A Real-Time Vehicle License Plate Recognition (LPR) System

A Thesis report

submitted towards the partial fulfillment of

the requirements of the degree of

Master of Engineering

In

Electronics Instrumentation and Control Engineering

Submitted By:

Mukesh Kumar

Roll No-80751015

Under the supervision of:

Dr. Yaduvir Singh

Associate Professor
EIED.

JULY 2009

DEPARTMENT OF ELECTRICAL AND INSTRUMENTATION ENGINEERING

THAPAR UNIVERSITY

PATIALA – 147004

Declaration

I hereby declare that the report entitled ***"A Real-Time Vehicle License Plate Recognition (LPR) System"*** *is an authentic record of my own work carried out as requirements for the award of degree of M.E. (Electronic Instrumentation & Control) at Thapar University, Patiala, under the guidance of Dr. Yaduvir Singh (AP, EIED) during January to June 2009.*

MUKESH KUMAR

Roll. No. 80751015

Date: 6/7/09

It is certified that the above statement made by the student is correct to the best of my knowledge and belief.

06/07/09

Dr. Yaduvir Singh
Associate Professor, EIED
Thapar University, Patiala.

S. Ghosh 07/07/2009

Dr. Smarajit Ghosh
Professor & Head EIED
Thapar University, Patiala.

Dr. R. K. Sharma
Dean of Academic Affairs
Thapar University, Patiala.

Acknowledgment

The real spirit of achieving a goal is through the way of excellence and austere discipline. I would have never succeeded in completing my task without the cooperation, encouragement and help provided to me by various personalities.

With deep sense of gratitude I express my sincere thanks to my esteemed and worthy supervisor, Dr. Yaduvir Singh, Associate Professor, Department of Electrical & Instrumentation Engineering, Thapar University, Patiala, for his valuable guidance in carrying out this work under his effective supervision, encouragement, enlightenment and cooperation.

I shall be failing in my duties if I do not express my deep sense of gratitude towards Dr.Smarajit Ghosh, Professor & Head of the Department of Electrical & Instrumentation Engineering, Thapar University, Patiala who has been a constant source of inspiration for me throughout this work.

I am also thankful to all the staff members of the Department for their full cooperation and help.

This acknowledgement would be incomplete if I do not mention the emotional support and blessings provided by my friends. I had a pleasant enjoyable and fruitful company with them

My greatest thanks are to all who wished me success especially my parents, my brother and sister. Above all I render my gratitude to the ALMIGHTY who bestowed self-confidence, ability and strength in me to complete this work.

Place: Thapar University, Patiala

Date: 6/7/09

Mukesh Kumar

Abstract

A License Plate Recognition (LPR) System is one kind of an Intelligent Transport System and is of considerable interest because of its potential applications in highway electronic toll collection and traffic monitoring systems. This type of applications puts high demands on the reliability of an LPR System. A lot of work has been done regarding LPR systems for Korean, Chinese, European and US license plates that generated many commercial products. However, little work has been done for Indian license plate recognition systems.

The purpose of this thesis was to develop a real time application which recognizes license plates from cars at a gate, for example at the entrance of a parking area or a border crossing. The system, based on regular PC with video camera, catches video frames which include a visible car license plate and processes them. Once a license plate is detected, its digits are recognized, displayed on the User Interface or checked against a database. The focus is on the design of algorithms used for extracting the license plate from a single image, isolating the characters of the plate and identifying the individual characters.

The Proposed system has been implemented using Vision Assistant 7.1 & LabVIEW 7.1 The performance of the system has been investigated on real images of about 100 vehicles. Recognition of about 98% vehicles shows that the system is quite efficient.

Organization of Thesis

Chapter 1 :- The first chapter briefly reviews the literature and the previous work done.

Chapter 2:- The second chapter gives a brief introduction to the system elements its applications, working and structure of proposed system.

Chapter 3:- The third chapter gives a detailed description of analysis and processing tools available in application software Vision Assistant 7.1, on which our work is focused.

Chapter 4:- The fourth chapter gives the problem definition and proposed solution.

Chapter 5:- The fifth chapter discusses the implementation of various tools of application software for simulation and testing part of thesis.

Finally, concluding thesis in sixth chapter with future scope.

Contents

List of Figures

Chapter 1
Literature Review

1.1 Introduction

License plate recognition systems have received a lot of attention from the research community. Much research has been done on Korean, Chinese, Dutch and English license plates. A distinctive feature of research work in this area is being restricted to a specific region, city, or country. This is due to the lack of standardization among different license plates (i.e., the dimension and the layout of the license plates). This section gives an overview of the research carried out so far in this area and the techniques employed in developing an LPR system in lieu of the following four stages: image acquisition, license plate extraction, license plate segmentation and license plate recognition phases. In the next section various existing or novel methods for the image acquisition phase are presented.

1.2 Image Acquisition

Image Acquisition is the first step in an LPR system and there are a number of ways to acquire images, the current literature discusses different image acquisition methods used by various authors. Yan et. al. [20] used an image acquisition card that converts video signals to digital images based on some hardware-based image preprocessing. Naito et. al. [13,14,16] developed a sensing system, which uses two CCDs (Charge Coupled Devices) and a prism to split an incident ray into two lights with different intensities. The main feature of this sensing system is that it covers wide illumination conditions from twilight to noon under sunshine, and this system is capable of capturing images of fast moving vehicles without blurring. Salgado et. al. [15] used a Sensor subsystem having a high resolution CCD camera supplemented with a number of new digital operation capabilities. Kim et. al. [17] uses a video camera to acquire the image. Comelli et. al. [6] used a TV camera and a frame grabber card to acquire the image for the developed vehicle LPR system.

1.3 License Plate Extraction

License plate extraction is the most important phase in an LPR system. This section discusses some of the previous work done during the extraction phase. Hontani et. al. [21] proposed a method for extracting characters without prior knowledge of their position and size in the image. The technique is based on scale shape analysis, which in turn is based on the assumption that, characters have line-type shapes locally and blob-type shapes globally. In the scale shape analysis, Gaussian filters at various scales blur the given image and larger size shapes appear at larger scales. To detect these scales the idea of principal curvature plane is introduced. By means of normalized principal curvatures, characteristic points are extracted from the scale space x-y-t. The position (x, y) indicates the position of the figure and the scale t indicates the inherent characteristic size of corresponding figures. All these characteristic points enable the extraction of the figure from the given image that has line-type shapes locally and blob-type shapes globally. Kim et. al. [17] used two Neural Network-based filters and a post processor to combine two filtered images in order to locate the license plates. The two Neural Networks used are vertical and horizontal filters, which examine small windows of vertical and horizontal cross sections of an image and decide whether each window contains a license plate. Cross-sections have sufficient information for distinguishing a plate from the background. Lee et. al. [5] and Park et. al. [11] devised a method to extract Korean license plate depending on the color of the plate. A Korean license plate is composed of two different colors, one for characters and other for background and depending on this they are divided into three categories. In this method a neural network is used for extracting color of a pixel by HLS (Hue, Lightness and Saturation) values of eight neighboring pixels and a node of maximum value is chosen as a representative color. After every pixel of input image is converted into one of the four groups, horizontal and vertical histogram of white, red and green (i.e. Korean plates contains white, red and green colors) are calculated to extract a plate region. To select a probable plate region horizontal to vertical ratio of plate is used. Dong et. al [10] presented histogram based approach for the extraction phase. Kim G. M [9] used Hough transform for the extraction of the license plate. The algorithm behind the method consists of five steps. The first step is to threshold the

gray scale source image, which leads to a binary image. Then in the second stage the resulting image is passed through two parallel sequences, in order to extract horizontal and vertical line segments respectively. The result is an image with edges highlighted. In the third step the resultant image is then used as input to the Hough transform, this produces a list of lines in the form of accumulator cells. In fourth step, the above cells are then analyzed and line segments are computed. Finally the list of horizontal and vertical line segments is combined and any rectangular regions matching the dimensions of a license plate are kept as candidate regions. The disadvantage is that, this method requires huge memory and is computationally expensive.

1.4 Segmentation

This section discusses previous work done for the segmentation of characters. Many different approaches have been proposed in the literature and some of them are as follows, Nieuwoudt et. al. [8] used region growing for segmentation of characters. The basic idea behind region growing is to identify one or more criteria that are characteristic for the desired region. After establishing the criteria, the image is searched for any pixels that fulfill the requirements. Whenever such a pixel is encountered, its neighbors are checked, and if any of the neighbors also match the criteria, both the pixels are considered as belonging to the same region. Morel et. al. [7] used partial differential equations (PDE) based technique, Neural network and fuzzy logic were adopted in for segmentation into individual characters.

1.5 Recognition

This section presents the methods that were used to classify and then recognize the individual characters. The classification is based on the extracted features. These features are then classified using either the statistical, syntactic or neural approaches. Some of the previous work in the classification and recognition of characters is as follows, Hasen et. al. [23] discusses a statistical pattern recognition approach for recognition but their technique found to be inefficient. This approach is based on the probabilistic model and uses statistical pattern recognition approach. Cowell et. al. [24] discussed the recognition of individual Arabic and Latin characters. Their approach identifies the characters based on

the number of black pixel rows and columns of the character and comparison of those values to a set of templates or signatures in the database. Cowell et. al. [22] discusses the thinning of Arabic characters to extract essential structural information of each character which may be later used for the classification stage. Mei Yu et. al. [18] and Naito et. al. [12] used template matching. Template matching involves the use of a database of characters or templates. There is a separate template for each possible input character. Recognition is achieved by comparing the current input character to each of template in order to find the one which matches the best. If I(x,y) is the input character, T„(x,y) is template n, then the matching function s(l,Tn) will return a value indicating how well template n matches the input. Hamami et. al. [25] adopted a structural or syntactic approach to recognize characters in a text document, this technique can yield a better result when applied on the recognition of individual characters. This approach is based on the detection of holes and concavities in the four directions (up, down, left and right), which permits the classification of characters into different classes. In addition, secondary characteristics are used in order to differentiate between the characters of each class. The approaches discussed in this paragraph are based on the structural information of the characters and uses syntactic pattern recognition approach. Hu [1] proposed seven moment that can be used as features to classify the characters. These moments are invariant to scaling, rotation and translation. The obtained moments acts as the features, which are passed to the neural network for the classification or recognition of characters. Zemike moments have also been used by several authors [4,2,3] for recognition of characters. Using zernike moments both the rotation variant and rotation invariant features can be extracted. These features then uses neural network for the recognition phase. Neural network accepts any set of distinguishable features of a pattern as input. It then trains the network using the input data and the training algorithms to recognize the input pattern (In this case characters).

1.6 Commercial Products

The various products in the market today are described briefly below.

1.6.1 IMPS (Integrated Multi-Pass System)

An IMP [26] is a Singaporean commercially developed license plate recognition system. It is a high performing robust system that gives consistent results under all weather conditions. Using advanced image processing and artificial intelligent techniques such as AI best first breadth-wise search algorithm, combined template and neural network recognizers, fuzzy logic and an arsenal of image processing tools, it automatically locates vehicle license plates and reads the numbers accurately each time every time.

1.6.2 Perceptics

Perceptics [27] is the world leader in license plate reader technology. Current LPR system read Latin (A-Z) and Korean (Hangul) letter and Arabic number (0-9); however, the LPR can be programmed to read any language or symbol in any alphanumeric combination or context on both retro and non-retro reflective plates. With milliseconds the LPR system locates, captures and identifies a vehicle's license plate data and makes a read decision. The system's reliability and flexibility allow it to accommodate some of the most stringent needs in some of the worst conditions. Features of this LPR technology includes.

- Automatic and within milliseconds.
- Reads accurately in most weather conditions.
- Reads accurately at highway speeds.
- Works 24 hours a day, 7 days a week.

1.6.3 Vehicle Identification System for Parking Areas (VISPA)

PPI's Vehicle Identification System for Parking Areas (VISPA) [28], uses video imaging for better recognition, identification and improved security. VISPA provides for state-of-the-art video technology, easy installation and has accessories and features for most parking security surveillance needs.

Features are

- Open architecture to most common video-systems.
- Compatible with standard hardware and software.
- Can be customized according to specific user needs.

VISPA is available in two forms

Basic Version: - An image of the car and/or the driver (depending on the location of your camera) will be taken as soon as the car approaches the triggering device. The image will be linked to the ticket. The basic system version connects to 4 cameras and can be upgraded to 8 cameras.

Enhanced Version:- License Plate Identification, The VISPA controller with an integrated frame grabber card for 4, 8, or 16 cameras automatically identifies the license plate from the video image and stores it in a database. The license plate can then be encoded on the ticket.

1.6.4 Hi-Tech Solution

Hi-Tech Solutions [29] is a system and software company that develops cutting edge optical character recognition (OCR) solutions by implementing the company's unique image processing software and hardware in a wide range of security and transportation applications. There technology is based on computer vision, the system read the camera images and extract the identification data from the images. The recognition result is then logged together with the images. This is the main advantage of vision based recognition, the records include both the image plus the extracted result. There product includes,

SeeCar License Plate Recognition:- Detects and reads Vehicle license plates for parking, access control, traffic surveillance, law enforcement and security applications. Available as a complete system which is based on a background Windows application, Windows DLL or Linux library, as a stand-alone turn-key version, or in form of different special-task systems.

SeeContainer Identification System:- Tracks and reads Shipping container identification marking, and transmits the ID string to the port or gate computer, or to a client process. Available as complete systems, such as SeeGate - a recognition system for the Tracks and Containers, or SeeCrane - crane mounted Container recognition system.

1.7 Summary

This chapter reviewed material relevant to the license plate recognition system. The relevant techniques used in the four phases of an LPR system were discussed. Several commercially available and developed LPR systems is also presented. In the case of image acquisition, a sensing system using two Charge Coupled Devices along with a prism gives better input to the system. Because the main feature of this sensing system is that it covers wide illumination conditions from twilight to noon under sunshine, and this system is capable of capturing images of fast moving vehicles without blurring video camera with a frame. In the case of license plate extraction, Hough transform was used to extract the license plate by using storing the horizontal and vertical edge information. But the disadvantage is that, this method requires huge memory and is computationally expensive. Various segmentation techniques were presented in the segmentation stage. Then the literature for recognition of characters using various approaches was also discussed. Lastly, some of the number plate recognition systems which have been developed commercially were presented.

Chapter 2
Introduction

2.1 Introduction

License plate recognition (LPR) is an image-processing technology used to identify vehicles by their license plates. This technology is gaining popularity in security and traffic installations. Much research has already been done for the recognition of Korean, Chinese, European, American and other license plates, This thesis presents a license plate recognition system as an application of computer vision. Computer vision is a process of using a computer to extract high level information from a digital image. This chapter will set the scene by first presenting some applications of a license plate recognition system. Next, we discuss the elements that are commonly used in a license plate recognition system. Following this, the working of a typical LPR system is described. Next, we present the structure of proposed license plate recognition system. Finally, the objectives of the work are stated. The chapter ends with a brief overview of the rest of this thesis.

2.2 Applications of LPR Systems

Vehicle license plate recognition is one form of automatic vehicle identification system. LPR systems are of considerable interest, because of their potential applications to areas such as highway electronic toll collection, automatic parking attendant, petrol station forecourt surveillance, speed limit enforcement, security, customer identification enabling personalized services, etc. Real time LPR plays a major role in automatic monitoring of traffic rules and maintaining law enforcement on public roads. This area is challenging because it requires an integration of many computer vision problem solvers, which include Object Detection and Character Recognition. The automatic identification of vehicles by the contents of their license plates is important in private transport applications. There are many applications of such recognition systems, some of them are discussed below.

Law Enforcement :- The plate number is used to produce a violation fine on speeding vehicles, illegal use of bus lanes, and detection of stolen or wanted vehicles. License

plate recognition technology has gained popularity in security and traffic applications as it is based on the fact that all vehicles have a license plate and there is no need to install any additional tracking apparatus. The main advantage is that the system can store the image record for future references. The rear part of the vehicle is extracted off the filmed image and is given to the system for processing. The processed result is fed into the database as input. The violators can pay the fine online and can be presented with the image of the car as a proof along with the speeding information.

Parking :- The LPR system is used to automatically enter pre-paid members and calculate parking fee for non-members (by comparing the exit and entry times). The car plate is recognized and stored and upon its exit the car plate is read again and the driver is charged for the duration of parking.

Automatic Toll Gates :- Manual toll gates require the vehicle to stop and the driver to pay an appropriate tariff. In an automatic system the vehicle would no longer need to stop. As it passes the toll gate, it would be automatically classified in order to calculate the correct tariff.

Border Crossing :- This application assists the registry of entry or exits to a country, and can be used to monitor the border crossings. Each vehicle information is registered into a central database and can be linked to additional information.

Homeland Security :- The LPR system's ability to read strings of alpha-numeric characters and compare them instantaneously to Hot Lists allows a Command Center to organize and strategize efforts in reaction to the information captured. Fixed LPR systems, which can be mounted to bridges, gates and other high traffic areas can help keep a tight watch on entire cities, ports, borders and other vulnerable areas. Every LPR camera is capturing critical data such as color photos, date and time stamps, as well as GPS coordinates on every vehicle that passes or is passed. This incredible database provides a wealth of clues and proof, which can greatly aid Law Enforcement with

- Pattern recognition
- Placing a suspect at a scene
- Watch list development
- Identifying witnesses

- Possible visual clues revealed within the image of a car's immediate environment

2.3 Elements of Typical LPR System

LPR systems normally consist of the following units:

Camera :- Takes image of a vehicle from either front or rear end.

Illumination :- A controlled light that can bright up the plate, and allows day and night operation. In most cases the illumination is Infra-Red (IR) which is invisible to the driver.

Frame Grabber :- An interface board between the camera and the PC that allows the software to read the image information.

Computer :- Normally a PC running Windows or Linux. It runs the LPR application that controls the system, reads the images, analyzes and identifies the plate, and interfaces with other applications and systems.

Software :- The application and the recognition package.

Hardware :- Various input/output boards used to interface the external world (such as control boards and networking boards).

Database :- The events are recorded on a local database or transmitted over the network. The data includes the recognition results and (optionally) the vehicle or driver-face image file.

2.4 Working of Typical LPR System

When the vehicle approaches the secured area, the LPR unit senses the car and activates the illumination (invisible infra-red in most cases) as shown in Figure below. The LPR unit takes the pictures from either the front or rear plates from the LPR camera. The image of the vehicle contains the license plate. The LPR unit feeds the input image to the system. The system then enhances the image, detects the plate position, extracts the plate, segments the characters on the plate and recognizes the segmented characters, Checks if the vehicle appears on a predefined list of authorized vehicles, If found, it signals to open the gate by activating its relay. The unit can also switch on a green "go-ahead" light or red "stop" light. The unit can also display a welcome message or a message with personalized data. The authorized vehicle enters into the secured area. After passing the gate its detector closes the gate. Now the system waits for the next vehicle to approach the secured area.

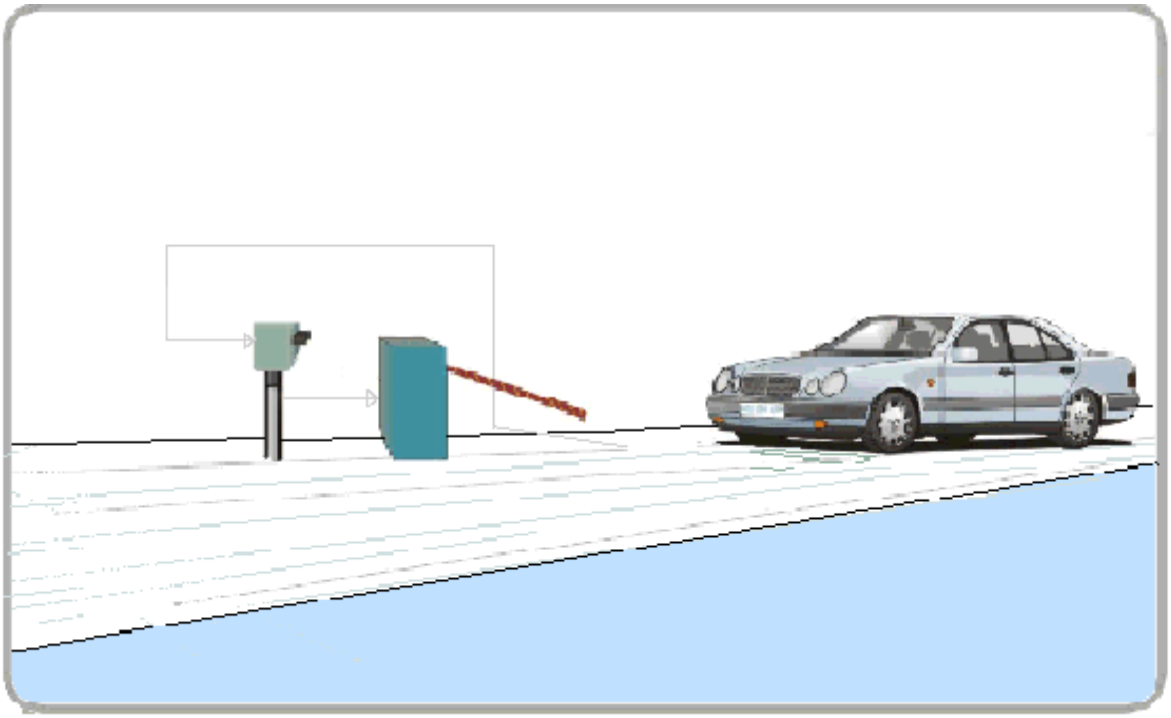


Figure 2.1 A car approaching a license plate recognition system

2.5 Structure of the Proposed System

The system presented is designed to recognize license plates from the front and rear of the vehicle. Input to the system is an image sequence acquired by a digital camera that consists of a license plate and its output is the recognition of characters on the license plate. The system consists of the standard four main modules in an LPR system, viz. Image acquisition, License plate extraction, License plate segmentation and License plate recognition. The first task acquires the image. The second task extracts the region that contains the license plate. The third task isolates the characters, letters and numerals (total of 10 digits), as in the case of Indian License Plates. The last task identifies or recognizes the segmented characters.

2.5.1 Image Acquisition

This is the first phase in an LPR system. This phase deals with acquiring an image by an acquisition method. In our proposed system, we used a high resolution digital camera to acquire the input image. The input image is 1200 x 1600 pixels.

2.5.2 License Plate Extraction

License Plate Extraction is a key step in an LPR system, which influences the accuracy of the system significantly. This phase extracts the region of interest, i.e., the license plate, from the acquired image. The proposed approach involves “Masking of a region with high probability of license plate and then scanning the whole masked region for license plate”.

2.5.3 License Plate Segmentation

License Plate Segmentation, which is sometimes referred to as Character Isolation takes the region of interest and attempts to divide it into individual characters. In the proposed system segmentation is done in the OCR section which will be described in next chapters.

2.5.4 License Plate Recognition

The last phase in LPR system is to recognize the isolated characters. After splitting the extracted license plate into individual character images, the character in each image can be identified. There are many methods used to recognize isolated characters. In the proposed system we are using Optical Character Recognition which is an inbuilt feature in Vision Assistant 7.1 . Optical Character Recognition is described in detail in next chapters.

2.6 Objective

The work presented here aims at the following aspects.

- Study the existing license plate recognition systems,
- Develop a new technique or enhance existing techniques for each phase in a license plate recognition system,
- Compare the various techniques at hand with the proposed system, and
- Build a system that delivers optimal performance both in terms of speed and accuracy.

Chapter 3
Software Development

3.1 Digital Images

This section contains information about the properties of digital images, image types, file formats, the internal representation of images in IMAQ Vision, image borders, and image masks.

3.1.1 Definition of a Digital Image

An image is a 2D array of values representing light intensity. For the purposes of image processing, the term image refers to a digital image. An image is a function of the light intensity.

$$F(x,y)$$

where f is the brightness of the point (x, y), and x and y represent the spatial coordinates of a picture element, or pixel. By convention, the spatial reference of the pixel with the coordinates (0, 0) is located at the top, left corner of the image. Notice in Figure 3.1 that the value of x increases moving from left to right, and the value of y increases from top to bottom.

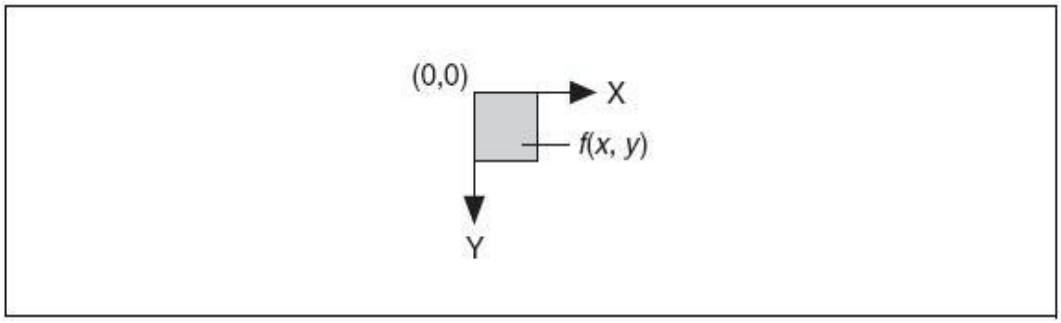


Fig 3.1 Special reference of the (0,0) pixel.

3.2 Vision Assistant: An overview

A detailed overview of vision assistant 7.1 is given as under.

3.2.1 Acquiring Images

Vision Assistant offers three types of image acquisitions: snap, grab, and sequence. A snap acquires and displays a single image. A grab acquires and displays a continuous set of images, which is useful while focusing the camera. A sequence acquires images according to settings that are specified and sends the images to the Image Browser. Using Vision Assistant, images can be acquired with various National Instruments digital and analog IMAQ devices. Vision Assistant provides specific support for several Sony,

JAI, and IEEE 1394 cameras. IMAQ devices can be configured in National Instruments Measurement & Automation Explorer (MAX). The sequence can be stopped at any frame, capture the image, and send the image to the Image Browser for processing.

(A) Opening the Acquisition window

Complete the following steps to acquire images.

1. Click **Start » Programs » National Instruments » Vision Assistant 7.1**.

2. Click **Acquire Image** in the Welcome screen to view the Acquisition functions.

If Vision Assistant is already running, click the **Acquire Image** button in the toolbar. We must have one of the following device and driver software combinations to acquire live images in Vision Assistant.

- National Instruments IMAQ device and NI-IMAQ 3.0 or later
- IEEE 1394 industrial camera and NI-IMAQ for IEEE 1394 Cameras 1.5 or later

Click **Acquire Image**. The Parameter window displays the IMAQ devices and channels installed on the computer.

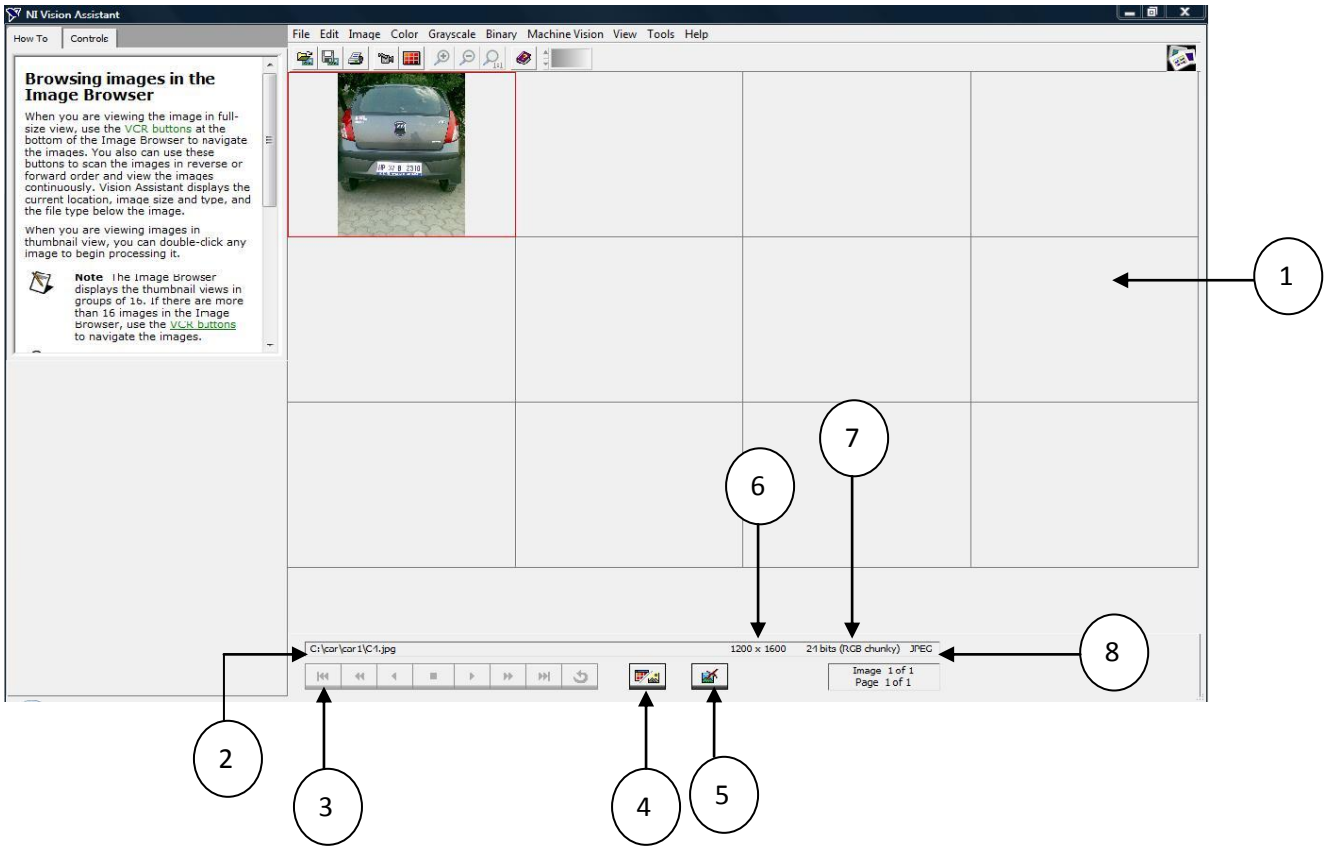
(B) Snapping an image

1. Click **File » Acquire** Image.
2. Click **Acquire Image** in the Acquisition function list.
3. Select the appropriate device and channel.
4. Click the **Acquire Single Image** button to acquire a single image with the IMAQ device and display it. 
5. Click the **Store Acquired Image in Browser** button to send the image to the Image Browser. 
6. Click **Close** to exit the Parameter window.
7. Process the image in Vision Assistant.

3.2.2 Managing Images

1. Select **Start » Programs » National Instruments » Vision Assistant 7.1**

2. To load images, click Open **Image** in the Welcome screen.



1. Image Browser **4.** Thumbnail/ Full-Size Toggle **7.** Image Type

2. Image Location **5.** Close Selected Image(s) **8.** File Format

3. Browse Buttons **6.** Image Size

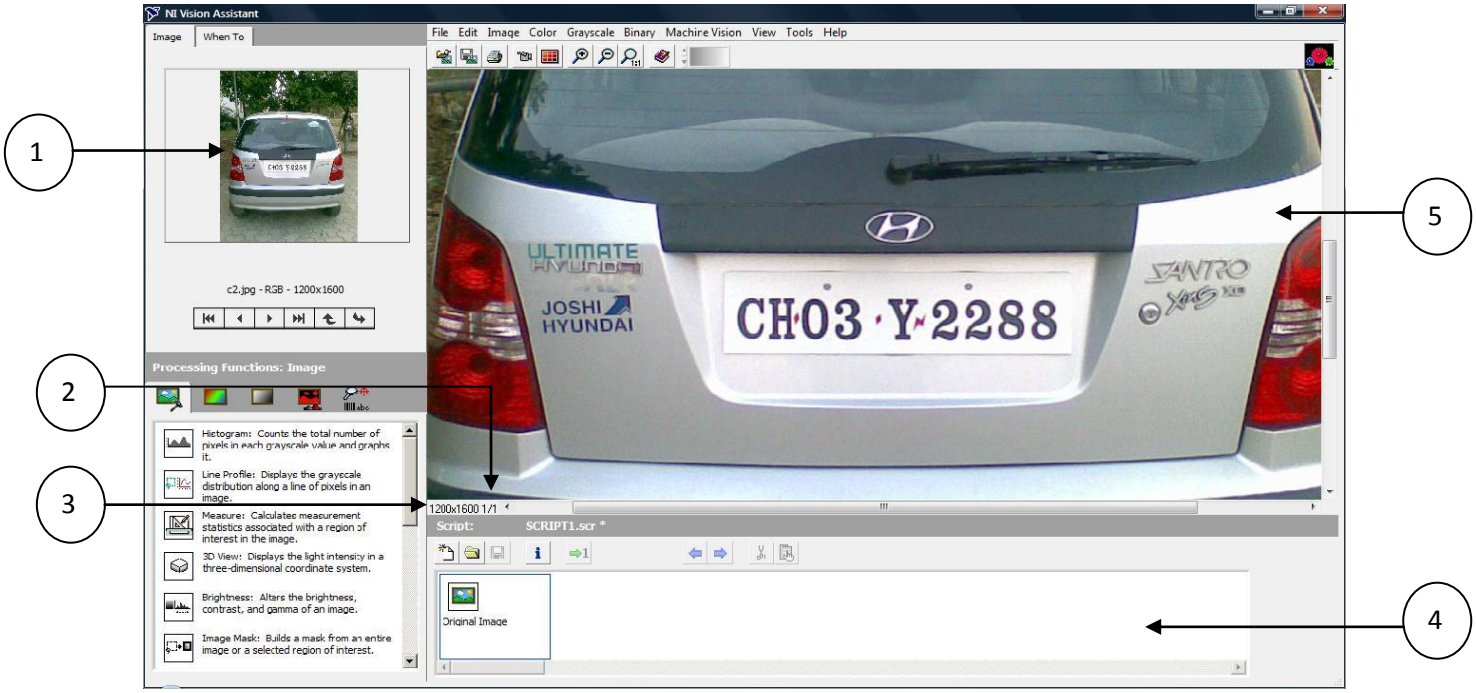
Fig: 3.2 Image Browser

3. Navigate to select the image which we want to process. If analysis is to be done on more than one image then there is Select All Files option also available in Vision Assistant. It previews the images in the Preview Image window and displays information about the file type.

4. Click OK. Vision Assistant loads the image files into the Image Browser, as shown in Figure 5.2. The Image Browser provides information about the selected image such as

image size, location, and type. We can view new images in either thumbnail view, as shown in Figure 3.2 or in full-size view, which shows a single full-size view of the selected image.

5. Click the **Thumbnail/Full-Size View Toggle** button to view the first image in full size.



1.Reference Window **2.** Zoom Ratio **3.** Image Size **4.** Script Window **5.** Processing Window

Fig: 3.3 Processing an Image

6. Double-click on one of the selected images to begin processing it. Vision Assistant loads the image into the Processing window, as shown in Figure 3.3.

7. Now, image is ready for processing as per requirements. We can apply various functions on image that are available in processing function window as shown in figure given above. These processing steps get recorded in the Script window. The script records the processing operations and all its parameters. If we want to run the same operation on other images, we can save the script and use it again.

8. Select **File » Save** Script and name the script.

9. To run script on other images follow the steps given below:

(i) Load the image. (ii)Select **File » Open Script**

(iii) Click the **Run Once** button in the script window

10. Select **File » Exit** to close Vision Assistant.

3.2.3 Image Processing Functions

The various functions available in Vision Assistant that can be used for image processing and analysis are listed below. The following gives an overview of available functions in Vision Assistant 7.1.

(i) Image analysis functions.

Vision Assistant provides the following image analysis functions:-

- ***Histogram*** counts the total number of pixels in each grayscale value and graphs the result.
- ***Line Profile*** returns the grayscale values of the pixels along a line that is drawn with the Line Tool from the Tools palette and graphs the result
- ***Measure*** calculates measurement statistics associated with a region of interest in the image.
- ***3D View*** displays an image using an isometric view.
- ***Image Mask*** builds a mask from an entire image or region of interest.
- ***Geometry*** modifies the geometrical representation of an image.
- ***Image Buffer*** stores and retrieves images from buffers.
- ***Get Image*** opens a new image from a file.

(ii) Colour image processing functions.

Vision Assistant provides the following set of functions for processing and analyzing colour images:

- ***Color Operators*** applies an arithmetic operation between two images or between an image and a constant .
- ***Extract Color Planes*** extracts the Red, Green, or Blue (RGB) plane or the Hue, Saturation, or Luminance (HSL) plane of a color image.
- ***Color Threshold*** applies a threshold to the three planes of an RGB or HSL image.
- ***Color Location*** locates colors in an image.
- ***Color Matching*** compares the color content of one or multiple regions in an image to a reference color set.
- ***Color Pattern Matching*** searches for a color template in an image.

(iii) Grayscale image processing and analysis functions.

Vision Assistant provides the following functions for grayscale image processing and analysis:

- **Lookup Table** applies predefined lookup table transformations to the image to modify the dynamic intensity of regions in the image with poor contrast.
- **Filters** include functions for smoothing, edge detection, and convolution.
- **Gray Morphology** modifies the shape of objects in grayscale images using erosion, dilation, opening, and closing functions.
- **FFT Filters** applies a frequency filter to the image.
- **Threshold** isolates the pixels we specify and sets the remaining pixels as background pixels.
- **Operators** perform basic arithmetic and logical operations between two images or between an image and a constant.
- **Conversion** converts the current image to the specified image type.
- **Centroid** computes the energy center of a grayscale image or an area of interest on a grayscale image.

(iv) Binary processing and analysis functions.

Vision Assistant provides the following functions for binary processing and analysis:

- **Basic Morphology** performs morphology transformations that modify the shape of objects in binary images.
- **Adv. Morphology** performs high-level operations on particles in binary images.
- ***Particle Filter*** filters objects based on shape measurements.
- ***Invert Binary Image*** reverses the dynamic of an image that contains two different grayscale populations.
- ***Shape Matching*** finds image objects that are shaped like the object specified by the template.
- ***Particle Analysis*** computes more than 80 measurements on objects in an image, including the area and perimeter of the objects.

- ***Circle Detection*** separates overlapping circular objects and classifies them according to their radii.

(v) Machine vision functions.

Vision Assistant provides the following machine vision functions:

- **Edge Detection** finds edges along a line that we draw with the Line Tool.
- **Find Straight Edge** finds points along the edge of an object and then finds a line describing the edge.
- **Find Circular Edge** locates the intersection points between a set of search lines within a circular area, or annulus, and then finds the best fit circle.
- **Clamp** finds edges within a rectangular region of interest (ROI) drawn in the image and measures the distance between the first and last edge.
- **Pattern Matching** locates regions of a grayscale image that match a predetermined template. Pattern Matching can find template matches regardless of poor lighting, blur, noise, shifting of the template, and rotation of the template.
- **Caliper** computes measurements such as distances, areas, and angles—based on results returned from other machine vision and image processing functions.

3.3 Script Development

Once the desired image is loaded in the Processing window as shown in Figure 5.3, we follow the steps discussed below to develop the script.

3.3.1 Extracting color planes from image

Since the color information is redundant so we extract color plane from the acquired 32-bit colored image to make it an 8-bit grayscale image.

1. Click **Color » Extract Color Plane** or select **Extract Color Plane** in the **Color** tab of the Processing Functions Palette.
2. Select the **color plane** to be extracted.
3. Click **OK** to add this step to the script.

The choice of color plane to be extracted is made by trying all of them one by one and selecting the one which gives the best image thereafter. We have selected to extract

green plane from the image. Figure 5.4 shows how the image looks after applying this function.

3.3.2 Brightness, Contrast, Gamma adjustment

This function is used to alter the brightness, contrast, and gamma of an image.

- **Brightness -** Brightness of the image. Brightness is expressed in gray levels, centered at 128. Higher values (up to 255) result in a brighter image, and lower values (down to 0) result in a darker image.
- **Contrast -** Contrast of the image, expressed as an angle. A value of 45° specifies no change in contrast. Higher values (up to 89°) increase contrast, and lower value (down to 1°) decrease contrast.
- **Gamma -** Gamma value used in Gamma Correction. The higher the Gamma coefficient, the stronger the intensity correction.

Reset—Reverts **Brightness**, **Contrast**, and **Gamma** to neutral default settings.

1. Click **Image » Brightness** or select **Brightness** in the **Image** tab of the Processing Functions palette.
2. Modify the **Brightness, Contrast** and **Gamma** values with the slide or by manually entering the value.

Click **OK** to add the step to the script

During our project we have applied the following values to these functions as shown in Figure 3.4 below

Brightness: 150

Contrast: 68.00

Gamma: 1.14

For different images these three parameters are first checked with varying values and then the optimum values are taken. The values above are the optimum values which work satisfactory for most lightening conditions.

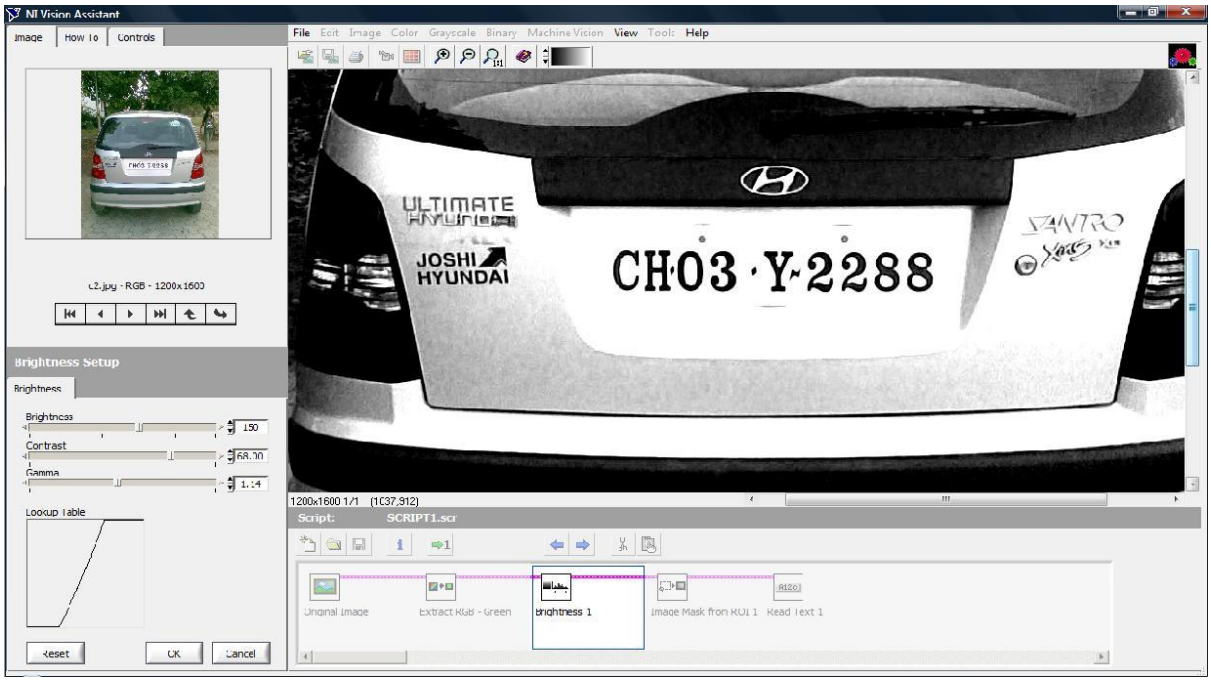


Figure: 3.4 Brightness Adjustments

3.3.3 Image Mask

An image mask isolates parts of an image for processing. If a function has an image mask parameter, the function process or analysis depends on both the source image and the image mask. An image mask is an 8-bit binary image that is the same size as or smaller than the inspection image. Pixels in the image mask determine whether corresponding pixels in the inspection image are processed. If a pixel in the image mask has a nonzero value, the corresponding pixel in the inspection image is processed. If a pixel in the image mask has a value of 0, the corresponding pixel in the inspection image is not processed. Pixels in the source image are processed if corresponding pixels in the image mask have values other than zero. A mask affects the output of the function that inverts the pixel values in an image.

A B C D

Fig 3.5 Effect of an image mask.

Figure 3.5A shows the inspection image. Figure 3.5B shows the image mask. Pixels in the mask with zero values are represented in black, and pixels with nonzero values are represented in white. Figure 3.5C shows the inverse of the inspection image using the image mask. Figure 3.5D shows the inverse of the inspection image without the image mask. We can limit the area in which our function applies an image mask to the bounding rectangle of the region we want to process. This technique saves memory by limiting the image mask to only the part of the image containing significant information. To keep track of the location of this region of interest (ROI) in regard to the original image, IMAQ Vision sets an offset. An offset defines the coordinate position in the original image where you want to place the origin of the image mask. Figure 3.6 illustrates the different methods of applying image masks. Figure 3.6a shows the ROI in which you want to apply an image mask.

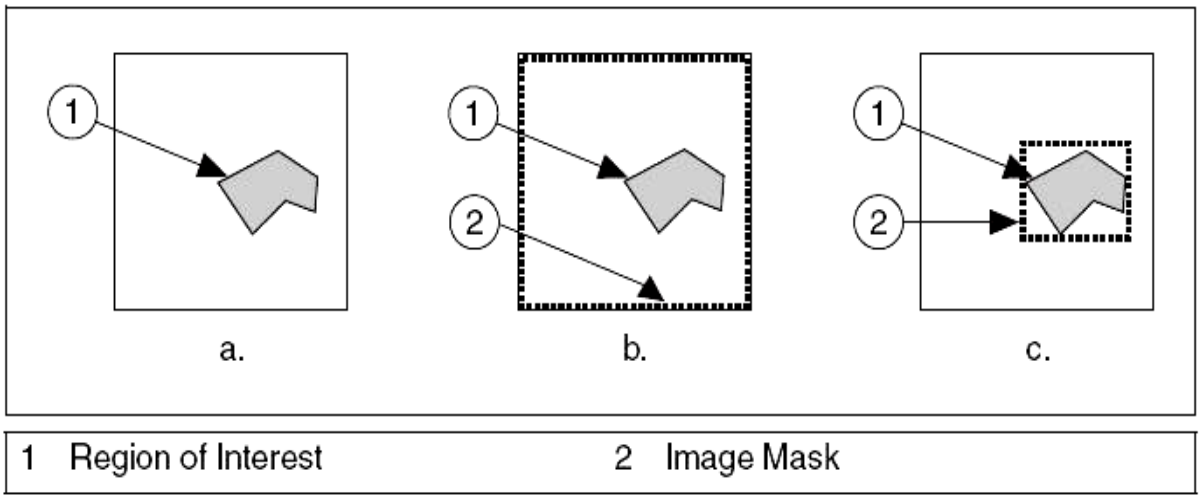


Fig 3.6 Using an Offset to limit an image mask.

Figure 3.6b shows an image mask with the same size as the inspection image. In this case, the offset is set to [0, 0]. A mask image also can be the size of the bounding rectangle of the ROI, as shown in Figure 3.6c, where the offset specifies the location of the mask image in the reference image. We can define this offset to apply the mask image to different regions in the inspection image.

3.4 Optical Character Recognition (OCR)

The exact mechanisms that allow humans to recognize objects are yet to be understood, but the three basic principles are already well known by scientists – integrity, purposefulness and adaptability. These principles constitute the core of OCR allowing it to replicate natural or human-like recognition. Optical Character Recognition provides machine vision functions we can use in an application to perform OCR. OCR is the process by which the machine vision software reads text and/or characters in an image.

3.4.1 What is OCR

Optical Character Recognition (OCR) is a type of document image analysis where a scanned digital image that contains either machine printed or handwritten script is input into an OCR software engine and translating it into an editable machine readable digital text format (like ASCII text). OCR works by first pre-processing the digital page image into its smallest component parts with layout analysis to find text blocks, sentence/line blocks, word blocks and character blocks. Other features such as lines, graphics, photographs etc are recognised and discarded. The character blocks are then further broken down into components parts, pattern recognized and compared to the OCR engines large dictionary of characters from various fonts and languages. Once a likely match is made then this is recorded and a set of characters in the word block are recognized until all likely characters have been found for the word block. The word is then compared to the OCR engine's large dictionary of complete words that exist for that language. These factors of characters and words recognised are the key to OCR accuracy by combining them the OCR engine can deliver much higher levels of accuracy. Modern

OCR engines extend this accuracy through more sophisticated pre-processing of source digital images and better algorithms for fuzzy matching, sounds-like matching and grammatical measurements to more accurately establish word accuracy.

3.4.2 When to Use

Typically, machine vision OCR is used in automated inspection applications to identify or classify components. For example, we can use OCR to detect and analyze the serial number on an automobile engine that is moving along a production line. Using OCR in this instance helps you identify the part quickly, which in turn helps you quickly select the appropriate inspection process for the part. We can use OCR in a wide variety of other machine vision applications, such as the following

- Inspecting pill bottle labels and lot codes in pharmaceutical applications.
- Verifying wafers and IC package codes in semiconductor applications.
- Controlling the quality of stamped machine parts.
- Sorting and tracking mail packages and parcels.
- Reading alphanumeric characters on automotive parts.
- **And of course for license plate recognition applications**.

3.4.3 Training Characters

Training involves teaching OCR the characters and/or patterns you want to detect during the reading procedure. Figure 3.7 illustrates the steps involved in the training procedure.

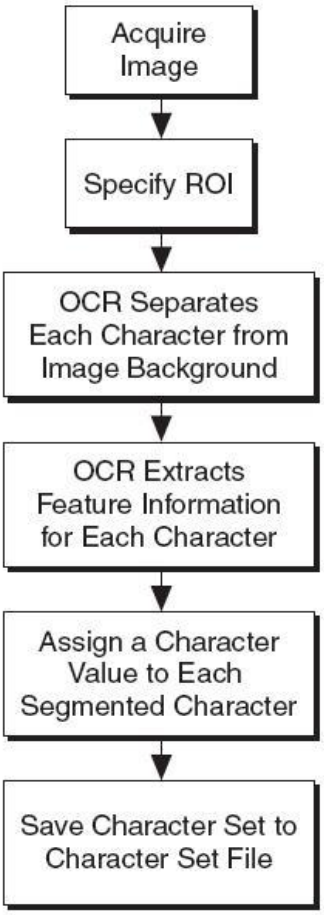


Fig 3.7 Steps of an OCR training procedure

The process of locating characters in an image is often referred to as character segmentation. Before we can train characters, we must set up OCR to determine the criteria that segment the characters you want to train. When we finish segmenting the characters, we'll use OCR to train the characters, storing information that enables OCR to recognize the same characters in other images. We train the OCR software by providing a character value for each of the segmented characters, creating a unique representation of each segmented character. You then save the character set to a character set file to use later in an OCR reading procedure.

3.4.4 Reading Characters

When we perform the reading procedure, the machine vision application we created with OCR functions segments each object in the image and compares it to characters in the character set you created during the training procedure. OCR extracts unique features from each segmented object in the image and compares each object to each

character stored in the character set. OCR returns the objects that best match characters in the character set as the recognized characters.

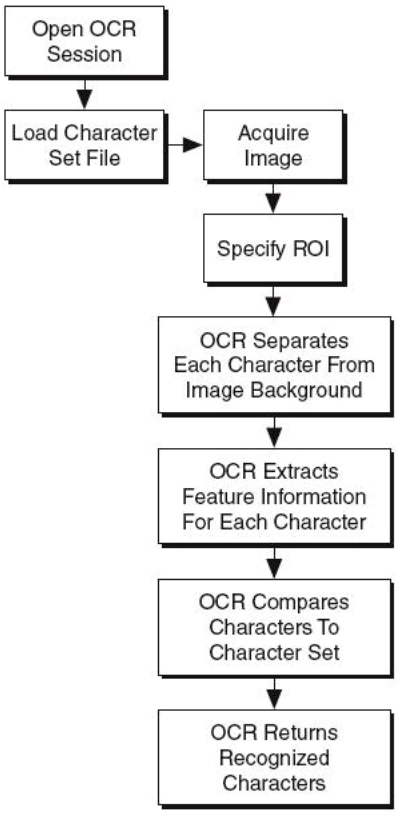


Fig 3.8 Steps of an OCR reading procedure

3.4.5 OCR Session

An OCR session applies to both the training and reading procedures. An OCR session prepares the software to identify a set of characters during either the training procedure or the reading procedure. A session consists of the properties we set and the character set that we train or read from a file. OCR uses session information to compare objects with trained characters to determine if they match. If we want to process an image containing characters that we stored in multiple character sets, use multiple OCR sessions simultaneously to read all the characters simultaneously. We also can merge several character sets in one session. If we choose to merge multiple character sets, train each of the character sets with the same segmentation parameters.

3.4.6 Region of Interest (ROI)

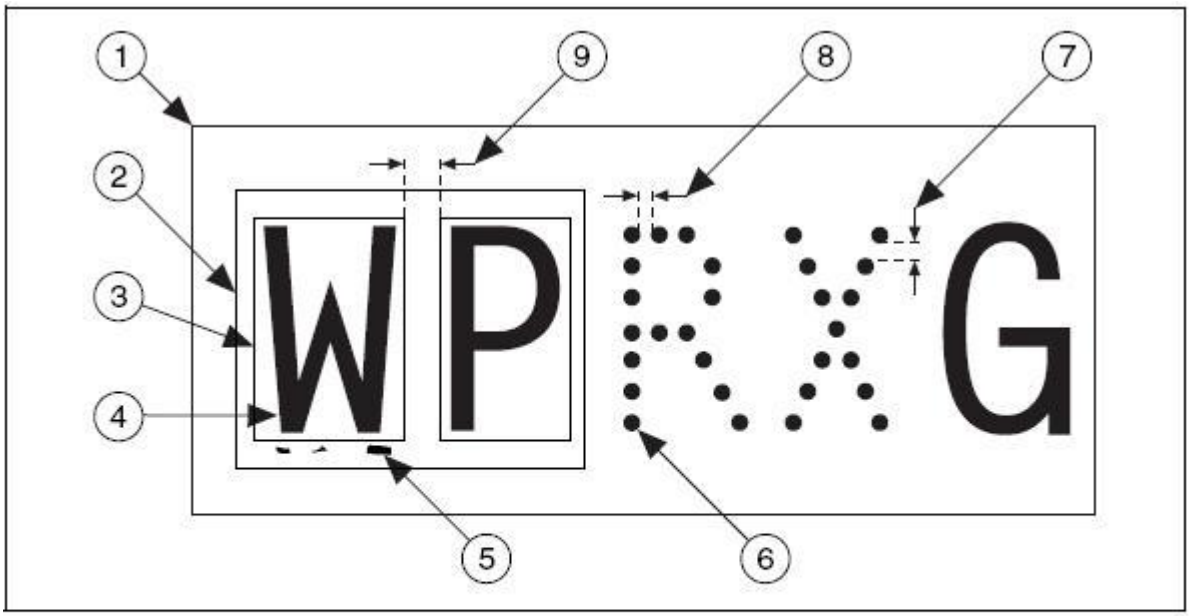
The ROI applies to both the training and reading procedures. During training, the ROI is the region that contains the objects we want to train. During reading, the ROI is the region that contains the objects we want to read by comparing the objects to the character set. We can use the ROI to effectively increase the accuracy and efficiency of OCR. During training, we can use the ROI to carefully specify the region in the image that contains the objects we want to train while excluding artifacts. During reading, we can use the ROI to enclose only the objects we want to read, which reduces processing time by limiting the area OCR must analyze.

3.4.7 Particles, Elements, Objects, and Characters

Particles, elements, objects, and characters apply to both the training and reading procedures. Particles are groups of connected pixels. Elements are particles that are part of an object. For example, the dots in a dot-matrix object are elements. A group of one or more elements forms an object based on the element spacing criteria. A character is a trained object.

3.4.8 Character Segmentation

Character segmentation applies to both the training and reading procedures. Character segmentation refers to the process of locating and separating each character in the image from the background.



1	Acquired Image	6	Element
2	ROI	7	Vertical Element Spacing
3	Character Bounding Rectangle	8	Horizontal Element Spacing
4	Character	9	Character Spacing
5	Artifact		

Fig 3.9 Concepts involved in Character segmentation

3.4.9 Thresholding

Thresholding is one of the most important concepts in the segmentation process. Thresholding is separating image pixels into foreground and background pixels based on their intensity values. Foreground pixels are those whose intensity values are within the lower and upper threshold values of the threshold range. Background pixels are those whose intensity values lie outside the lower and upper threshold values of the threshold range. OCR includes one manual method and three automatic methods of calculating the thresholding range:

Fixed Range is a method by which you manually set the threshold value. This method processes grayscale images quickly, but requires that lighting remain uniform across the ROI and constant from image to image.The following three automatic thresholding methods are affected by the pixel intensity of the objects in the ROI. If the objects are dark on a light background, the automatic methods calculate the high threshold value and set the low threshold value to the lower value of the threshold limits. If the objects

are light on a dark background, the automatic methods calculate the low threshold value and set the high threshold value to the upper value of the threshold limits.

- **Uniform** is a method by which OCR calculates a single threshold value and uses that value to extract pixels from items across the entire ROI. This method is fast and is the best option when lighting remains uniform across the ROI.
- **Linear** is a method that divides the ROI into blocks, calculates different threshold values for the blocks on the left and right side of an ROI, and linearly interpolates values for the blocks in between. This method is useful when one side of the ROI is brighter than the other and the light intensity changes uniformly across the ROI.
- **Non linear** is a method that divides the ROI into blocks, calculates a threshold value for each block, and uses the resulting value to extract pixel data.

OCR includes a method by which you can improve performance during automatic thresholding, which includes the Uniform, Linear, and Non linear methods.

- **Optimize for Speed** allows us to determine if accuracy or speed takes precedence in the threshold calculation algorithm. If speed takes precedence, enable Optimize for Speed to perform the thresholding calculation more quickly, but less accurately. If accuracy takes precedence, disable Optimize for Speed to perform the thresholding calculation more slowly, but more accurately

If we enable Optimize for Speed, we also can enable Bi modal calculation to configure OCR to calculate both the lower and upper threshold levels for images that are dominated by two pixel intensity levels.

3.4.10 Threshold Limits

Threshold limits are bounds on the value of the threshold calculated by the automatic threshold calculation algorithms. For example, if the threshold limits are 10 and 240, OCR uses only intensities between 10 and 240 as the threshold value. Use the threshold limits to prevent the OCR automatic threshold algorithms from returning too low or too high values for the threshold in a noisy image or an image that contains a low population of dark or light pixels. The default range is 0 to 255.

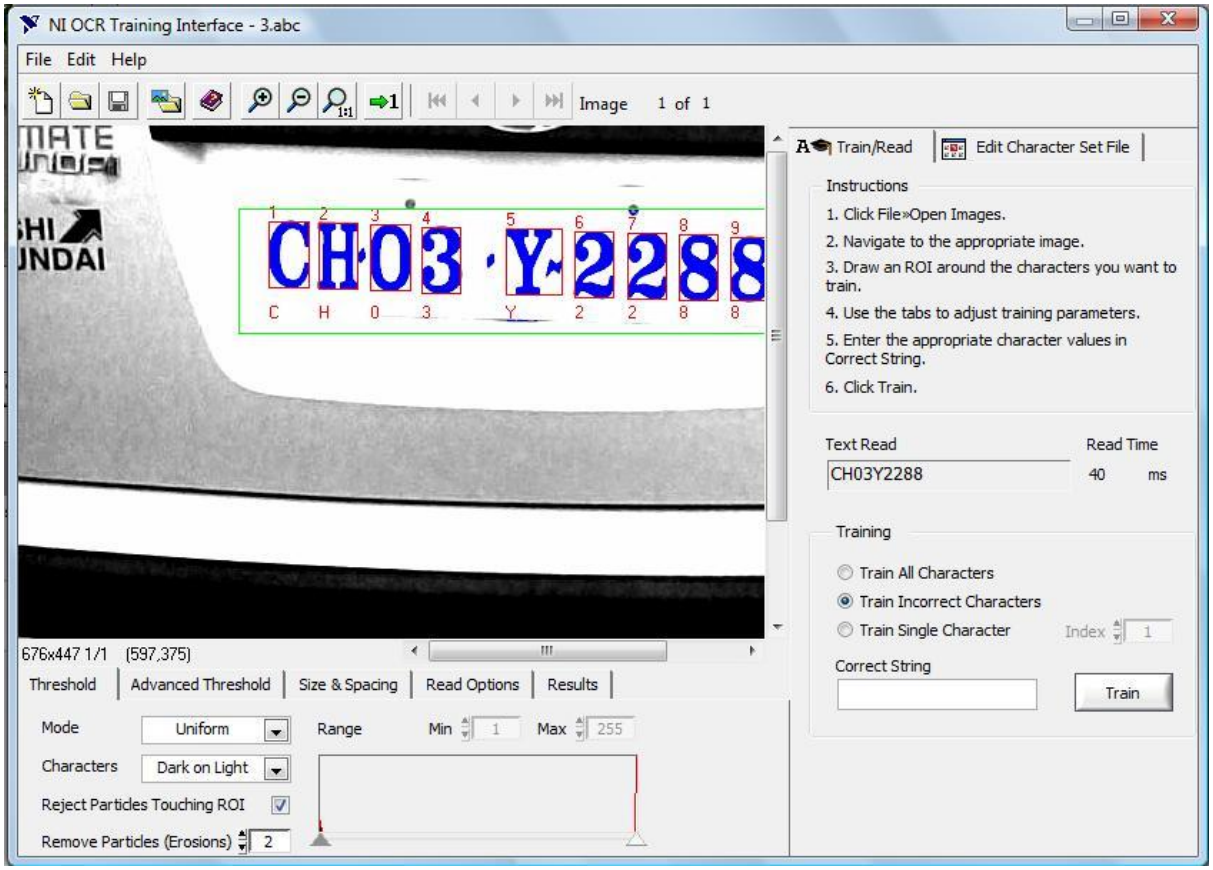


Fig 3.10 An OCR Training Interface

3.4.11 Character Spacing

Character spacing is the horizontal distance, in pixels, between the right edge of one character bounding rectangle and the left edge of the next character bounding rectangle. If an image consists of segmented or dot-matrix characters and the spacing between two characters is less than the spacing between the elements of a character, we must use individual ROIs around each character.

3.4.12 Element Spacing

Element spacing consists of horizontal element spacing and vertical element spacing. Horizontal element spacing is the space between two horizontally adjacent elements. Set this value to 1 or 2 for stroke characters and 4 or 5 for dot-matrix or segmented characters. Dot-matrix or segmented characters are characters comprised of a series of small elements. Stroke characters are continuous characters in which breaks are due only to imperfections in the image. If we set the horizontal element spacing too low, we might accidentally eliminate elements of an object. If we set the horizontal element spacing too high, we might include extraneous elements in the object, resulting in a

trained object that does not represent a matchable character. Vertical element spacing is the space between two vertically adjacent elements. Use the default value, 0, to consider all elements within the vertical direction of the ROI to be part of an object. If we set vertical element spacing too high, we might include artifacts as part of an object.

3.4.13 Character Bounding Rectangle

The character bounding rectangle is the smallest rectangle that completely encloses a character.

(a) Auto Split

Auto Split applies to both the training and reading procedures. Use AutoSplit when an image contains characters that are slanted. AutoSplit, which works in conjunction with the maximum character bounding rectangle width, uses an algorithm to analyze the right side of a character bounding rectangle and determine the rightmost vertical line in the object that contains the fewest number of pixels. AutoSplit moves the rightmost edge of the character bounding rectangle to that location. The default value is False.

(b) Character Size

Character size is the total number of pixels in a character. Generally, character size should be between 25 and 40 pixels. If characters are too small, training becomes difficult because of the limited data. The additional data included with large characters is not helpful in the OCR process, and the large characters can cause the reading process to become very slow.

(c)Substitution Character

Substitution character applies to the reading procedure only. OCR uses the substitution character for unrecognized characters. The substitution character is a question mark (?) by default.

(d) Acceptance Level

Acceptance level applies to the reading procedure. Acceptance level is a value that indicates how closely a read character must match a trained character to be recognized. The valid range for this value is 0 to 1000. The higher the value, the more closely you expect an object to match a trained character. The default value is 700.

3.4.14 Read Strategy

Read strategy applies only to the reading procedure. Read strategy refers to the criteria OCR uses to determine if a character matches a trained character in the character set. The possible modes are Aggressive and Conservative. In Aggressive mode, the reading procedure uses fewer criteria than Conservative mode to determine if an object matches a trained character. Aggressive mode works well for most applications. In Conservative mode, the reading procedure uses extensive criteria to determine if an object matches a trained character.

3.4.15 Read Resolution

Read resolution applies to the reading procedure. When we save a character set, OCR saves a variety of information about each character in the character set. Read resolution is the level of character detail OCR uses to determine if an object matches a trained character. By default, OCR uses a low read resolution, using few details to determine if there is a match between an object and a trained character. The low read resolution enables OCR to perform the reading procedure more quickly. We can configure OCR to use a medium or high read resolution, and therefore use more details to determine if an object matches a trained character. Using a high read resolution reduces the speed at which OCR processes. The low resolution works well with most applications, but some Applications might require the higher level of detail available in medium or high resolutions.

3.4.16 Valid Characters

Valid characters apply only to the reading procedure. Valid characters refer to the practice of limiting the characters that the reading procedure uses when analyzing an image. For example, if you know that the first character in an ROI should be a number, you can limit the reading procedure to comparing the first character in the ROI only to numbers in the character set. Limiting the characters that the reading procedure uses when analyzing an image increases the speed and accuracy of OCR.

3.4.16 Removing Small Particles

Removing small particles applies to both the training and reading procedures. The process of removing small particles involves applying a user-specified number of 3x3 erosions to the thresholded image. OCR fully restores any objects that remain after applying the erosions. For example, in Figure 3.11 if any portion of the letters X and G remains after removing small particles, OCR fully restores the X and G.

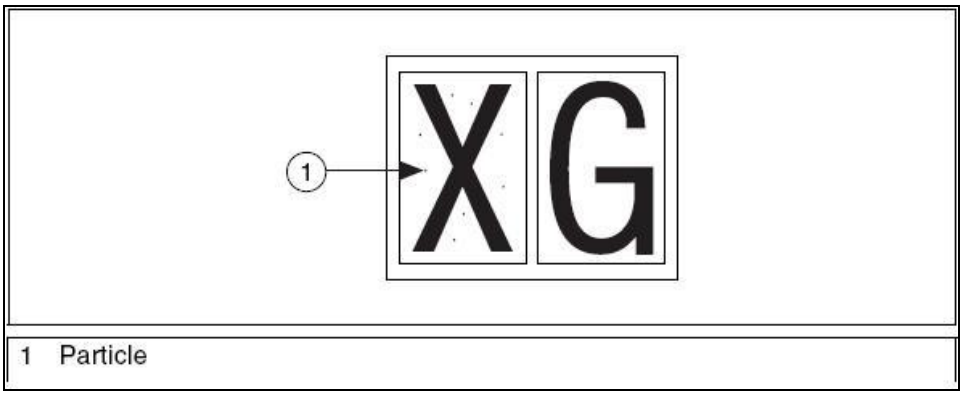


Fig 3.11 Unwanted Particles in an image

3.4.17 Removing Particles That Touch the ROI

Removing particles that touch the ROI applies to both the training and reading procedures.

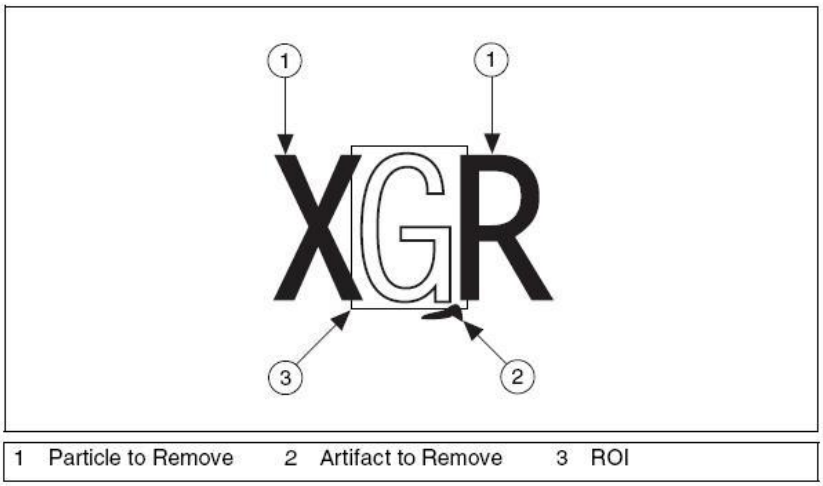


Fig 3.12 Unwanted Particles that touch ROI

We can configure OCR to remove small particles that touch an ROI we specified. Figure 3.12 shows particles that touch the ROI and these particles can be neglected by checking the remove particles touching ROI check box in an OCR training interface.

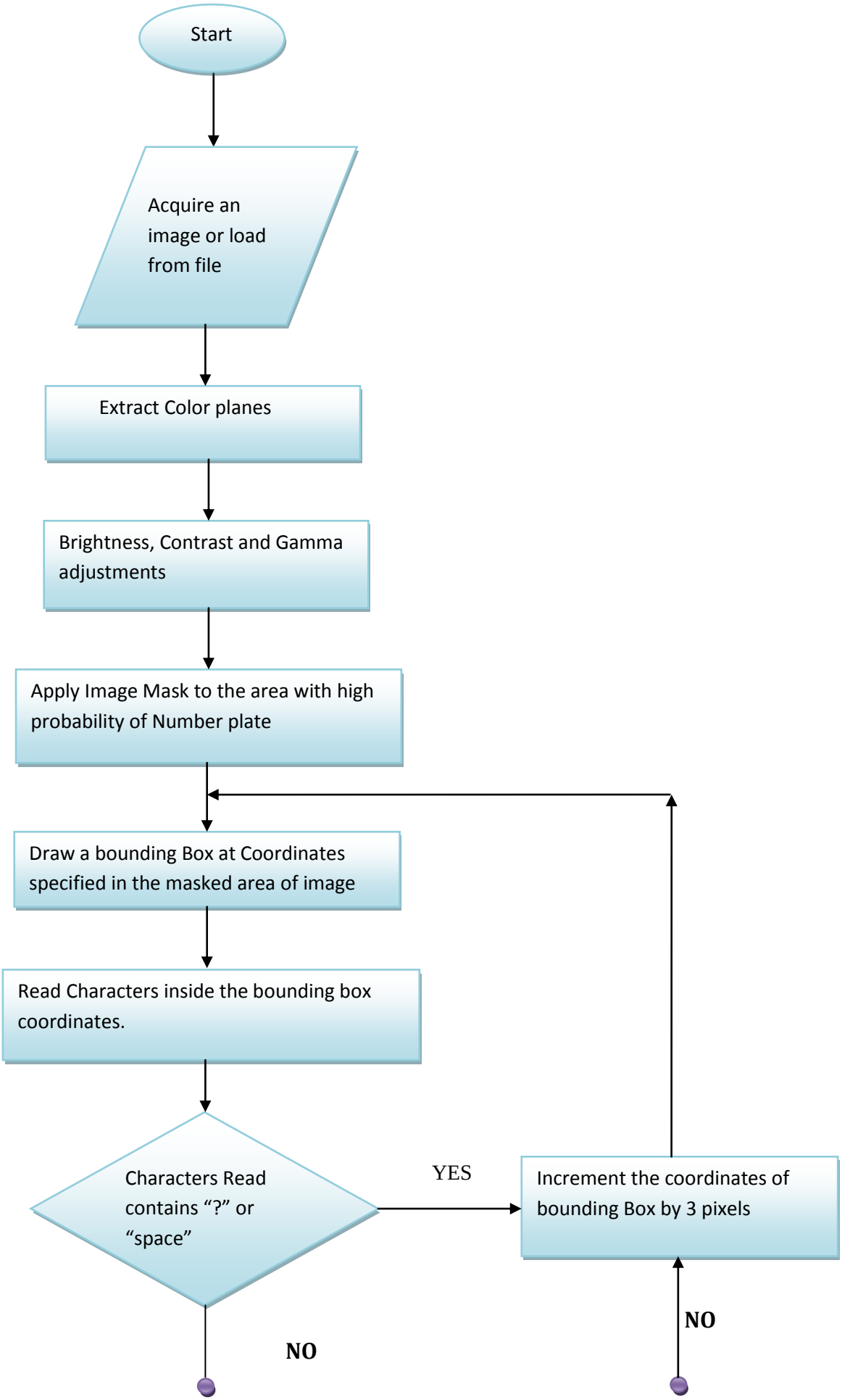
Chapter 4
Problem Formulation & Proposed Solution

4.1 Problem Definition

Traffic problems are significant in a developing or developed country. Massive integration of information technologies into all aspects of modern life caused demand for processing vehicles as conceptual resources in information systems. Because a standalone information system without any data has no sense, there was also a need to transform information about vehicles between the reality and information systems. This can be achieved by a human agent, or by special intelligent equipment which is able to recognize vehicles by their number plates in a real environment and reflect it into conceptual resources. Because of this, various recognition techniques have been developed and number plate recognition systems are today used in various traffic and security applications, such as parking, access and border control, or tracking of stolen cars.
Till now, all the LPR systems have been developed using neural networks and MATLAB. This work proposes to implement the system using LabVIEW and Vision Assistant to make the system faster and more efficient.

4.2 Proposed Solution

For real time application, the system requires a video camera (frame grabber) which acquires the images of vehicles from rear or front view. But for the present work, due to unavailability of the required hardware, we have used 2.0 Megapixels camera inbuilt in Nokia 5130 mobile.
The images of various parked vehicles have been acquired manually, thereafter fed to the software where they are first converted in to grayscale images. Brightness, contrast and gamma adjustments are made to optimum values to enhance the number plate, the and its digits. Then the region with highest probability of number plate is masked and extracted. Then the resulting region of interest is scanned for characters and numerals by continuously changing the coordinates of bounding box in an OCR session. The output of OCR is saved in a spreadsheet and then for each iteration the result is checked if it qualifies to contain all the digits in number plate. Whenever the results meet the conditions specified, the software displays the number and terminates the execution of program so that next image can be processed.
The sequence of the steps followed to develop the system is shown in Figure 4.1 below.


Start
Acquire an image or load from file
Extract Color planes
Brightness, Contrast and Gamma adjustments
Apply Image Mask to the area with high probability of Number plate
Draw a bounding Box at Coordinates specified in the masked area of image
Read Characters inside the bounding box coordinates.
Characters Read contains "?" or "space"
YES
Increment the coordinates of bounding Box by 3 pixels
NO
NO

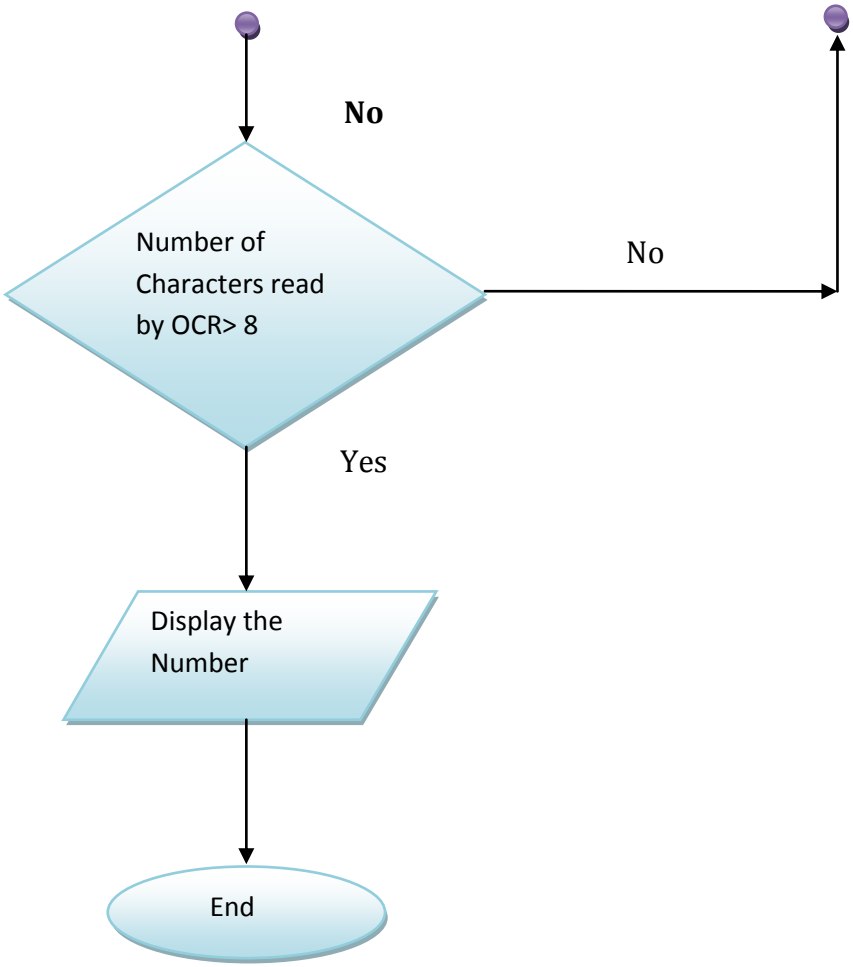


Fig 4.1 Flow Chart of the Proposed System

Chapter 5
Simulation and Testing

5.1 Introduction to LabVIEW

LabVIEW or Laboratory Virtual Instrument Engineering Workbench is graphical programming language software used for data acquisition and instrument control. The programs that take a long time using conventional programming languages can be completed in hours using LabVIEW. The LabVIEW programs are called virtual instruments (*VIs*) because their appearance and operation imitate actual instruments. A VI has two main parts

Front panel – It is the interactive user interface of a VI. The front panel can contain knobs, push buttons etc. which are the interactive input controls and graphs, LED's etc. which are indicators. Controls simulate instrument input devices and supply data to block diagram of the VI. Indicators simulate output devices and display the data that block diagram generates. The front panel objects have corresponding terminal on the block diagram

Block diagram – It is the VI's source code, constructed in LabVIEW's graphical programming language, G. The block diagram is the actual executable program. The components of the block diagram are lower level VIs, built in functions, constants and program execution control structures. Wires are used to connect the objects together to indicate the flow of data between them. The data that we enter into the front panel controls enter the block diagram through these terminals and after execution the output data flow to indicator terminals where they exit the block diagram, re-enter the front panel and appear in front panel indicators giving us final results. Various steps involved in Vision Assistant to create LabVIEW VI are:

1. Click **Tools Create LabVIEW VI.**
2. Browse path for creating a new file in VI.
3. Click **Next.**

4. If we want to create LabVIEW of current script click on **Current script** option or else click on **Script file** option and give the path for the file in the field given for browsing. Click **Next**.
5. Select image source (Image File).
6. Click **Finish**.

5.2 Code Development

Once the script is converted in to LabVIEW code it will look like the image shown in figure 5.1 below. The LabVIEW code generated by our script will be performing the following operations on an image.

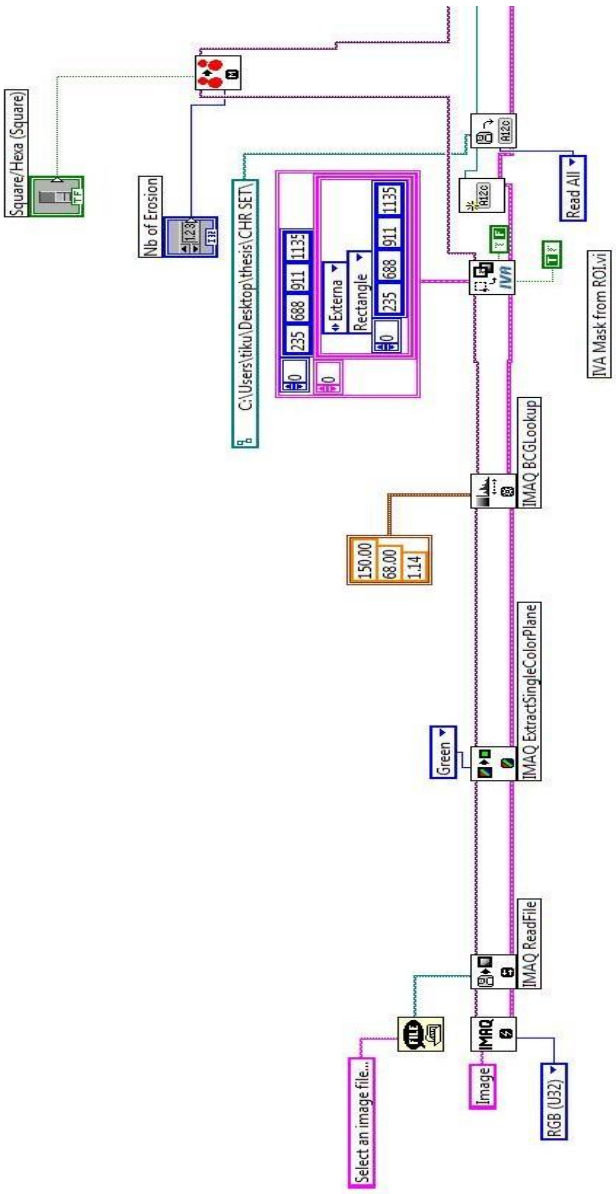


Fig 5.1 A part of LabVIEW block diagram for image acquisition and filtering

5.2.1 Extracting colour planes from image

Grayscale images are the images in which the value of each pixel is a single sample, that is, it carries only intensity information. Such images are composed exclusively of gray shade varying from black at the weakest intensity to white at the strongest. A color image is encoded in memory as a red, green, and blue (RGB) image or a hue, saturation, and luminance (HSL) image. Color image pixels are a composite of four values. RGB images store color information using 8 bits each for the red, green and blue planes. HSL images store color information using 8 bits each for hue, saturation and luminance. The camera can provide a color or monochrome signal. The choice of which to choose depends on the type of image to be processed; often grayscale intensity information is adequate to extract all necessary information from an image, but in some cases the color information is critical. There are applications which sort identically shaped objects by color and others where the grayscale contrast is too low to accurately identify objects. Since the color information is redundant in our application so we extract color plane from the acquired 32-bit colored image to make it an 8-bit grayscale image.

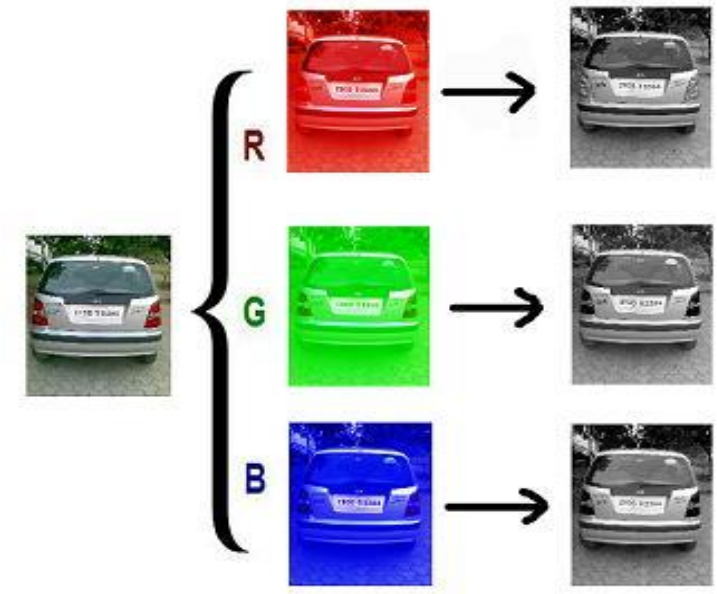


Fig 5.2 Extraction of RGB planes from an image

To convert any color to a grayscale representation of its luminance, first one must obtain the values of its red, green, and blue (RGB) primaries in linear intensity encoding, by gamma expansion. Then, add together 30% of the red value, 59% of the green value, and 11% of the blue value (these weights depend on the exact choice of the RGB primaries,

but are typical). Regardless of the scale employed (0.0 to 1.0, 0 to 255, 0% to 100%, etc.), the resultant number is the desired linear luminance value; it typically needs to be gamma compressed to get back to a conventional grayscale representation. Color images are often built of several stacked color channels, each of them representing value levels of the given channel. For example, RGB images are composed of three independent channels for red, green and blue primary color components; CMYK images have four channels for cyan, magenta, yellow and black ink plates, etc. The choice of color plane to be extracted is made by trying all of them one by one and selecting the one which gives the best image thereafter. **In the proposed system we have extracted the green plane.**

5.2.2 Brightness, Contrast, Gamma adjustment

The images are taken at different times in the day and in different light conditions therefore to neutralize the effect of lightening and other possible reflections in the image, we have used Brightness, Contrast, and Gamma adjustments so that the number plate can be enhanced before being fed to OCR session for better results. We use LUT transformations to improve the contrast and brightness of an image by modifying the dynamic intensity of regions with poor contrast. A LUT transformation converts input gray-level values in the transformed image. It applies the transform T(x) over a specified input range [rangemin, rangemax] in the following manner.

$$T(x) = \begin{cases} \textbf{dynamicMin} \text{ if } x \leq \text{rangeMin} \\ \textbf{f(x)} \text{ if rangeMin} < x \leq \text{rangeMax} \\ \textbf{dynamicMax} \text{ if } x > \text{rangeMax} \end{cases}$$

where,

x represents the input gray-level value;

dynamicMin = 0 (8-bit images) or the smallest initial pixel value (16-bit and floating point images);

dynamicMax = 255 (8-bit images) or the largest initial pixel value (16-bit and floating point images);

dynamicRange = dynamicMax – dynamicMin

f(x) represents the new value.

The function scales $f(x)$ so that f(rangeMin) = dynamicMin and f(rangeMax) = dynamicMax. $f(x)$ behaves on [rangeMin, rangeMax] according to the method we select.

The following example uses the source image shown in Figure 5.3. And it is seen that in the linear histogram of the source image, the gray-level intervals [0, 49] and [191, 254] do not contain significant information.

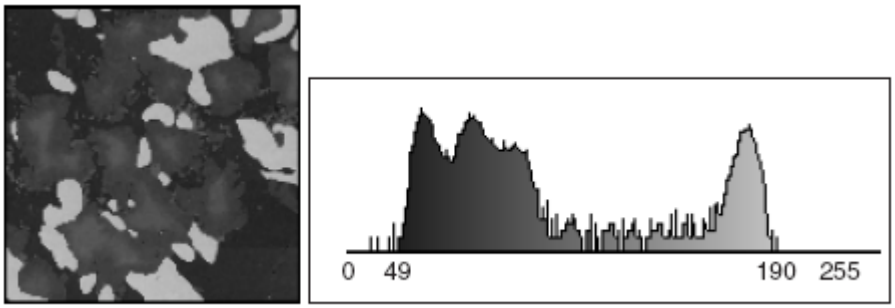


Fig 5.3 Source Image along with its histogram

Using the linear LUT transformation as shown in Figure 5.4, any pixel with a value less than 49 is set to 0, and any pixel with a value greater than 191 is set to 255. The interval [50, 190] expands to [1, 254], increasing the intensity dynamic of the regions with a concentration of pixels in the grey-level range [50, 190].

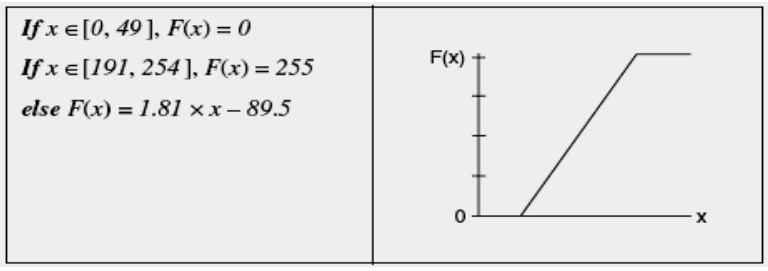


Fig 5.4 Linear LUT

The LUT transformation produces image shown in Figure 5.5. The linear histogram of the new image contains only the two peaks of the interval [50, 190].

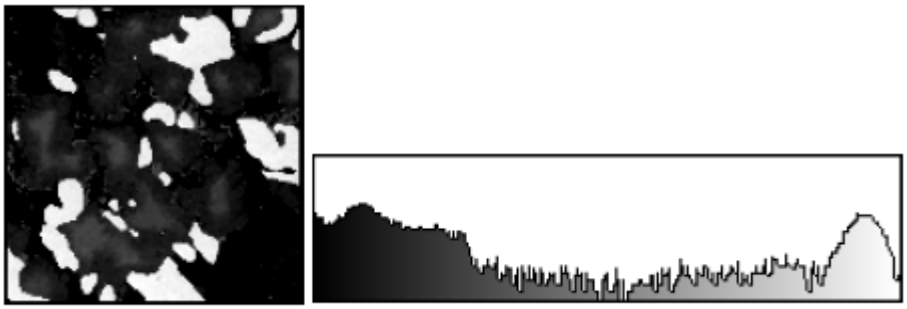


Fig5.5 Image After applying LUT

5.2.3 Image Masking

Now when the color planes have been extracted and the Brightness, Contrast and Gamma adjustments has been made the next step is to mask the image. An image mask isolates parts of an image for processing. If a function has an image mask parameter. An image mask is an 8-bit binary image that is the same size as or smaller than the inspection image. Pixels in the image mask determine whether corresponding pixels in the inspection image are processed. If a pixel in the image mask has a nonzero value, the corresponding pixel in the inspection image is processed.

(a) **(b)**

Fig 5.6 (a) Image before Masking (b) Image after Masking

If a pixel in the image mask has a value of 0, the corresponding pixel in the inspection image is not processed. Image masking is already described in previous chapter in detail. In Image masking we specify four coordinates for (X_1,Y_1) and (X_2,Y_2) and the function extracts the part of image which falls inside these coordinates. We have selected the coordinates (235, 688, 911, 1135) in such a way so that the number plate is always in the masked region as shown in the figure 5.6 (b). The resolution of image before masking was 1200x1600 pixels, after masking the new image resolution is 676x447.

5.2.4 Number Detection in the Region of Interest

The image has passed the primary steps of filtering and masking and we have a masked region which contains the number plate. The OCR session specifically goes to specific coordinates and checks for numerals or alphabets. But the number plate in the masked region could be anywhere. The task is to find it and to make the OCR read all the characters in it. For this we have used the following technique.

Fig 5.7 Bounding box specified by ROI descriptor, searching for number plate

In figure 5.7 the green box on the upper side of image is called bounding box and the OCR session have a ROI description system which specifies the coordinates where the OCR's bounding box is supposed to look for numerals and alphabets. These

coordinates are fixed and therefore for every picture supplied, the OCR goes only to those coordinates.

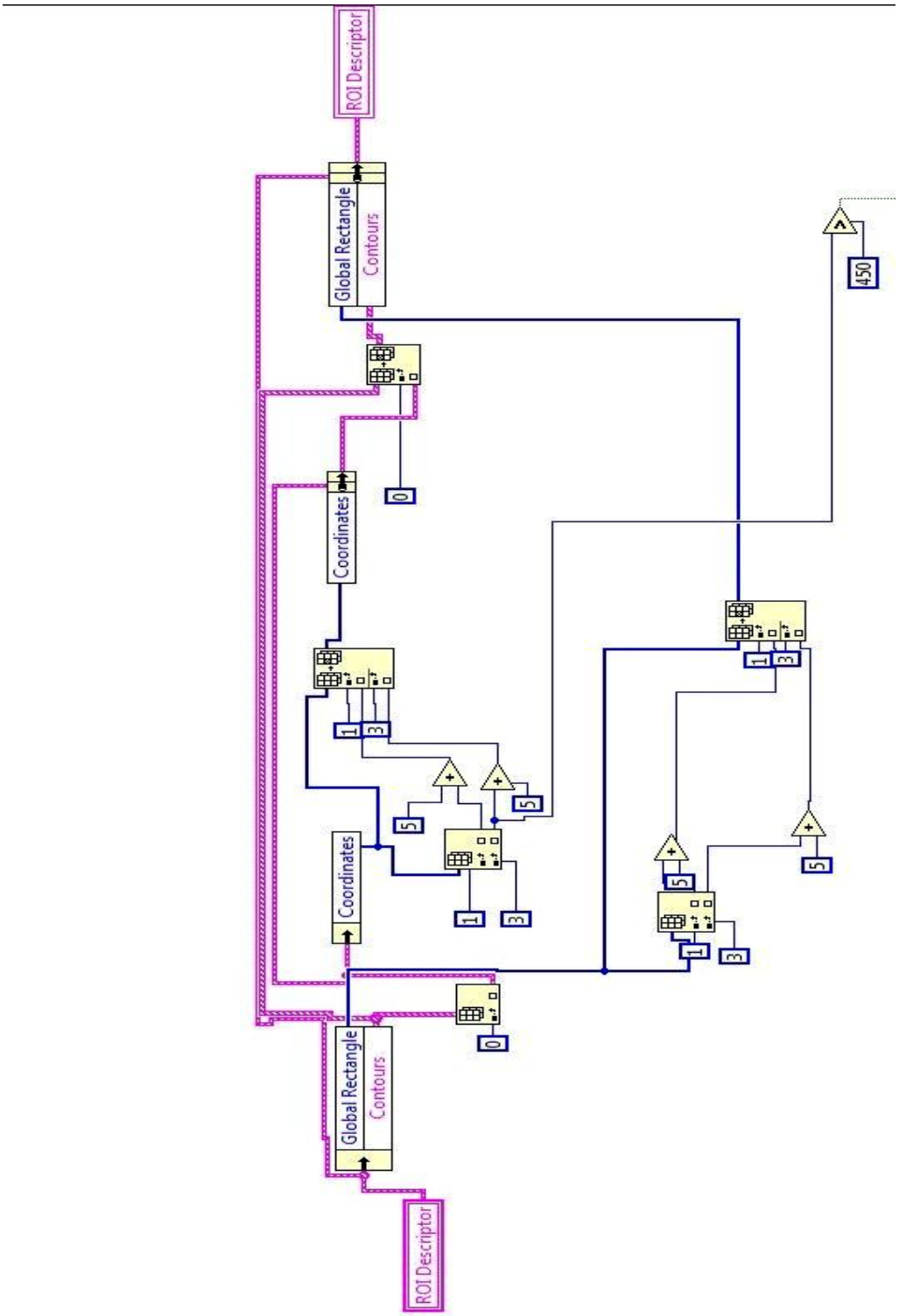


Fig 5.8 A part of LabVIEW block diagram for Character scanning in Masked region

We have changed this strategy of OCR reading by unbundling the ROI descriptor and to make it search the whole masked region from top to bottom. And for this we have

supplied the coordinates starting from top and incrementing them after each reading till the bounding box reaches the end of the masked image. An increment of three pixels at a time is made the bounding box reads the region inside the supplied coordinates and gives the output as read numbers those numbers then checked against a " **?** " sign and "**space**" whenever any of these two is present in the read number, the OCR rejects that entry and moves to next location with new coordinates.

Fig 5.9 Digits being recognized by OCR session

The sign of interrogation comes as output whenever OCR encounters an unknown character. If we have not trained the software for all the characters the OCR will give result as "?". Spaces are present before and after the numbers read if they are present then again the software rejects that entry and moves to next location. Sometimes the OCR reads characters without any spaces or signs of interrogation but it do not read all the digits due to constraints specified. Therefore the total number of digits are counted and if they are greater than a specified limit the number is displayed on the screen. This cross check is included in the software because Indian vehicles have a total of 9 or more digits in the number plate as **"CH 03 Y 2288".**

Whenever all specified conditions are fulfilled and the number is displayed on screen the program terminates itself from further checking the masked region. Its time saving and makes the software perform faster.

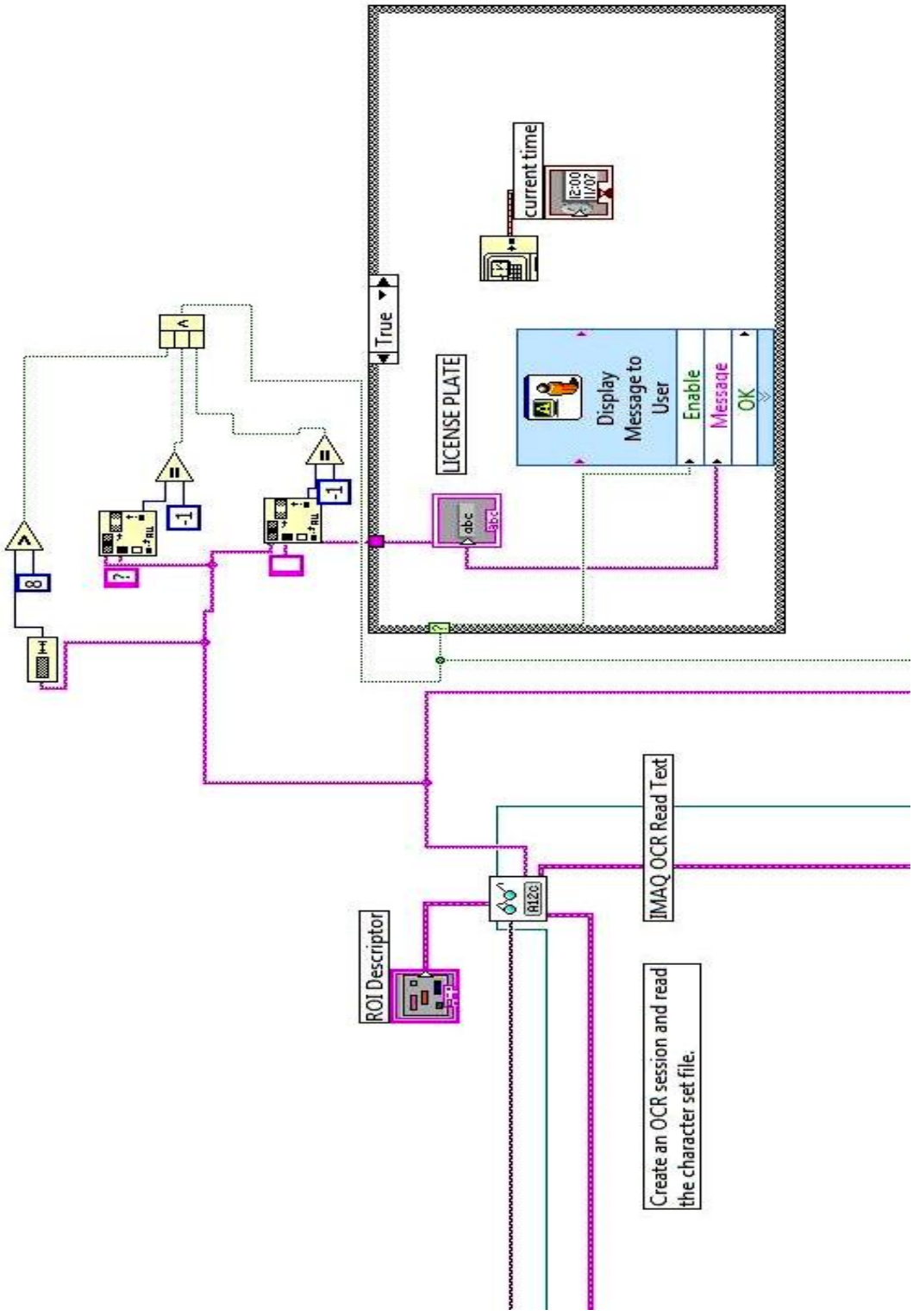


Fig 5.10 A part of LabVIEW block diagram for checking number in the spread sheet

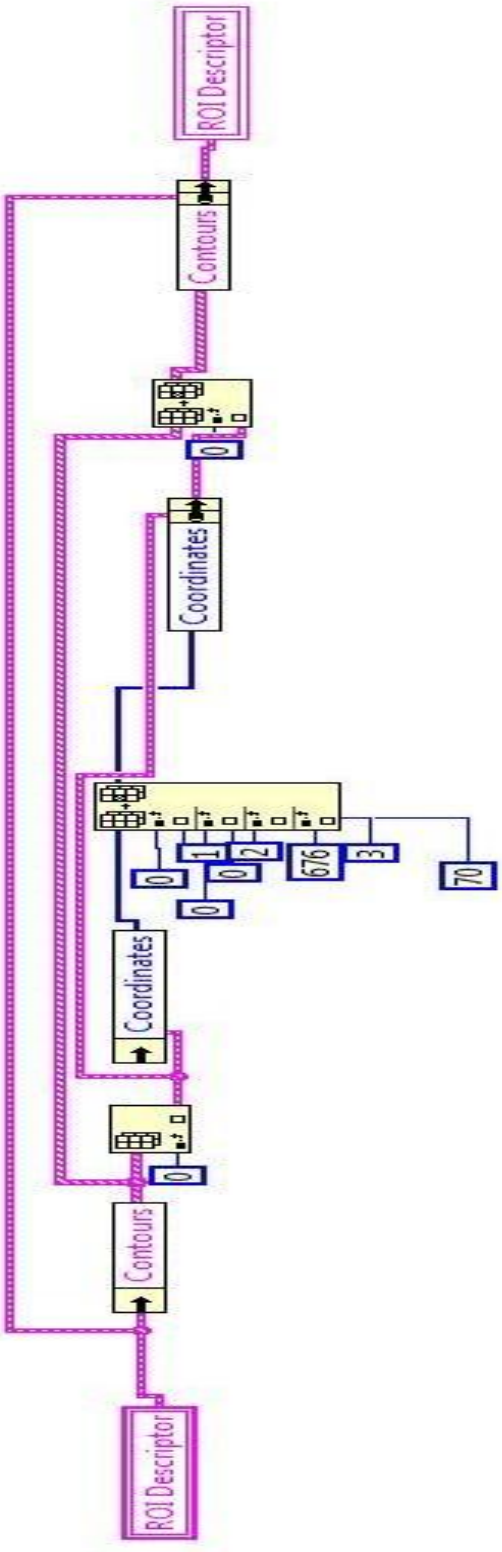


Fig 5.11 A part of LabVIEW block diagram for continuous updating of global coordinates.

In some cases the program completes all the iterations and checks all the conditions but fails to read all the characters that is when there is any condition which remains unsatisfied then it displays the error message to user as **" Sorry unable to detect the number plate".** Figure 5.14 shows the block diagram for the error message.

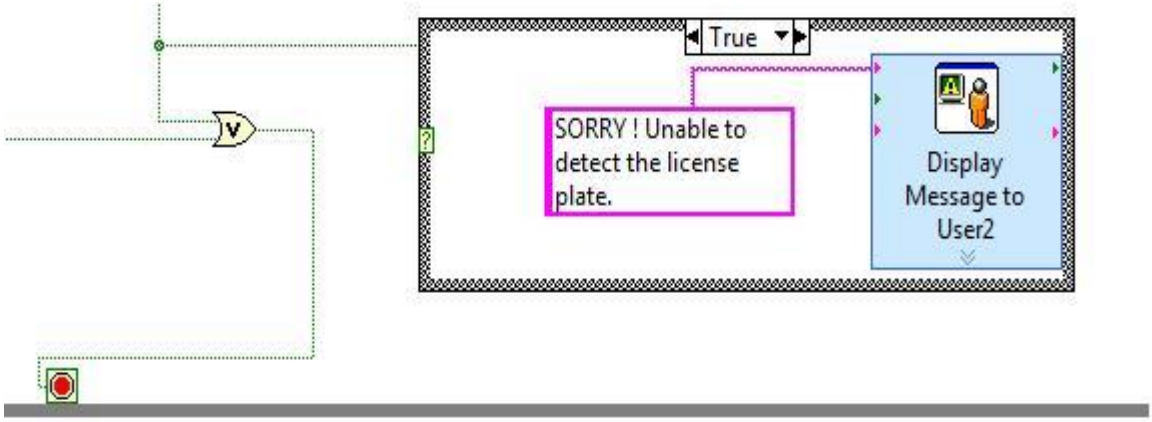


Fig 5.12 A part of LabVIEW block diagram for displaying error message if the software is not able to detect the number plate.

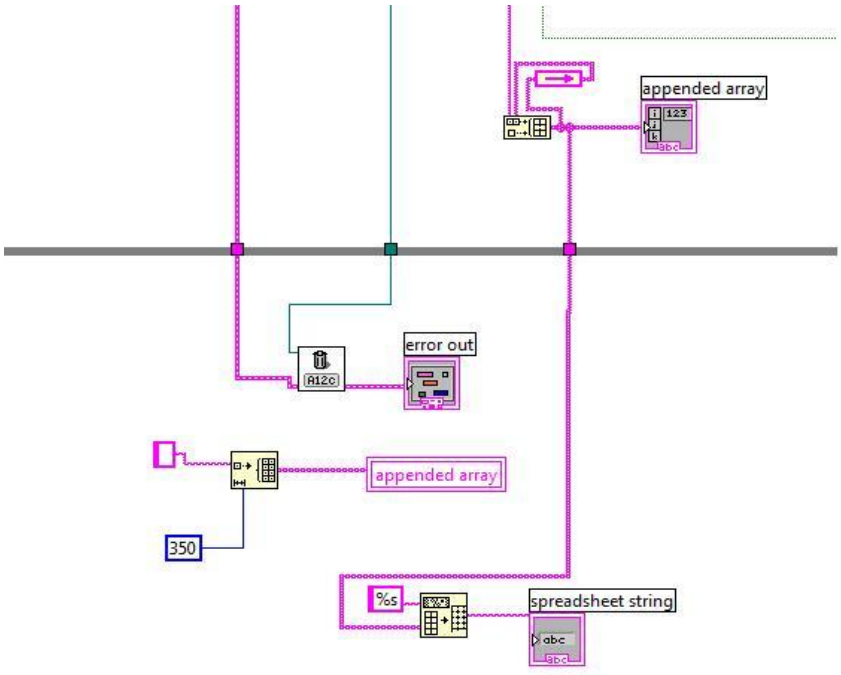


Fig 5.13 A part of LabVIEW block diagram for displaying the number plate in spread sheet.

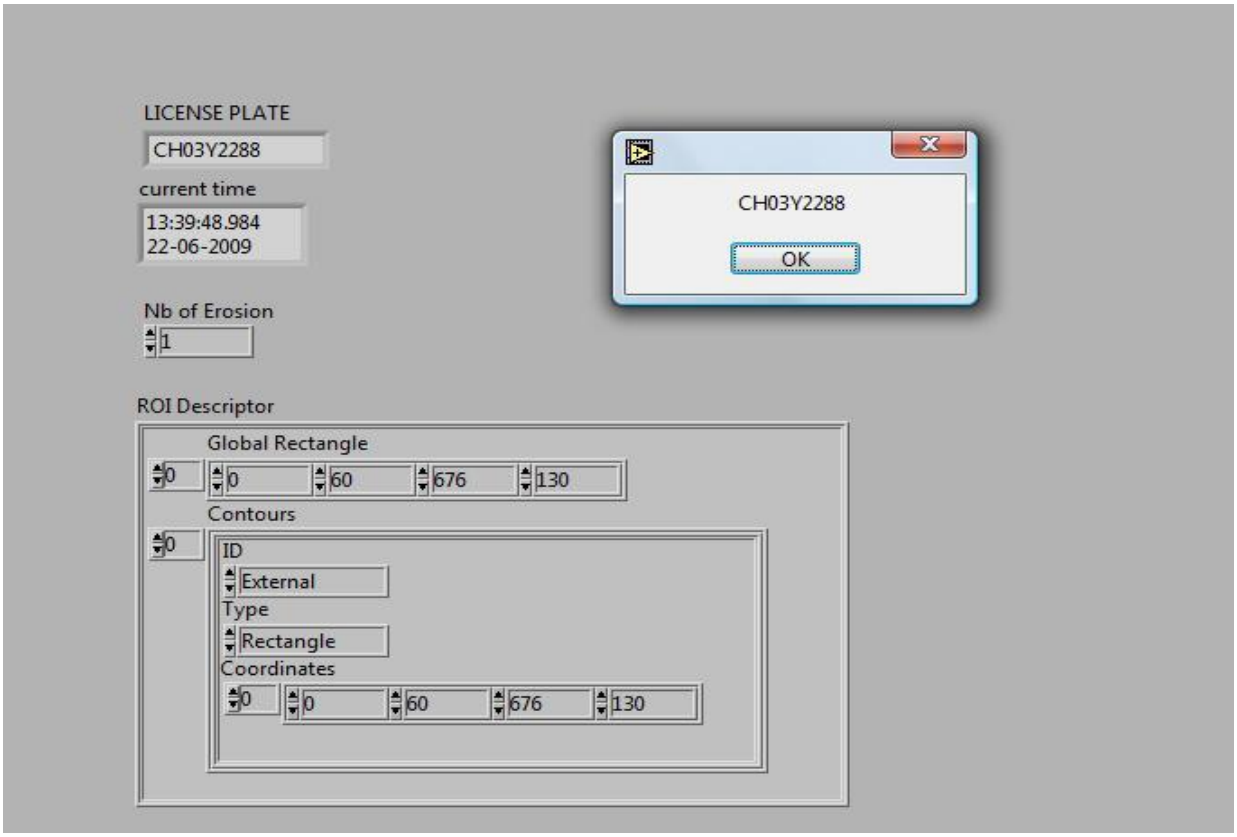
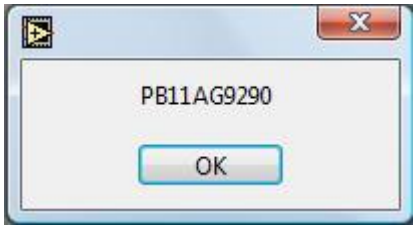
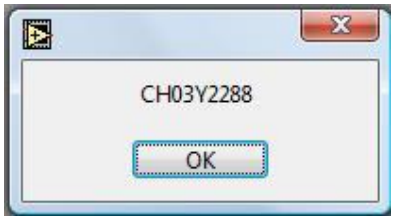
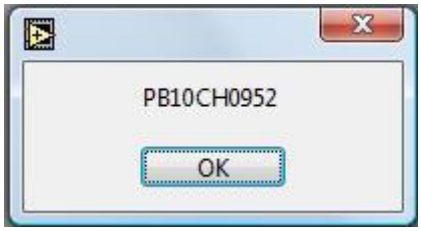
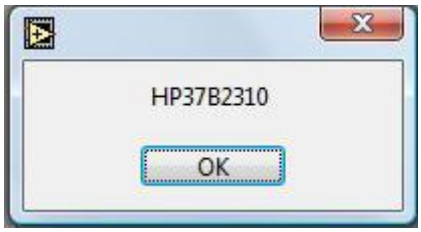


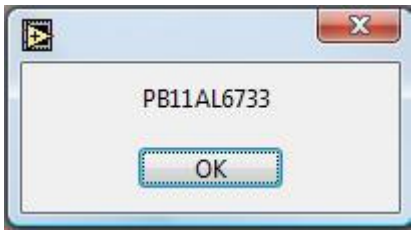
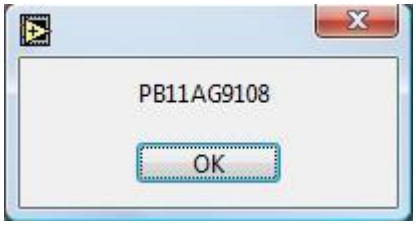
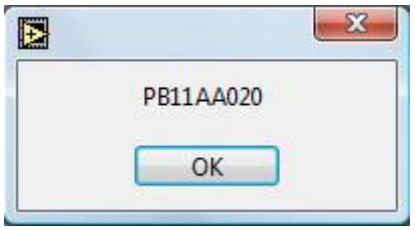
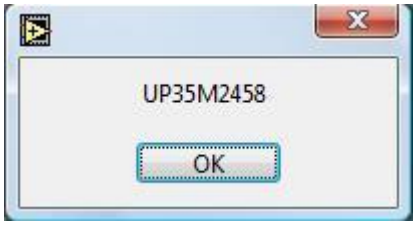
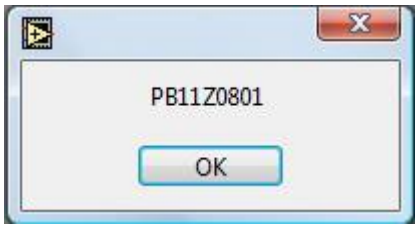
Fig 5.14 Front Panel view for displaying number plate

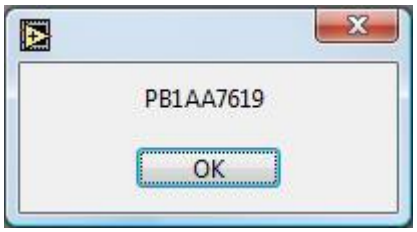
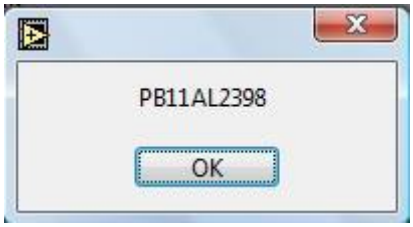
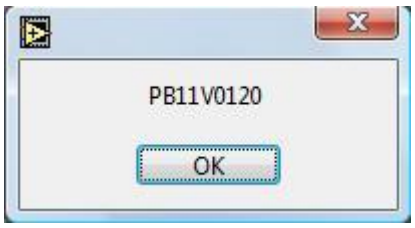
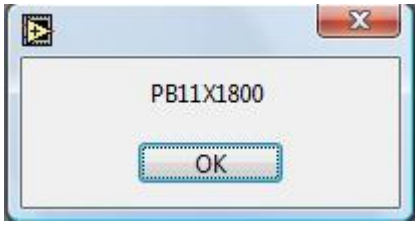
But if everything goes fine and all the conditions are satisfied then the program displays the number on screen as a pop up window. The block diagram for displaying the number on screen is shown in figure 5.13 above. And the corresponding front panel is shown in figure 5.14. As we can see the front panel shows the vehicle number and the time and date stamp at which the vehicle has passed. Once the number plate is recognized and displayed on the screen. The number can be checked against a database for granting entry to a parking or to another country. If the system is online and connected to local police stations then the number can be checked against the stolen vehicle number database and if it matches in it the vehicle can be taken in custody. The system can be used for homeland security and if it is installed on every police patrolling van then it can be used to monitor the vehicles on roads as well as parked in the parking. For human beings it's tough to remember all the numbers which may be useful for investigating a case but the system can work 24x7 without forgetting anything. All we need to do is make database of numbers we are interested in and then make our software check against the database each time it recognize any number plate.

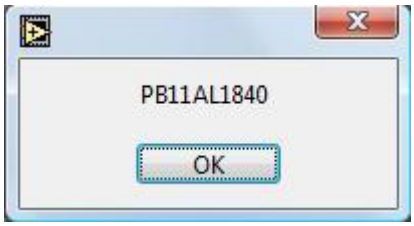
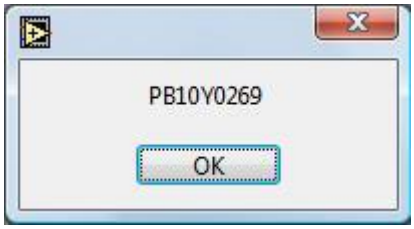
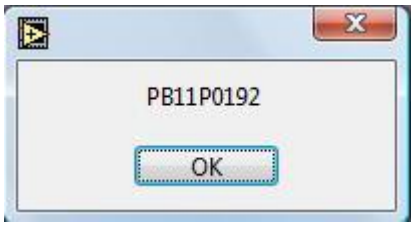
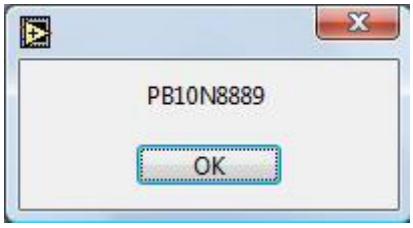
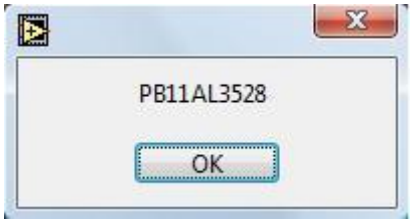
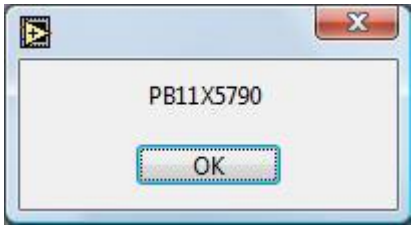
Software Testing

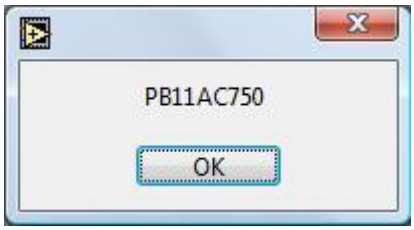
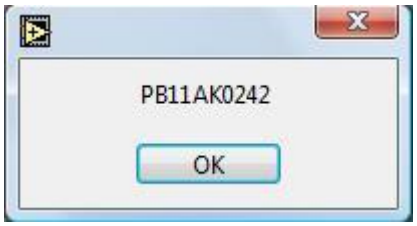
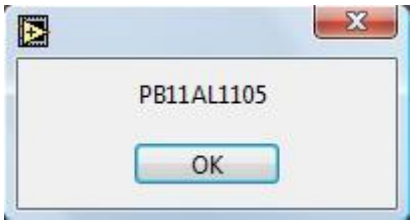
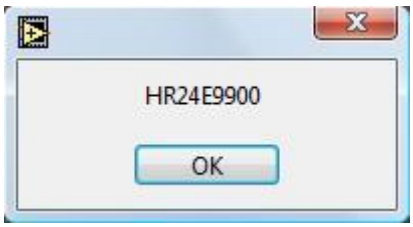
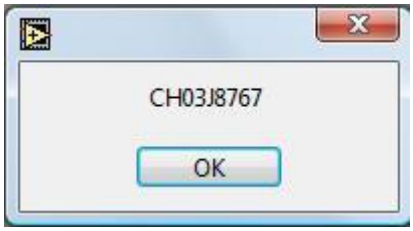
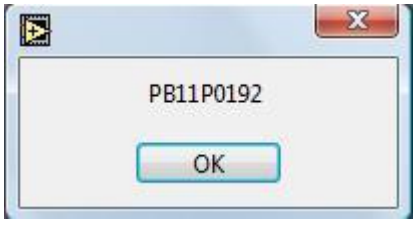
The software has been tested for 100 different vehicle images. The results of some images are as under. The table below consist vehicle image, its correct number, number read by our system, and the result.

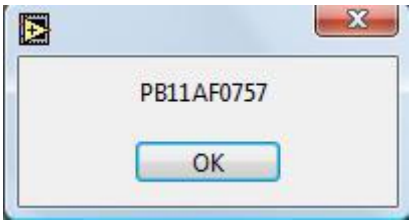
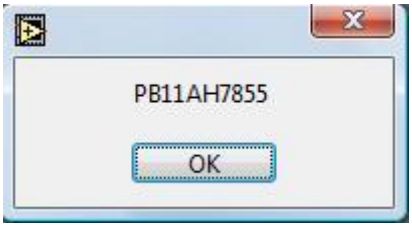
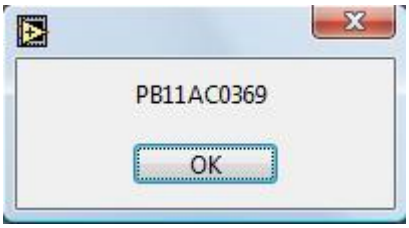
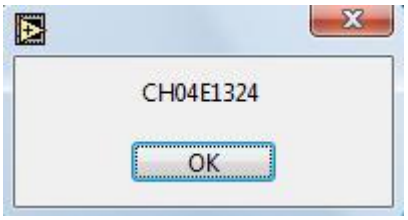
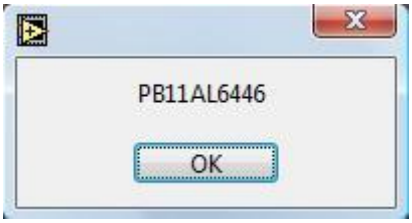
Sr.No	Input image	Correct Number	Number read by our system	Result
1		(Director Thapar university) PB11AG9290		SUCCESSFULL
2		CH03Y2288		SUCCESSFULL
3		PB10CH0952		SUCCESSFULL
4		HP37B2310		SUCCESSFULL

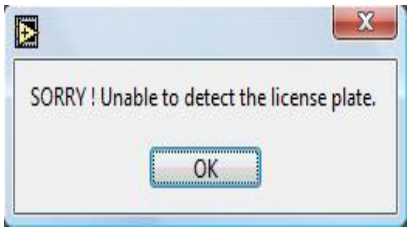
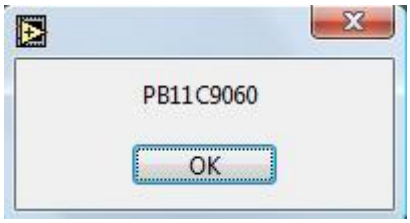
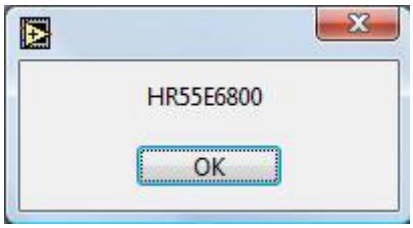
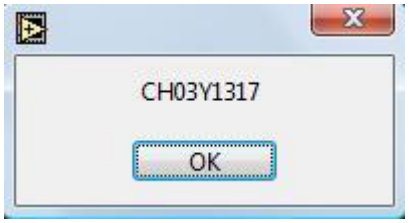
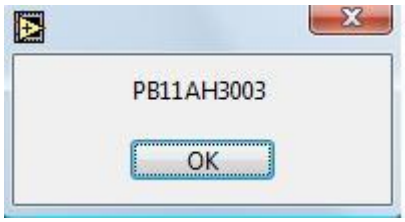
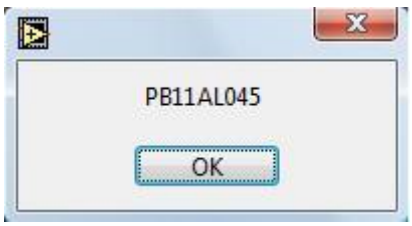
5		PB11AL6733		SUCCESSFULL
6		PB11AG9108		SUCCESSFULL
7		PB11AA0204		NOT 100% SUCCESSFULL
8		UP35M2458		SUCCESSFULL
9		PB11Z0801		SUCCESSFULL

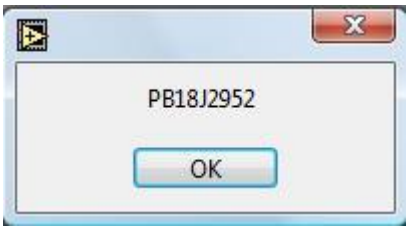
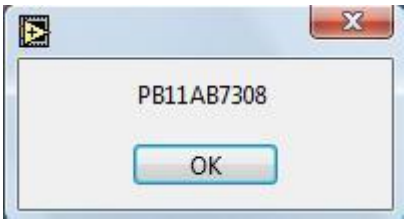
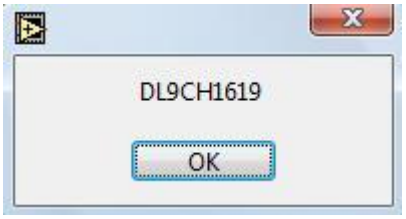
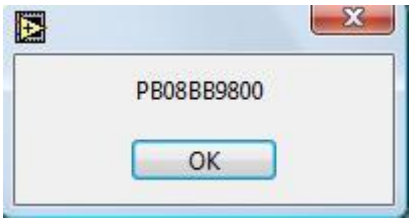
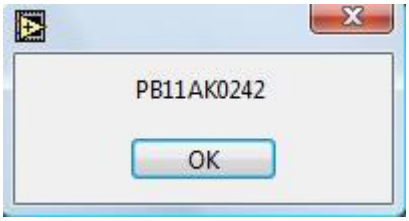
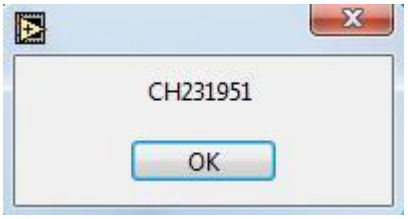
10		PB1AA7619		SUCCESSFULL
11		PB11AL2398		SUCCESSFULL
12		PB11V0120		SUCCESSFULL
13		CH01Y6097		SUCCESSFULL
14		PB11X1800		SUCCESSFULL

15		PB11AL1840		SUCCESSFULL
16		PB10Y0269		SUCCESSFULL
17		PB11P0192		SUCCESSFULL
18		PB10N8889		SUCCESSFULL
19		PB11AL3528		SUCCESSFULL
20		PB11X5790		SUCCESSFULL

21		PB11AC750		SUCCESSFULL
22		PB11AK0242		SUCCESSFULL
23		PB11AL1105		SUCCESSFULL
24		HR24E9900		SUCCESSFULL
25		CH03J8767		SUCCESSFULL
26		PB11P0192		SUCCESSFULL

27		PB11AF0757		SUCCESSFULL
28		DL9CE3476		SUCCESSFULL
29		PB11AH7855		SUCCESSFULL
30		PB11AC0369		SUCCESSFULL
31		CH04E1324		SUCCESSFULL
32		PB11AL6446		SUCCESSFULL

33		PB08AS5361		FAILURE
34		PB11C9060		SUCCESSFULL
35		HR55E6800		SUCCESSFULL
36		CH03Y1317		SUCCESSFULL
37		PB11AH3003		SUCCESSFULL
38		PB11AL045		SUCCESSFULL

39		PB18J2952		SUCCESSFULL
40		PB11AB7308		SUCCESSFUL
41		DL9CH1619		SUCCESSFUL
42		PB08BB9800		SUCCESSFULL
43		PB11AK0242		SUCCESSFULL
44		CH231951		SUCCESSFULL

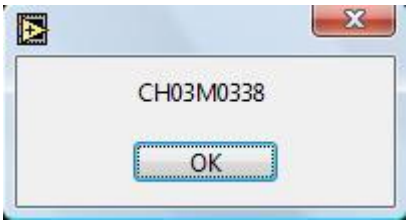
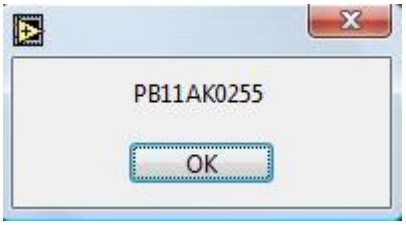
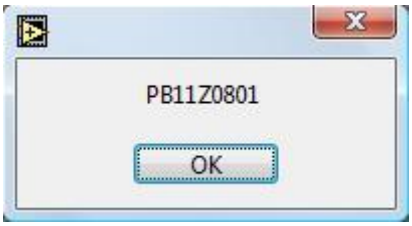
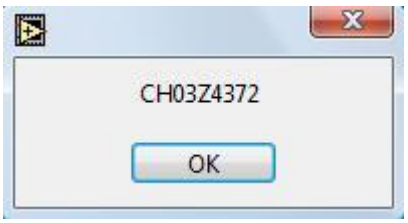
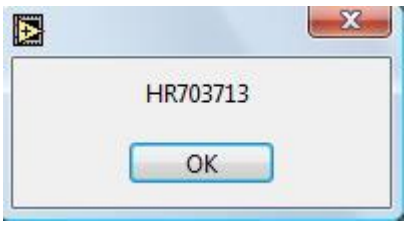
45		UK07U9520		SUCCESSFUL
46		CH03M0338		SUCCESSFUL
47		PB11AK0255		SUCCESSFULL
48		PB11Z0801		SUCCESSFULL
49		CH03Z4372		SUCCESSFULL
50		HR703713		SUCCESSFULL

Table 1 Results of 50 out of 100 samples tested by the system

5.3 Problems Encountered

The major problems that we faced during our project are discussed below.

1. There is no standard size of Indian number plates no standard of font style or size either which makes it quiet difficult for the software to recognize the alphanumeric characters because training the software for a specific size, shape and style is easy as compared to different. We need to give extensive training to the software in order to make it able to recognize the numbers in different cities and states as all the states have different first two characters in the vehicle number plates.
2. In image processing the systems are greatly affected by lightening conditions, our system is also sensitive to lightening conditions and lightening conditions are hard to kept constant while you are working in the middle of a busy road.
3. The proposed system is sensitive also to the angle at which images are being taken. For better efficiencies the image must be taken In a way so that vehicle number plate comes in the middle of 1200x1600 resolution picture. Also the resolution of images must be kept 1200x1600 for better results.

Chapter 6
Conclusions and Future Scope

Conclusions

The process of vehicle number plate recognition requires a very high degree of accuracy when we are working on a very busy road or parking which may not be possible manually as a human being tends to get fatigued due to monotonous nature of the job and they cannot keep track of the vehicles when there are multiple vehicles are passing in a very short time .To overcome this problem, many efforts have been made by the researchers across the globe for last many years. A similar effort has been made in this work to develop an accurate and automatic number plate recognition system. We have used Vision assistant 7.1 along with LabVIEW 7.1 to obtain the desired results. The setup has been tested for 100 vehicles containing different number plates from different states. In the process of final evaluation after optimizing the parameters like brightness, contrast and gamma, adjustments, optimum values for lightening and the angle from which the image is to be taken. We get an overall efficiency of 98% for this system. Though this accuracy is not acceptable in general, but still the system can be used for vehicle identification. It may be concluded that the project has been by and far successful. It can give us a relative advantage of data acquisition and online warning in case of stolen vehicles which is not possible by traditional man handled check posts. While thousands of vehicles pass in a day.

Future Scope

Though we have achieved an accuracy of 98% by optimizing various parameters, it is required that for the task as sensitive as tracking stolen vehicles and monitoring vehicles for homeland security an accuracy of 100% cannot be compromised with. Therefore to achieve this, further optimization is required.

Also, the issues like stains, smudges, blurred regions & different font style and sizes are need to be taken care of. This work can be further extended to minimize the errors due to them.

References

[1] Hu, M. K., "Visual Pattern Recognition by Moment Invariant", IRE Transaction on Information Theory, vol IT- 8, pp. 179-187, 1962.

[2] Khotanzad, A., and Hong, Y.H., "Invariant image recognition by zeraike moments,"

IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 12, no. 5, pp. 489-497,1990.

[3] Khotanzad, A., and Hong, Y.H., "Rotation in-variant image recognition using features selected via a systematic method," Pattern Recognition, vol. 23, no. 10, pp. 1089-1101, 1990.

[4] Belkasim, S.O., Shridhar, M., and Ahmadi, A., "Pattern Recognition with moment invariants: A Comparative study and new results," Pattern Recognition, vol. 24, pp.

1117-1138,1991.

[5] Lee, E. R., Earn, P. K., and Kim, H. J., "Automatic recognition of a car license plate using color image processing", IEEE International Conference on Image Processing 1994, vol. 2, pp.301-305, 1994.

[6] Comelli, P., Ferragina, P., Granieri. M. N., and Stabile, F., "Optical recognition of motor vehicle license plates", IEEE Transactions on Vehicular Technology, vol. 44, no. 4, pp: 790-799,1995.

[7] Morel, J., and Solemini, S., "Variational Methods in Image Segmentation", Birkhauser, Boston, 1995.

[8] Nieuwoudt, C, and van Heerden, R., "Automatic number plate segmentation and recognition", Seventh annual South African workshop on Pattern Recognition, pp. 88-93, IAPR, 1996.

[9] Kim, G. M., "The automatic recognition of the plate of vehicle using the correlation coefficient and Hough transform", Journal of Control, Automation and System Engineering, vol. 3, no.5, pp. 511-519, 1997.

[10] Cho, D. U., and Cho, Y. Ft., "Implementation of pre-processing independent of environment and recognition and template matching ", The Journal of the Korean Institute of Communication Sciences, vol. 23, no. 1, pp. 94-100, 1998.

[11] Park, S. FL, Kim, K. I., Jung, K., and Kim, H. J., "Locating car license plates using neural network", IEE Electronics Letters, vol.35, no. 17, pp. 1475-1477, 1999.

[12] Naito, T. Tsukada, T. Yamada, K. Kozuka, K. and Yamamoto, S., "Robust recognition methods for inclined license plates under various illumination conditions outdoors", Proceedings IEEE/IEEJ/JSAI International Conference on Intelligent Transport Systems, pp. 697-702,1999

[13] Naito, T., Tsukada, T., Yamada, K., Kozuka, K., and Yamamoto, S., "License plate recognition method for inclined plates outdoors", Proceedings International Conference on Information Intelligence and Systems, pp. 304-312, 1999.

[14] Naito, T. Tsukada, T. Yamada, K. Kozuka, K. and Yamamoto, S., "Robust recognition methods for inclined license plates under various illumination conditions outdoors", Proceedings IEEE/IEEJ/JSAI International Conference on Intelligent Transport Systems, pp. 697-702,1999.

[15] Salagado, L., Menendez, J. M., Rendon, E., and Garcia, N., "Automatic car plate detection and recognition through intelligent vision engineering", Proceedings of IEEE 33r Annual International Carnahan Conference on Security Technology, pp. 71-76, 1999.

[16] Naito, T., Tsukada, T., Yamada, K.s Kozuka, K., and Yamamoto, S., "Robust license-plate recognition method for passing vehicles under outside environment", IEEE Transactions on Vehicular Technology, vol: 49 Issue: 6, pp: 2309-2319, 2000.

[17] Kim, K. K., Kim, K. I., Kim, J.B., and Kim, H. J., "Learning based approach for license plate recognition", Proceedings of IEEE Processing Society Workshop on Neural Networks for Signal Processing, vol. 2, pp: 614-623, 2000.

[18] Yu, M., and Kim, Y. D., "An approach to Korean license plate recognition based on vertical edge matching", IEEE International Conference on Systems, Man, and Cybernetics, vol. 4, pp. 2975-2980, 2000.

[19] Yan, Dai., Hongqing, Ma., Jilin, Liu., and Langang, Li, "A high performance license plate recognition system based on the web technique, Proceedings IEEE Intelligent Transport Systems, pp. 325-329, 2001.

[20] Yan, Dai., Hongqing, Ma., Jilin, Liu., and Langang, Li, "A high performance license plate recognition system based on the web technique, *Proceedings IEEE Intelligent Transport Systems,* pp. 325-329, 2001.

[21] Hontani, H., and Koga, T., "Character extraction method without prior knowledge on size and information", Proceedings of the IEEE International Vehicle Electronics Conference (IVEC'01), pp. 67-72, 2001.

[22] Cowell, J., and Hussain, F., "Extracting features from Arabic characters",Proceedings of the IASTED International Conference on COMPUTER GRAPHICS AND IMAGING, Honolulu, Hawaii, USA, pp. 201-206, 2001.

[23] Hansen, H., Kristensen, A. W., Kohler, M. P., Mikkelsen, A. W. , Pedersen J. M., and Trangeled, M., "Automatic recognition of license plates", Institute for Electronic System, Aalhorg University, May 2002.

[24] Cowell, J., and Hussain, F., "A fast recognition system for isolated Arabic characters", Proceedings Sixth International Conference on Information and Visualisation, IEEE Computer Society, London, England, pp. 650-654, 2002.

[25] Hamami, L., and, Berkani, D., "Recognition System for Printed Multi-Font and Multi-Size Arabic Characters", The Arabian Journal for Science and Engineering, vol. 27, no. IB, pp. 57-72, 2002.

[26] Optasia Systems Pvt Ltd, http ://www. Singapore gateway. com/optasia/imps ,Singapore.

[27] Percerptics, http://www.perceptics.com/lpr.html , northrop grumman information technology, USA.

[28] Parking Products, Inc., http://www.parkingproducts.com/. Vehicle Identification System for Parking Areas (VISPA), USA, 2002.

[29] Hi-Tech Solutions, http://www.htsol.com/, Israel.

[30] LabVIEW Machine Vision and Image Processing Course Manual

[31] NI Vision Assistant tutorial manual.