumented source files. This is very useful to quickly find your way in large source distributions. Doxygen can also visualize the relations between the various elements by means of include dependency graphs, inheritance diagrams, and collaboration diagrams, which are all generated automatically.
3. We can also use doxygen for creating normal documentation.

## 3.2 Hardware Requirements

The minimum hardware requirement specification for developing this project is as follows:

Processor	:	Core i3
RAM	:	4GB or more
Hard Disk	:	10GB or more
Monitor	:	Standard Color Monitor
Keyboard	:	Standard Keyboard
Mouse	:	Standard Mouse

## 3.3 Implementation

### 3.3.1 Classification of Vehicles

We use vehicle dimensions to classify vehicles into four different categories. We use deep learning approach to detect the vehicle type. For the detection we make use of motion parameters of the vehicle and its dimensions and compare it with the values and the category in which the parameters fall in is the result.

The four broadly made categories are: AUTO, CAR, TWO WHEELER and BUS/TRUCK. These categories are then sub categorized on the basis of the position of the License Plate position either on the upper half (high) or the lower half (low).

### 3.3.2 Different Effects Used & their Parameters

In my project, we have used eleven different kinds of Effects which adds different kinds of artefacts to the original License Plate Image and these different effects uses different parameters and have their default values set,



if the user is not satisfied with the output image corresponding to the default parameters, user can change the parameter values accordingly.

### 1) Random Blur

It adds undesired artefacts to the image and makes the text on the image blur, making it some difficult to read for the system to interpret it. This effect is used because sometimes the images of the License plate we capture are not clear enough and contains some undesired artefacts.

Parameter	Details	Range	Step	Default Value
Kernel Size	It is the size of square in pixels whose areas average intensity is calculated for comparison.	3-35	2	7
Block Size	It is the size of square in pixels whose area is to be blurred.	1-25	2	5
Percent	(1 – Percent Value), the total points of the images to be blurred.	0-1	0.01	0.80

Table 3.1 Parameters of Random Blur

### 2) Add Shadows

This effect adds shadow to the license plate as many a times due to the shade of nearby objects the License plate has the effect of shadow. So we want to handle all such cases.

Parameter	Details	Range	Step	Default Value
Location	The side of the image from where shadow is applied. (0-Top, 1-Right, 2-Bottom & 3-Left)	0-3	1	0
Strength	How dark or light the shadow should be. (0 : Absolute Dark & 1 : Absolute Light)	0-1	0.1	0.50
Percent 1	The ratio of length of side1 till where the shadow falls.	0-1	0.01	0.20
Percent 2	The ratio of length of side2 till where the shadow falls.	0-1	0.01	0.10

Table 3.2 Parameters of Add Shadows

### 3) Random Scratch

This effect makes a scratch on the image due to which some characters of the License number are not continuous and a portion leaving two parts of the characters. Our system should be able to handle all such cases.

Parameter	Details	Range	Step	Default Value
Length	The length of the scratch to be made on image.	0-150	1	50
Intensity	How dark or light the scratch should be.	0-255	1	128

Table 3.3 Parameters of Random Scratch

### 4) Motion Blur

Sometimes due to the speed of the vehicle we are not able to capture the clear image of the License plate and get some stretched image which is called as motion blur. With this effect we will generate data set for these cases.

Parameter	Details	Range	Step	Default Value
Length	The length, how much the text stretch due to motion.	1-35	2	9
Theta	The angle in which the motion happens.	0-360	45	0

Table 3.4 Parameters of Motion Blur

### 5) Tail Light

Mostly at the night time we switch on the tail light of our vehicles, due to which a lot of times the License number becomes difficult to read. With different strengths of the tail light we will generate cases of this type.

Parameter	Details	Range	Step	Default Value
Strength	Describes the power with which the tail light makes impact on the numbers.	0-15	0.5	7

Table 3.5 Parameters of Tail Light

### 6) Erode Numbers

This effect makes the characters thin and light.

Parameter	Details	Range	Step	Default Value
Size	Size of the resulting square.	0-15	1	2

Table 3.6 Parameters of Erode Numbers

### 7) Dilate Numbers

This effect makes the characters very strong i.e. fat.

Parameter	Details	Range	Step	Default Value
Size	Size of the resulting square.	0-15	1	2

Table 3.7 Parameters of Dilate Numbers

### 8) Suppress Contrast

Sometimes the contrast of the image is not good, so using this effect we can shift the contrast of the image as per our needs.

Parameter	Details	Range	Step	Default Value
Minimum Value	The minimum intensity visible on the image.	0-255	1	25
Maximum Value	The maximum intensity visible on the image.	0-255	1	100

Table 3.8 Parameters of Suppress Contrast

### 9) Join Characters

This is seen many a times that characters on the License plate are joined and the system can get confused and can read two joined characters as one and interpret it wrong. So we are generating these cases.

Parameter	Details	Range	Step	Default Value
Size	The size to which random data squares are to be scaled to.	1-25	2	5
Percent	The value by which data points are to be spread (1-Percent Value).	0-1	0.01	0.50

Table 3.9 Parameters of Join Characters

### 10) Salt & Pepper Noise

This effect is used to generate cases when noise enters the channel and disturbs the camera image.

Parameter	Details	Range	Step	Default Value
Percent	The percent of total pixels of image whose intensities are to be reverted.	0-1	0.01	0.35

Table 3.1 Parameters of Salt & Pepper Noise

### 11) Compression Artefacts

This effect compresses the image data using some compression techniques.

### 3.3.3 Proposed Approach

For License Plate TestBench:

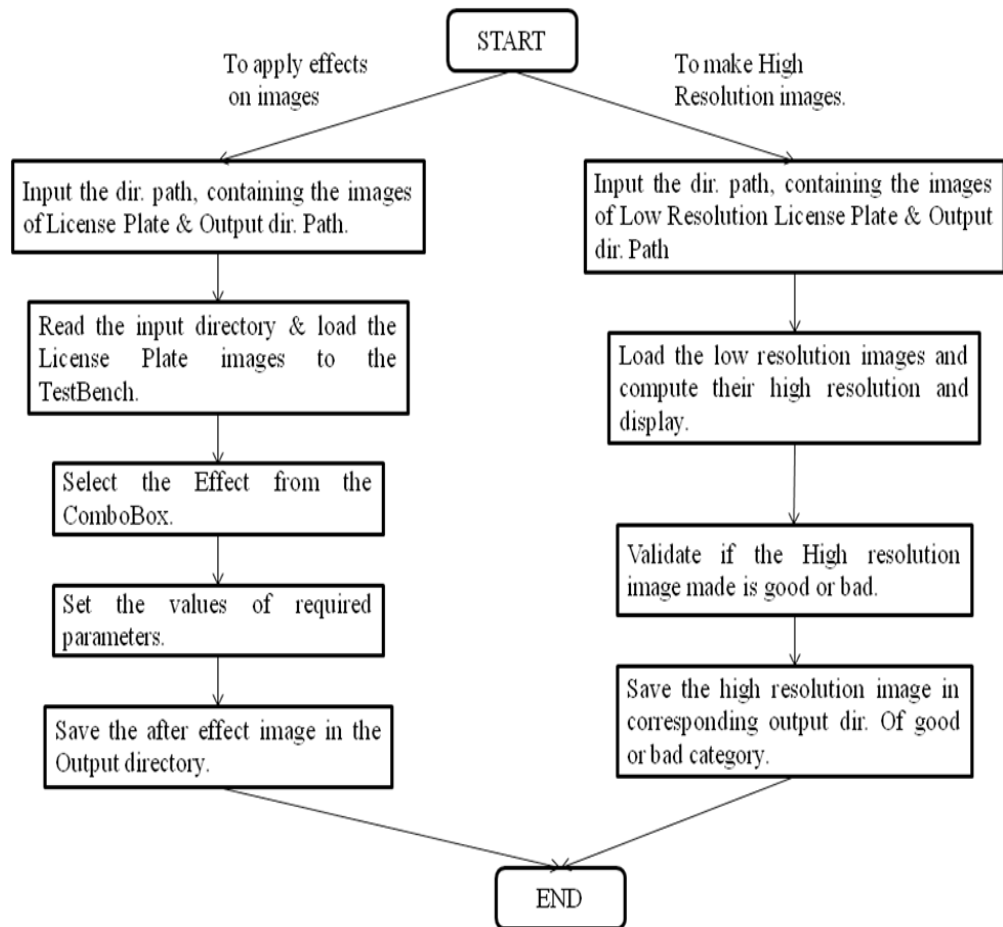


Fig 3.1 Approach for License Plate TestBench

For Vehicle Type Detection TestBench:

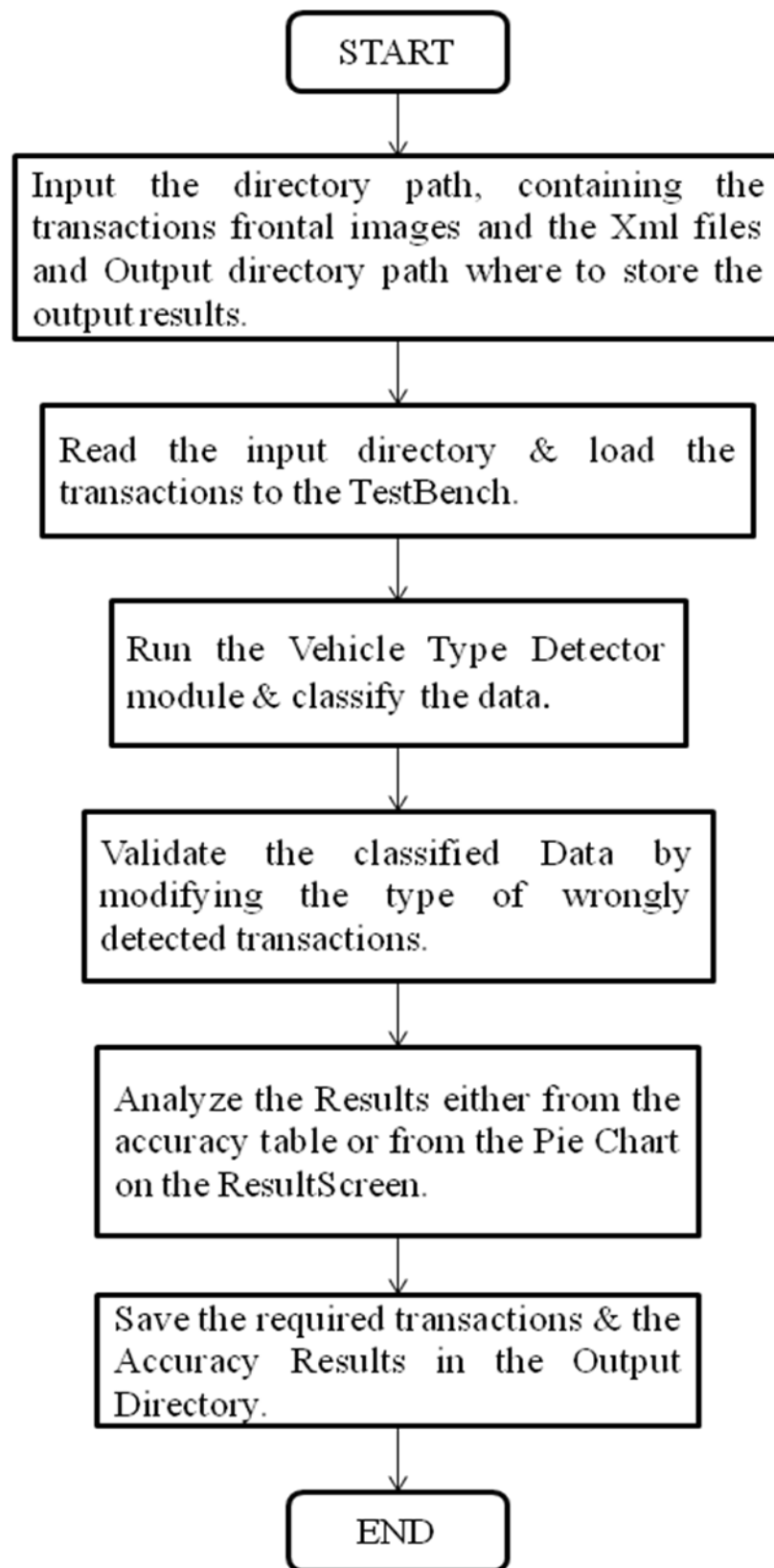


Fig 3.2 Approach for Vehicle Type Detection TestBench

## Chapter 4

# PERFORMANCE ANALYSIS

### 4.1 Performance Measures

For Vehicle Type Detection the performance measures will be Total Accuracy Percentage and the Accuracy percentage of individual type. Also False Positive for each vehicle type will be calculated, which will become one of the most important performance measures.

For License Plate Effects and Super Resolution performance measures will be percentage of after effect images which are good after the Super Resolution and the accuracy percentage of the ANPR module with these images.

### 4.2 Result Analysis

The Vehicle Type Detection TestBench has been used to find out the accuracy and do the analysis on different set of Data from different sites.

The Results achieved after running the module on data of syn camera of Site1 for time duration of 11am to 7pm are shown below:

<b>Result</b>			
<b>S.NO</b>	<b>Vehicle</b>	<b>Total</b>	<b>Accuracy (%)</b>
1	AUTOL	91	1.1
2	AUTOH	65	67.69
3	LMV	2003	86.22
4	HMV	146	83.56
5	HMVH	0	0
6	TWOWH	317	91.48
7	TWOWHH	81	3.7
8	TWOWHL	2	50

Table 4.1 Result of VTD Site1

Wrong Detected Distribution %								
S.N O	Vehicle	HM V	HMV H	AUTO L	LM V	TWOW H	TWOW HL	TWOWH H
1	AUTOL	27.5	13.19	0	52.7 5	0	5.49	0
2	AUTOH	1.54	15.38	1.54	3.08	0	7.69	3.08
3	LMV	6.69	6.94	0	0	0	0.15	0
4	HMV	0	10.96	0	5.48	0	0	0
5	TWOWH	0	5.99	0.63	0.32	0	0.63	0.95
6	TWOWH H	0	25.93	1.23	0	66.67	2.47	3.7
7	TWOWH L	0	50	0	0	0	0	0

Table 4.2 Wrong Detection of VTD Site1

The Results achieved after running the module on data of syn camera of Site2 for time duration of 11am to 4pm are shown below:

Result			
S.NO	Vehicle	Total	Accuracy (%)
1	AUTOL	44	2.5
2	AUTOH	9	22.22
3	LMV	516	91.75
4	HMV	48	93.48
5	HMVH	2	100
6	TWOWH	77	96
7	TWOWHH	15	0
8	TWOWHL	0	0

Table 4.3 Result of VTD Site2

Wrong Detected Distribution %								
S.N O	Vehicle	HM V	HMV H	AUTO H	AUTO L	LM V	TWOW H	TWOWH L
1	AUTOL	32.5	12.5	5	0	40	0	7.5
2	AUTOH	0	44.44	0	0	11.1 1	0	22.22
3	LMV	3.81	4.02	0	0	0	0	0.42
4	HMV	0	4.35	2.17	0	0	0	0
5	HMVH	0	0	0	0	0	0	0
6	TWOWH	0	4	0	0	0	0	0
7	TWOWH H	0	50	0	0	0	50	0

Table 4.4 Wrong Detection of VTD Site2

4.3 Output Screenshots

4.3.1 Vehicle Type Detection TestBench



Fig 4.1 Vehicle Type Detect Screen

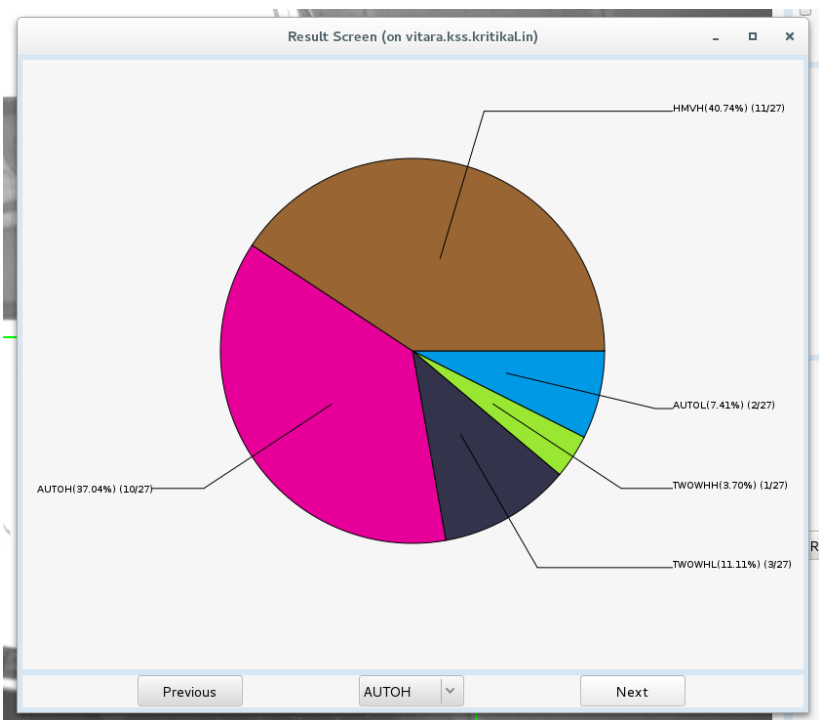


Fig 4.2 Result Screen



4.3.2 License Plate Effects TestBench



Fig 4.3 LP Effects Screen



Fig 4.4 Super Resolution Screen





















Sr. No.	Effect Used	Original Image	Parameters Value		After Effect Image
1	Random Blur		Kernel Size Block Size Percent	9 7 0.80	
2	Add Shadow		Location Strength Percent 1 Percent 2	3 0.3 0.66 0.10	
3	Random Scratch		Length Intensity	50 225	
4	Motion Blur		Length Theta	5 135	
5	Tail Light		Strength	5.50	
6	Erode Numbers		Size	2	
7	Dilate Numbers		Size	4	
8	Suppress Contrast		Min. Val. Max. Val.	18 54	
9	Join Characters		Size Percent	7 0.50	
10	Salt & Pepper Noise		Percent	0.35	

Table 4.5 Output of different Effect

## Chapter 5

# CONCLUSION

### 5.1 Conclusion

There cannot be always a perfect algorithm when it comes to open ended problems like in our case of Vehicle Type Detection and License Plate number reading or recognition because of a very large number of factors on which they depend. All we can try is to make our algorithms more and more near to perfect. The algorithm we have used in Vehicle Type detection is one of the best in this business but it still have some exceptions/cases where it is not able to find the correct type in the image. Also the best algorithm is used for ANPR but still we are having cases where we are not getting accurate results. Each algorithm yields good results in different cases. Cases need to be studied more and more efficient algorithm should be found. Recording has some limitations, clipping errors comes and recorded video is interleaved. The quality of recorded video is to be improved. This application can contribute more to improvisation of algorithm. With changing parameters that affect the performance of detection algorithm, we can list down those parameters. The videos recorded from traffic can be classified as day, night and videos having only bikes or cars or trucks. With some parameters enable the result gets improved but run time increases. Going through R&D using this application we can suggest that what parameters are to be set on a given type video condition.

### 5.2 Future Scope

- We would like to process and train the data produced by applying different effects on the LP images and find out the results of ANPR.
- Find out the areas of improvement with the classified data and try to improve the accuracy of our modules.
- Export Site transactions directly to the TestBench for analysis.

## References

- [1] "Glade and Gtk::Builder", Developer.gnome.org, 2017. [Online]. Available: <https://developer.gnome.org/gtkmm-tutorial/stable/chapter-builder.html>. [Accessed: 15- Feb- 2017].
- [2] Programming with gtkmm, 1st ed. Boston, USA: Free Software Foundation, 2002, pp. 15-166, 172-207.
- [3] M. Lades, J.C. Vorbruggen, J. Buhmann, J. Lange, C. von der Malsburg, R.P. Wurtz, and W. Konen. Distortion invariant object recognition in the dynamic link architecture. *Computers, IEEE Transactions on*, 42(3):300-311, Mar 1993.
- [4] Peter Shin, Hector Jasso, Sameer Tilak, Neil Cotofana, Tony Fountain, Linjun Yan, Mike Fraser, and Ahmed Elgamal. Automatic vehicle type classification using strain gauge sensors. *Pervasive Computing and Communications Workshops, 2007. PerCom Workshops '07. Fifth Annual IEEE International Conference on*, pages 425-428, March 2007.
- [5] T.R. Lim and A.T. Guntoro. Car recognition using gabor filter feature extraction. *Circuits and Systems, 2002. APCCAS '02. 2002 Asia-Pacific Conference on*, 2:451-455 vol.2, 2002.
- [6] "Doxygen: Main Page", Stack.nl, 2017. [Online]. Available: <http://www.stack.nl/~dimitri/doxygen/>. [Accessed: 15- Feb- 2017].
- [7] S. Gupte, O. Masoud, R. Martin and N. Papanikolopoulos, "Detection and classification of vehicles", *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 37-47, 2002.
- [8] C. Patel, D. Shah and A. Patel, "Automatic Number Plate Recognition System (ANPR): A Survey", *International Journal of Computer Applications*, vol. 69, no. 9, pp. 21-33, 2013.
- [9] A. K. Jain. *Fundamentals of Digital Image Processing*. Prentice Hall, 1989.
- [10] J. Banerjee and C. V. Jawahar, "Super-Resolution of Text Images Using Edge-Directed Tangent Field," 2008 The Eighth IAPR International Workshop on Document Analysis Systems, Nara, 2008, pp. 76-83. doi: 10.1109/DAS.2008.26.