Hand-Written Roll Number Recognition Followed by Non-Uniform Illumination Removal From Digital Images

Developed by

Asim Munir

Supervised by

Dr. M. Sikandar Hayat Khiyal

International Islamic Univ.
CENTRAL
LIBRARY
ISLAMABAD.

Department of Computer Science

Faculty of Applied Sciences

International Islamic University, Islamabad.

2005

Department of Computer Sciences

International Islamic University Islamabad

Date 31-10-2005

Final Approval

It is certified that we have read the project report submitted by Asim Munir, Registration number 43-CS/MS/02, and it is our judgment that this project is of sufficient standard to warrant its acceptance by the International Islamic University, Islamabad for the MS Degree in Computer Science.

Committee:

1. External Examiner
   Dr. Abdus Sattar
   House No. 143, St No. 60,
   Sector i-8/3
   Islamabad.

2. Internal Examiner
   Dr. Syed Afaq Hussain
   Head,
   Department of Computer Engineering &
   Department of Telecommunication Engineering,
   Faculty of Applied Sciences,
   International Islamic University, Islamabad.

3. Supervisor
   Dr. M. Sikandar Hayat Khiyal
   Head,
   Department of Computer Science,
   Faculty of Applied Sciences,
   International Islamic University,
   Islamabad.

A dissertation submitted to the
Department of Computer Science,
International Islamic University, Islamabad
as a partial fulfillment of the requirements
for the award of the degree of
Masters of Science in Computer Science

This project is lovingly dedicated to

To My Beloved Parents

Who provided unflagging support

Declaration

I, Asim Munir S/O Muhammad Munir, hereby declare and affirm that this software neither as a whole nor as a part thereof has been copied out from any source. It is further declared that we have developed this software and accompanied report entirely on the basis of my personal efforts, made under the sincere guidance of my teachers. If any part of this project is proven to be copied out or found to be a reproduction of some other, I shall stand by the consequences.

No portion of the work presented in this report has been submitted in support of an application for other degree or qualification of this or any other University or Institute of Learning.

Asim Munir

43-CS/MS/02

Acknowledgement

This project and all my efforts are fruitful only due to Allah Almighty, the Most Merciful and Beneficial Who gave me strength to complete this task to the best of abilities and knowledge.

I would like to take this opportunity of thanking my supervisor **Dr. M. Sikandar Hayat Khiyal,** who gave all his knowledge, guidance and support to boost my confidence and learning. He was always there to solve problems and helped me through this project. I am thankful to him for this.

Last but not the least; I would like to acknowledge **Mr. Muhammad Zaheer Aziz**, who provided me support with his technical help. I would like to admit that I owe all my achievements to my truly, sincere and most loving parents, sister, brothers and friends who mean the most to me, and whose prayers have always been a source of determination for me.

Asim Munir

Project In Brief

Project Title:	Handwritten Roll Number Recognition Followed by Non-Uniform Illumination Removal
Undertaken By:	Asim Munirr
Supervised By:	Dr. M. Sikandar Hayat Khiyal
Starting Date:	28th October, 2003
Completion Date:	20th January, 2005
Tool Used:	Microsoft Visual C++ 6.0 Matlab 6.0
Operating System:	Microsoft Wondows 2000 Professional
System Used:	Pentium IV

Abstract

The main subject of the present thesis is the use of projection method to find out the features of digit to be recognized after balancing the background irregular illumination in the given digital image.

During the acquisition of monochrome images, the classic monochrome aberrations and the various sources of non-uniformity of the illumination of the image plane makes the image undesirable for direct processing of any digital image application.

High-order Gaussian convolution kernel is used to estimate the illumination part of the digital image to produce a blurred image. Then using the illumination-reflectance model, reflectance part is obtained, as we have input image. A constant illumination addition to the image and other pre-processing methods produce an enhanced image.

Recognition module performs its role in a much better way if it is enhanced to remove noise and non-homogenous illumination. After obtaining the coordinates that enclose the character to be recognized, features based on the depth projections from left, right, top and bottom directions are obtained to be matched with the available test data to uniquely identify it.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1. Introduction

The perception in the human eye takes place by the relative excitation of light receptors. These transform radiant energy into electrical impulses that are ultimately decoded by brain. Subjective brightness that is energy perceived by the human visual system is a logarithmic function of the light incident on the eye. Visual system cannot operate over a large range offered by light. For any given set of conditions, the current sensitivity level of the visual system is called brightness adoption.

This inhomogeneity is found in mostly images taken by camera, scanner or any acquisition device on one hand causes human eye to use a small range of intensity levels that eye can discriminate. This allows loss of valuable information from the image especially due to background. On the other hand such a range may disturb the image processing operations such as noise removal, segmentation, recognition and classification of objects.

These problems may be solved by tackling the non uniform illumination present in the image that was introduced during the acquisition process.

One of the most interesting aspects of the world is that it can be considered to be made up of patterns. Any pattern can be described as the arrangement of features used to identify it uniquely from the other elements of the same class.

After dealing with improper background light, hand-written digits can be effectively recognized using the descriptors selected.,Many other digit recognition techniques are widely used at present to obtain good results but the technique presented is novice in the sense that claims its feature to be free of size and thickness of pen.

Hand-written recognition followed by enhancement is carried out to exhibit the details that may be corrupted due to improper source light providing illumination to the answer sheet or changing shadows during the course of the day.

1.1 Removal of Non-Uniform Illumination

Dealing with improper or unbalanced illumination comes under the fundamental step of digital image processing called image enhancement. The main aim of enhancement technique is to explore the details that were obscured during the acquisition process of the image. Enhancement is the subjective area of image processing.

Image Enhancement methods fall into two main categories:

1.1.1 Spatial Domain

In this approach pixels comprising an image are directly manipulated. Typically, a transformation function is selected does a mapping of pixels when applied. At any point in an image, enhancement depends only on pixel at that point, techniques in this category are referred to as point-processing.

Another approach allows using filters, (masks or kernels) that operate in a predefined neighborhood of any pixel to determine its new value in the convolution process.

1.1.2 Frequency Domain

As an image is two-dimensional signal, its Fourier Transform will give all the sinusoids it is composed of but no time information is available. Where as its inverse will produce the same image as that before taking Fourier transform giving its spatial arrangement.

After obtaining the Fourier transform, a filter function is applied that attenuates certain frequencies allowing the other frequencies to pass. Taking its inverse will produce an enhanced image based on the filter function applied.

Gaussian low pass of high order is selected to approximate the unbalanced light in the back ground of image. The filter takes in to account up to seven neighbors on both sides to obtain the brightness problem extent. Once the pattern of back ground light containing problem of illumination is know, this may be used to enhance the image to be used in a variety of applications.

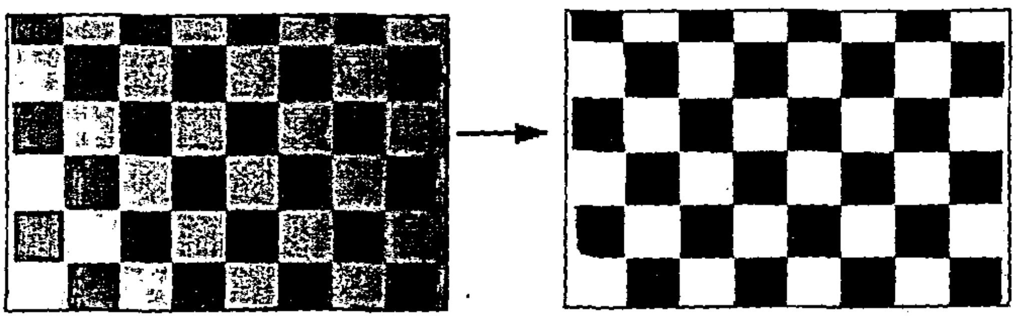

Figure 1.1 Removal of Non Uniform Illumination

Any application that performs segmentation, digit and/or character recognition and classification does need a pre-processing step. Same is true for this application where an effective recognition is dependent on how refined input image is being input to the recognition step.

1.2 Recognition of Hand-Written Digits

Document processing applications can be found in almost all computer systems and now is become widespread. Applications like text edition, desktop publishing and graphics are often used for most organizations and home offices. This technology has recently gone through a remarkable growth. Vision process functions and visual patterns recognition, are fields of major interest for many of the researches. ,

Character recognition, known as OCR (Optical Character Recognition) is an important subset within the pattern recognition area. OCR applications established some years ago the basis for the works within the research community in order to recognize and clarify pattern recognition and image

An important sub problem is the problem of recognizing hand-printed digits. This is not only so because there exist a significant number of practical applications for a system that reliably recognizes handwritten digits, but also because it is an important test case for more general recognition problems.

The notion of using computation to "recognize" digits as one of 10 classes (0 - 9) is an intriguing idea. This idea has led to many interesting algorithms and techniques for digit recognition, each with their own motivations

1.3 Purpose of the System

The goal of the research described is the construction of an intelligent system for the automatic recognition of the hand-written roll numbers in the space given on the answer sheet of entry tests. The recognition process under investigation is divided into 4 main steps: image acquisition; image pre-processing; finding minimum enclosing rectangle, and finally recognition based on traces obtained based upon shapes of digits.

The performance of an automatic recognition system depends on the quality of documents in its both forms original and digital. Many different approaches are used on the trial to compensate poor quality in the originals and in the captured images such as: contrast and noise level reduction. The problems related with quality and general picture handling are:

1. *Noise* – unconnected line segments, pixels, curves etc.
2. *Distortion* – Local variations, rounded corners, improper extrusions and etc.
3. *Style variation* – Different shapes represents the same characters including type-like serif slants and so on.
4. *Translation* – It represents relative shift of the character. It can be entirely or by parts.
5. *Scale* – Relative size of the character
6. *Rotation* – Orientation changes.
7. *Texture* – Variations in the paper texture and the writing handler.
8. *Trace* – Variations in the thickness.

Different approaches, such as statistical, syntactic and structural, and neural network approaches, have been proposed. Characters consist of line segments and curves. Different spatial arrangements of these elements form different characters. In order to recognize a character, we should find out the structural relationships between the elements which make up the character.
However, in the pattern recognition community, the syntactic and structural approach is always considered a nice idea, but is not particularly promising in most applications.
One of the concerns is the need for robust extraction of primitives. Using the structural approach, two-dimensional patterns, their method does not make use of directional information.

1.4 History

The research of character recognition starts on the years of 1870's with the creation of the retina scanner. This device is an image transmission system with the use of a photocell mosaic.

The sequential scanner created in 1890 was a real breakthrough in the development of the modern TV and optical reader devices. Character recognition itself had appeared as an aid system for blindness in the early 1900's. The digital computer development at 1940 introduced the modern version of OCR developed, at first, only for a limited business data processing application.

1.5 Main Features

Following are the main features resulted from the approaches selected for the application.

1.5.1 General Method of Enhancement for Variety of Applications

Removal of non-uniform illumination is the key benefit obtained that is valuable for all the applications that require classification of objects, recognition of patterns and other preliminary digital image processing methods.

Even all other models of enhancement start to work well once we improve the illumination that would have not resulted better other wise.

1.5.2 Computation Cost of Enhancement is Limited to Convolution Mainly

Low pass of an image contains gradually changing frequencies that is the main property of background. Gaussian low pass filter is selected to obtain the low pass by attenuating higher frequencies such that the actual image contents are not included in the blurred image. That results in a blurred image obtained by a convolution kernel of size 15x15 in one pass on the image getting illumination pattern with very less frequencies.

1.5.3 Recognition Process is Thickness Free

Descriptors are based on depth projection profile that does not require the process of thinning to be perfumed that is the need of many character and digit recognition methods.

The main concern prevents thinning process that is to estimate the pen movement on the paper, captured by drawing the projections. The projection of pen thickness is not required rather projection from object to rectangle's boundary gives us the description of digit shape.

1.5.4 Recognition can be Performed in Real Time.

Only binarization step and finding minimum enclosing rectangle around the digit is required to be performed in addition to projections of the digit that is confined only to small area. All these factors are suitable to make system as real-time digit recognizer.

1.5.5 Low Memory cost to Store Feature Vector for Digit

Peaks and trough resulted through depth projections are represented by 1 and 0 respectively. Only one bit is required to store the element. This allows saving a lot of memory in saving data on storage devices. While comparing this binary number can be converted into byte or word storage that will take space in memory rather on the storage device.

1.6 Future Enhancement

Exhaustive applications using digit recognition system and removal of non-uniform illumination are needed. Character recognition has been the subject of intensive research during the last decades. It is a very challenging scientific problem but provides a solution for processing large volumes of data automatically. Now a days character recognition, even of handwritten script has numerous applications such as address and zip code recognition, car number plate recognition, writer identification, etc.

Unique properties of signature allow identifying a person uniquely; the process may be manual or automated, in the same way hand-written text may also help to identify through the careful selection of features. The approach could be combine use of scalable vector temples instead of fixed and depth projection profile.

Enhancement plays an important role and background separation from the object to be recognized is also crucial. A care should be taken that reflectance component is not removed when illumination is tackled.

CHAPTER 2

LITERATURE SURVEY

2. Literature Survey

The approaches used in the project and their brief description is given below.

2.1 Image Enhancement

Techniques for editing an image such that it is more suitable for a specific application than the original image are generalized as image enhancement. The quality of the improvement can thereby actually only be graded in relation to the particular application domain. Main technique behind the enhancement in this work is dealing with the background improper illumination that takes the advantage of image constituents: Illumination and reflectance.

2.1.1 Image Composition

An image function is a mathematical representation of an image. The only attribute of achromatic light is its intensity, or amount. The term gray level generally is used to describe monochromatic intensity because it ranges from black, to grays, and finally to white.

Image function f(x,y) may be regarded as being characterized by two components: illumination i(x,y) and reflection r(x,y).

$$f(x,y) = i(x,y) * r(x,y)$$

where i(x,y) is determined by the light source, while r(x,y) is determined by the characteristics of the objects in a scene at coordinates (x, y). The value or amplitude of f at spatial coordinates (x, y) is a positive scalar quantity whose physical meaning is determined by the source of the image.

Both components are limited: $0 <= i(x) <= \infty$ and $0 <= r(x) <= 1$, correspond to the total absorption and total reflection. Often only r(x, y) is of importance, but with structured light and shape from shading i(x, y) plays an important role. In black and white images, f(x, y) is a scalar value and a vector value in multi-spectral images.

2.1.2 Problem During Acquisition

During the acquisition process of digital images, improper source of illumination may cause light effect on different parts of image. This may happen in an image acquired by X-ray, ultrasound machine or digital camera.

By considering the nature of non-uniformities in an image acquisition system due to the non-linear response of electronic devices and non-uniform lighting, methods can be devised to measure these non-uniformities to enable corrections to be made at the pre-processing stage. Recognizing objects from their surface reflectance properties (such as lightness and gloss) is a nontrivial accomplishment because of confounding effects of illumination.

2.1.3 Gaussian Filter for Smoothing in the spatial domain

The Gaussian smoothing operator is a 2-D convolution operator that is used to blur images and remove detail and noise. It also fills in of small gaps in lines, contours and planes. In this sense it is similar to the mean filter, but it uses a different kernel that represents the shape of a Gaussian hump.

In the spatial domain the simplest example of smoothing is the average or uniform filter:

$g(x,y) = (\Sigma_s f(n,m)) / M$ with M the number of pixels in the neighborhood of s around (x,y).

The Gaussian distribution in 1-D has the form:

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

where σ is the standard deviation of the distribution. It is also assumed that the distribution has a mean of zero (*i.e.* it is centered on the line $x=0$). The distribution is illustrated in Figure 2.1.

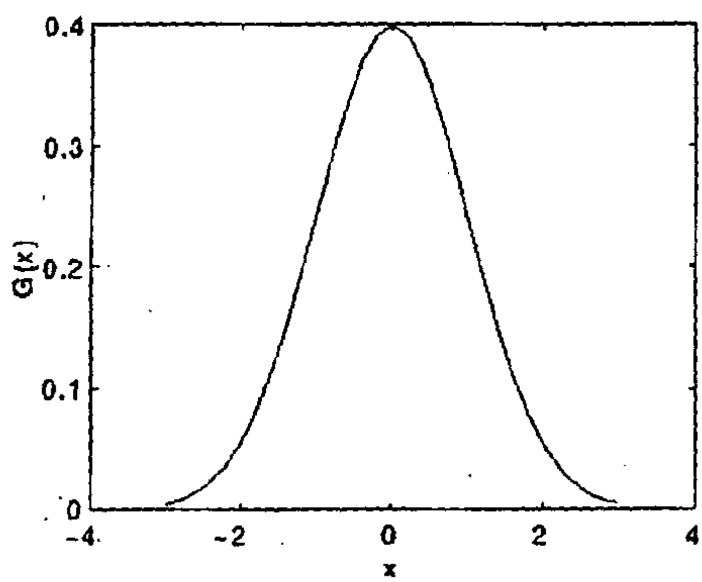

Figure 2.1 1-D Gaussian distribution with mean 0 and σ =1

The degree of smoothing is determined by the standard deviation of the Gaussian. Larger standard deviation Gaussians require larger convolution kernels in order to be accurately represented.

In 2-D, an isotropic (*i.e.* circularly symmetric) Gaussian has the form:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

This distribution is shown in Figure 2.2.

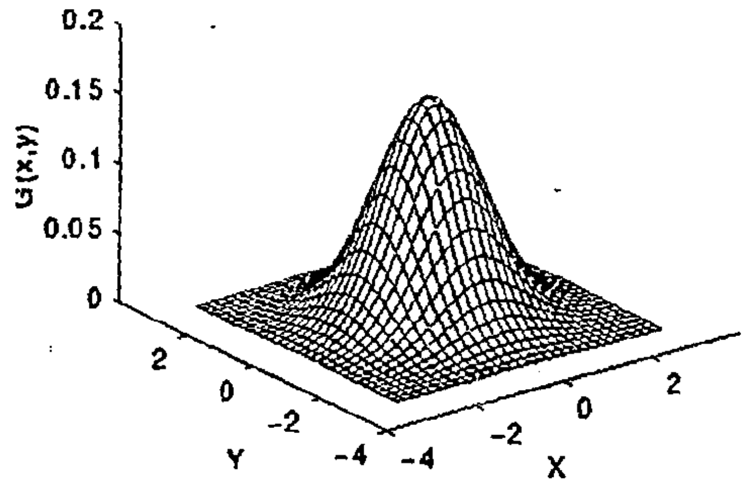

Figure 2.2 2-D Gaussian distribution with mean (0,0) and σ=1

The idea of Gaussian smoothing is to use this 2-D distribution as a point-spread function, and this is achieved by convolution. Since the image is stored as a collection of discrete pixels we need to produce a discrete approximation to the Gaussian function before we can perform the convolution.

The Gaussian is in fact the only completely circularly symmetric operator which can be decomposed in such a way.

2.1.3.1 Gentle Smoothing

The Gaussian outputs a 'weighted average' of each pixel's neighborhood, with the average weighted more towards the value of the central pixels. This is in contrast to the mean filter's uniformly weighted average. Because of this, a Gaussian provides gentler smoothing and preserves edges better than a similarly sized mean filter.

2.1.3.2 Low Pass Filter

One of the principle justifications for using the Gaussian as a smoothing filter is due to its frequency response. Most convolution-based smoothing filters act as lowpass frequency filters. This means that their effect is to remove high spatial frequency components from an image.

2.1.3.3 Selected Filter

As we have to deal with the non-uniform illumination changing gradually in background, a filter size of 15x15 is selected to accurately estimate the smoothness of light. Smaller filter sizes fail to approximate the illumination variation. Also Gaussian is a lowpass filter and high order lowpass of image also approximates the illumination part of the image, so it completes another argument of its selection to approximate the illumination in image.

2.1.4 Arithmetic Operations

Image function f(x,y) may be regarded as being composed of two components: illumination and reflection.

$$f(x,y) = i(x,y)\, r(x,y)$$

Once illumination of entire image is captured at relative coordinates of the image, reflectance part can be obtained in the following manner.

$$r(x,y) = f(x,y) / i(x,y)$$

where r(x,y) is the estimated reflectance that is the property of objects containing the image.

Multiplication of each spatial reflectance component is then multiplied with constant illumination making it suitable for further processing and visibility.

2.2 Digit Recognition

Applications have been developed to read postal addresses, bank cheque, vehicle number plate, tax forms and census forms, where typing of handwritten text is avoided. Each one of these requires some preprocessing steps to be performed making input in such a way to improve the recognition process.

2.2.1 Boundary Detection

Each digit to be recognized must be located as a connected region in the part of image with complete isolation from other regions constituting the image. An obvious practice is to build a rectangle with top-left and bottom-right corner enclosing the character to be recognized.

2.2.2 Minimum Enclosing Rectangle (MER)

The next step is to find the rectangle to tightly inscribe the character to be recognized. A minimum-enclosing rectangle (MER) of a line is defined by the minimum and maximum x and y coordinates of the line. Four lines on all sides are moved to wards the digit just before the boundary of digit begins as shown in figure 2.3.

Figure 2.3 Minimum Enclosing Rectangle Around Digit '5'

Projection calculation time and space between the selected boundary and minimum-enclosing rectangle can be saved. This will also prevent the noise present in this area that may contribute in deviating the final results.

2.2.3 Depth-Projection in Four Directions (Left, Top, Right, Bottom)

The better the descriptor is, the greater the difference in the descriptors of significantly different shapes and the lesser the difference for similar shapes. Regions can either describe boundary-based properties of an object or they can describe region-based properties.

A useful region-based signature is the *profile* or *projection*. The *vertical profile* is the number of pixels in the region in each column. The *horizontal profile* is the number of pixels in the region in each row. One can also define *diagonal profiles*, which count the number of pixels on each diagonal.
Profiles have been used in character recognition applications, where an L and a T have very different horizontal profiles (but identical vertical profiles), or where A and H have different vertical profiles (but identical horizontal profiles).
Profiles are useful as shape signatures, or they may be used separate different regions. A technique known as "signature parsing" uses vertical profiles to separate horizontally-separated regions, then individual horizontal profiles to recursively separate vertically-separated regions, etc. By alternating this process and recursively splitting each region, one "parse" binary objects that are organized horizontally and vertically, like text on a page.

(a) (b) (c) (d)

Figure 2.4 Depth Projections for digit '5'. (a) Left (b) Right (c) Bottom (d) Top

For left depth projection, profile is made by scanning first row of pixels from left to right till a pixel belonging to digit boundary is met. This process is repeated for next line to draw its profile line. Other side of the digit till the right extent of minimum enclosing rectangle will be covered during the generation of left side depth projection. This process is same for bottom and top depth projections.

2.2.4 Feature Extraction

In general, feature descriptors are some set of numbers that are produced to describe a given shape. The shape may not be entirely reconstructable from the descriptors, but the descriptors for different shapes should be different enough that the shapes can be discriminated.

Features in the proposed research are obtained by scanning the peaks and troughs in the projection profile in all the directions. Whenever if there is a peak followed by a trough we give a binary pattern 10, indicating "peak followed by deep profile" while scanning from left to right. Binary pattern 01 is just the reverse that is "trough followed by crest". 0 represents a constant trough that is the projection resulted from a line that appears in most characters.

Depending upon the nature of the problem domain the maximum length of a feature vector is made limited to 5- digit binary pattern in a direction. The features thus obtained are used to perform matching with already present data set obtained in the same way for all the digits ranging from 0 to 9.

2.3 Previous Research

Conventional image processing techniques are sensitive to non-uniform illumination and non-homogeneous background, which obstructs the derivation of reliable results for a large set of different images. For a successful digital image processing application, enhancement is crucial. A properly enhanced image may revolutionize the output results of segmentation and recognition process. Following are the different approaches to deal with improper illumination and the recognition of hand-printed characters and digits.

2.3.1 Non-Linear Enhancement and Segmentation Algorithm for The Detection of Age-Related Macular Degeneration (AMD) in Human Eye's Retina.

Rapantzikos and K. Zervakis [6] present a segmentation algorithm for automatic detection of abnormalities in images of the human eye's retina, acquired from a depth-vision camera.
Homomorphic filtering and a multilevel variant of histogram equalization are used for non-uniform illumination compensation and enhancement. We develop a novel segmentation technique to detect age-related macular degeneration in retina images by extracting the useful information without being affected by the presence of other structures.

Both, the enhancement and thresholding operators for segmentation are adaptive and based on histogram analysis.

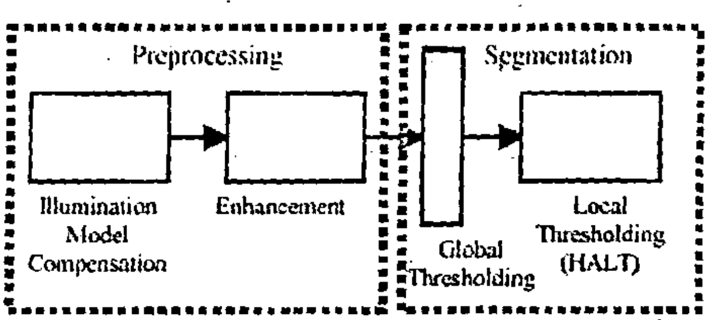

Figure 2.5 Technique to detect age-related macular degeneration in retina

Non-uniform illumination compensation is achieved by a commonly used technique, namely the homomorphic filtering. The illumination-reflectance model is also used as the basis for this frequency domain procedure that is useful for improving the appearance of an image by simultaneous brightness-range compression and contrast enhancement.

In order to avoid the selection of parameters for each retina image we process, a technique developed is based on sequential applications of histogram equalization. In fact, it is a multilevel (hierarchical) scheme that progresses from the entire image to smaller regions defined via windows.

The focus is on histogram-based techniques and threshold selection technique is also considered. This selection scheme is applied on non-overlapping local regions and also fits well with the region-adaptive approach developed for enhancement.

Histogram- based segmentation technique (HALT) to segment areas that differ slightly from their background regions is suggested that also performs segmentation of age-related macular degeneration.

2.3.2 Pre-enhancing Non-uniformly Illuminated Fingerprint Images

To improve the overall end-to-end performance from images obtained with fingerprint images with non-uniform inking or light, Norman Haas, Nalini k. Ratha and Ruud M. Bolle [8] implemented an image pre-enhancer, which attempts to compensate for the non-uniform illumination characteristics of the scanner.
Our algorithm contains some knowledge of the scanner design's geometry, but also attempts to adapt to effects caused by individual unit variations, in the presence of sampling noise.

Two desirable characteristics of such a setup are that the imaging plane be illuminated uniformly, and that the camera responds identically to areas of equal brightness, regardless of their position in the imaging plane. These goals could easily be achieved if the illumination source and the camera and were far from the imaging plane.

Figure 2.6 a) Actual Image b) Enhanced Image

But typically with such sensors, compactness of design is at a premium. A shallow design is sought, where, the length and width are constrained to be at least those of the finger area of interest, the depth is desired to be as close to zero as possible. There is pressure to employ such strategies as folding the optical path, and placing the camera and illumination to as close to the imaging plane as possible, typically compromising imaging uniformity.

Even for the same sensor, samplings of different fingers will produce differing plots, due to variation in the print pattern, pressure against the sensor surface, etc, so in making this plot, Norman Haas, Nalini k. Ratha and Ruud M. Bolle are measuring both the sensor itself and the data it is in turn supposed to be measuring[8].

Actually the error measurement in figure 2.6 is measured twice, once against the model curve shown, and once against the same curve flipped left-to-right; this gives us invariance to the in-use application of the sensor- whether fingers are being placed on it with the tips towards the constant-darkness end or the row-varying-darkness end. Whichever model has the lowest error, summed over all rows, is chosen.

When the imaging surface is inclined, and lighting is non-uniform on the finger, the images produced have a gray scale gradient that needs to be compensated for. We proposed a simple and elegant image pre-enhancement algorithm to improve the end-to-end performance of the system.

2.3.3 Nonlinear Filtering of Multiplied and Convolved Signals

Oppenheim, Schafer and Stockham [7] estimated the shading effect caused by non-uniform illumination in the image acquired through dusty camera.

They have tackled the shading by developing a model of shading in which they have decomposed the image into reflectance part, absorption part and fluorescent material. They took the logarithm concentration of fluorescent material to produce two terms, one of which is low frequency and the other is high frequency. They suppress the shading by high pass (homomorphic) filtering the logarithm of fluorescent material and then taking exponent to restore the image. They have used both background subtraction and shading correctness to eliminate the non-uniform illumination. [7]

2.3.4 On Chromatic and Geometrical Calibration

For the monochromatic issues of the calibration [4] presents the acquisition of monochrome images, the classic monochrome aberrations and the various sources of non-uniformity of the illumination of the image plane. Only the image deforming aberrations and the non-uniformity of illumination are included in the calibration models.

Some issues of the uneven intensities are (almost) common for all wavelengths and are considered with both colour and monochrome images. These therefore topics that also are interesting for the use of monochrome cameras. The uneven intensities that are wavelength dependent also causes problems with monochrome cameras but they are almost impossible to correct unless a significant amount of knowledge of the scene, light sources and objects is present.

When using a lens with a diaphragm between two lens groups, it can be optimized for no Light Falloff but the condition for this optimization is in contradiction with the Abbe sine condition and some aberrations must be accepted if the Light Falloff should be reduced.

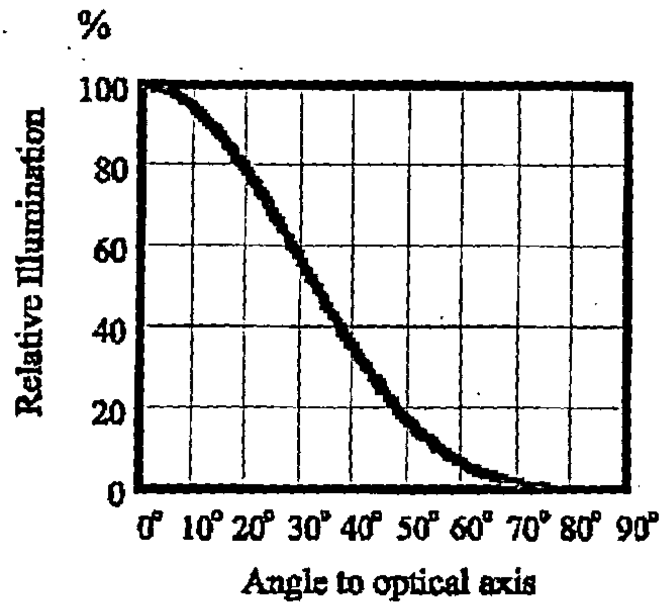

Figure 2.7 Non-uniformity of illumination caused by the cos4law.

Another method is to use a filter in the lens that spatially reduces the light accordingly to the 4th power of cosine. In an experiment where the chip and viewport are of fixed sizes, the Light Falloff can be reduced by using a lens with higher focal length and moving the lens/camera further away from the object. Using a 1/3" CCD chip and a wanted viewport in object space of 100mm x 75mm, the result is a reduction of the Light Falloff effect from 6.9% to 2.9% when using a 25 mm lens instead of a 16 mm. [4]

2.3.5 Cursive Character Recognition – A Character Segmentation Method Using Projection Profile-Based Technique

Roberto J. Rodrigues and Antonio Carlos Gay Thome [9] report the results of a study on a first sight decision tree algorithm for cursive script recognition based on the use of histogram as a projection profile technique. A postal code image data scanned is converted in a 2-dimension matrix representation to be used with a set of algorithms to provide full range segmentation.

Script recognition is classified as follows:

1. **Fixed-font character recognition**: It refers to the recognition of typewritten characters like pica, courier and so on.
2. **On-line recognition**: It is the method of hand-written character recognition where both the character image and the timing information of each trace are taken into account.
3. **Hand-written character recognition**: It refers to the recognition of typed hand-written characters.
4. **Script recognition**: It refers to those unrestricted handwritten characters that are cursive and may be connected.

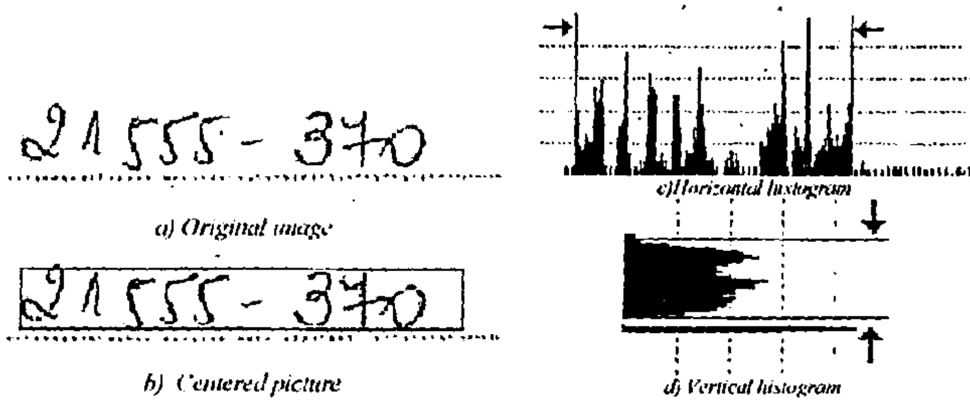

Figure 2.8 Projections of Centered Image

One of the most important and complex tasks in this process of individual character recognition is the segmentation. The method, based on a decision tree construction and on the use of projection profile histograms, has been investigated and proposed.

The use of projection histograms given in figure 2.8, for the process of character segmentation does not solve all presented problems, as trace slant for instance, however it does solve more than 70% of the cases with connected digits. Once the used algorithms has as base a very simple implementation, including the conventional filters, the process, in the whole, provide an excellent functional performance, taking the time necessary to carry through all the stages of the process.

2.3.6 Character Feature Extraction Using Polygonal Projection Sweep (Contour Detection)

Roberto J. Rodrigues, Gizelle K. Vianna, Antonio C. G. Thomé [10] present an approach based on the edge detection, where a set of feature vectors is taken from the source image. The images under this investigation are considered to be manuscript characters and the features are obtained by the distance from the contour of each character to several observation points placed around the image. Such observation points are arranged along different geometric polygons built in a way to surround the image. The approach is evaluated against the naïve bitmap matrix considering different types of polygon.

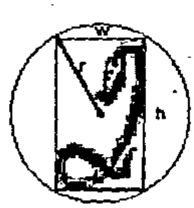

Figure 2.9 Graphical representation of method

This method can be separated in 3 stages: image quality compensation, initial segmentation and successive refinements. The image quality compensation step is used to improve the quality of the scanned image, reducing or enhancing details as noises and contrasts. Initial segmentation is the step where the image is initially segmented and it is based on the projection histogram. The algorithm is fast and it is able to segment all those disconnected characters. Dots and lines can be removed using the data retrieved from the histogram horizontal vector. The last product generated by the initial segmentation is the identification of those possible connected characters.

Using the power of reconstruction of the original image as a visual and first benchmark, it was observed that odd sided polygons were not as good as the even ones. Investigation is limited to two versions: one based on the radius and other on the diameter. As showed by the figures, the input vector is coded based on predefined dimensions and the features are extracted as the distances counted in number of pixels from each side of the polygon to the contour of the image. [10]

2.3.7 Vector Templates and Handprinted Digit Recognition

J. R. Parker [11] suggests the use of scalable vector templates, which can be used to generate a template with the same scale and line width attributes as an arbitrary input character image. The best match is the template having the smallest total distance between black pixels. Multiple templates are used for each character, and digits only are used as a sample data set.

3, 0 - 0, 3
0, 3 - 0, 6
0, 6 - 3, 9
3, 9 - 9, 0
9, 0 - 9, 9

(a)

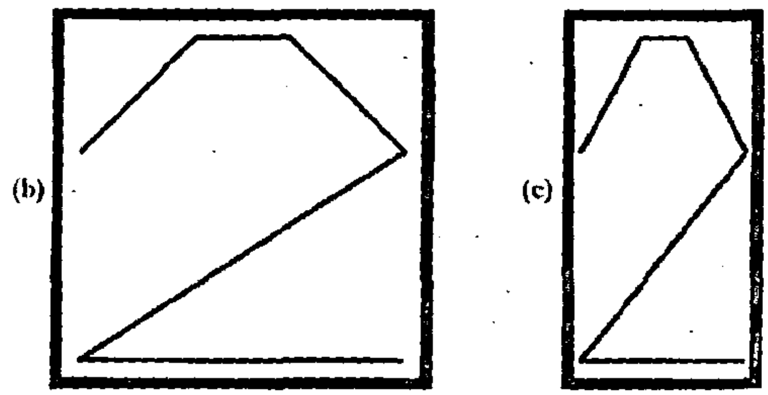

Figure 2.10 An Example Vector Template (a) Coordinates of vector end points for '2' (b) Vectors drawn on 10x10 grid (c) Vectors drawn on 20x10 grid

It is also possible to create a template from a data image, and may be desirable when starting to process data from a new source. The first step in this process is to threshold and then performs thinning on the input image.

A set of vectors is extracted from each curve using a recursive splitting technique, a relatively simple and common method for vectorizing small, simple images. Briefly, the endpoints of the curve are presumed initially to be the start and end points of a line, and the distances between all pixels in the curve and the mathematical line are computed.
Once all of the templates have been generated (there are multiple templates for each digit) the system is ready to recognize digits.

An incoming image is first pre-processed in any desired fashion and is then thresholded. The width of the lines in the image is then estimated using horizontal and vertical scans. A histogram containing the widths of the black portions of the image on all slices is

CHAPTER 3

REQUIREMENTS ANALYSIS

3. Requirements Analysis

The requirement analysis is the first step towards developing software. Analysis must be performed in a systematic and correct manner so as to have as few mistakes as possible in the software and to have an end product completely fulfilling the expectations of the client. The reliability and the robustness of the software are highly dependent on the fact that the analysis is carried out properly. The main objective of this phase is to identify all possible requirements and expectations kept of software. In it problems are identified and then a possible solution is proposed.

3.1 Problem Analysis

The report reveals the functional requirements of the system as under:

- Application of high order Gaussian low pass filter to estimate the non-uniform illumination.
- Image enhancement in terms of removal of non-uniform illumination from a digital image by using illumination and reflectance model.
- Adding a constant illumination to the image obtained from previous step and stretches the contrast.
- For the recognition phase to continue work, selection of digit boundary is carried out. After that Minimum Enclosing rectangle is drawn to get the compact shape of digit.
- Profiles are to be obtained starting from left edge of the rectangle tightly covering the digit till the boundary of the actual character starts.
- Application of smoothing is helpful in characterizing the projection shape to ignore unimportant variations.
- Features are then extracted based on the depth profiles looking for variations in crest and trough obtained from it.
- Comparing the statistical features obtained for each digit shape does matching of the input charater and the closest match is the successful key.

3.2 Use Case Analysis

Analysis of the project is presented in terms of use case diagrams indicating the actors and use case in expanded format. This helps visualizing the work and indicating the system boundaries while presenting the functionalities. The Use Case Model describes the proposed functionality of the new system.

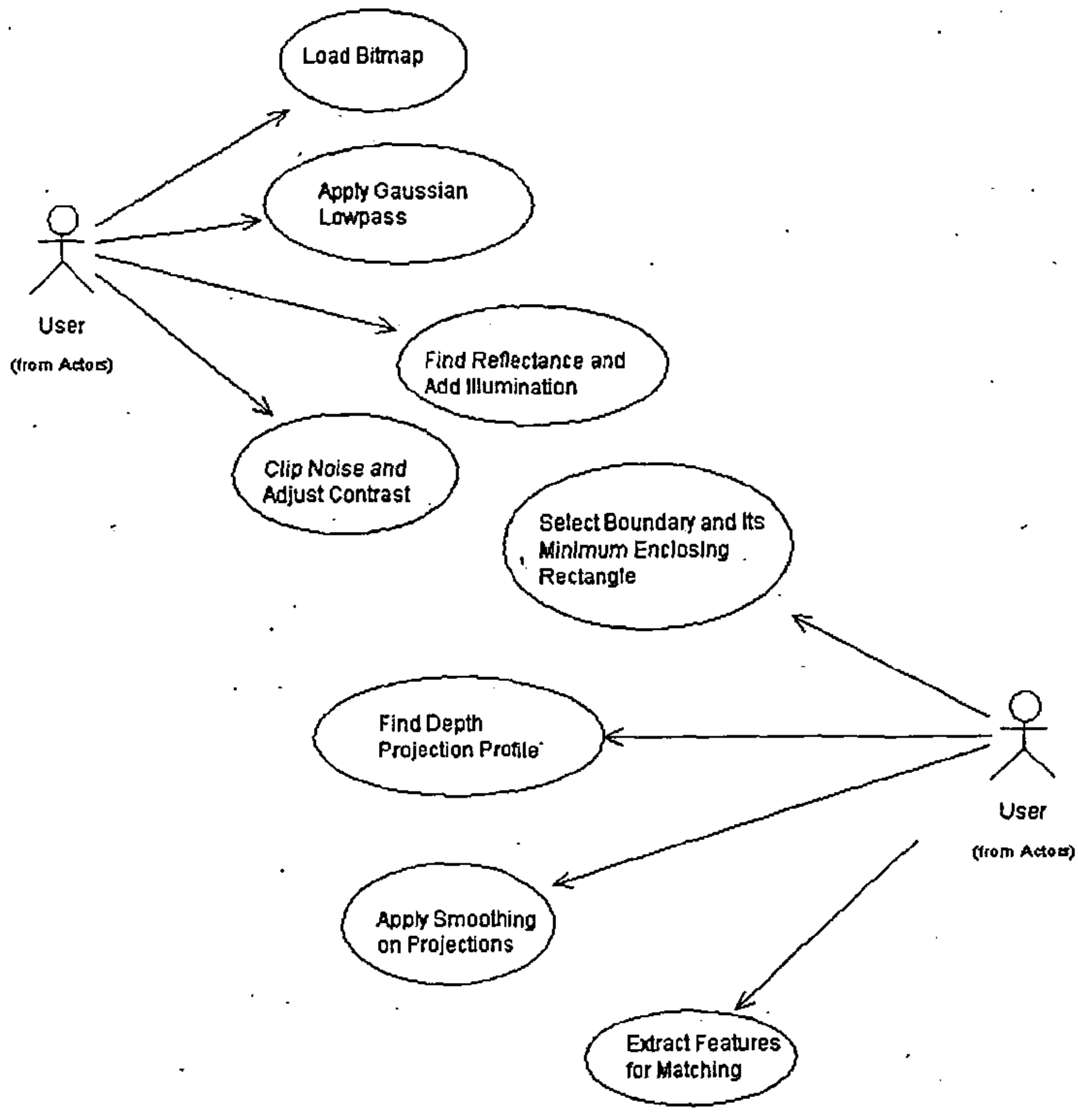

Figure 3.1 Use case Diagram of Handwritten Roll Number Recognition Followed by Non-Uniform Illumination Removal

3. Requirements Analysis

The requirement analysis is the first step towards developing software. Analysis must be performed in a systematic and correct manner so as to have as few mistakes as possible in the software and to have an end product completely fulfilling the expectations of the client. The reliability and the robustness of the software are highly dependent on the fact that the analysis is carried out properly. The main objective of this phase is to identify all possible requirements and expectations kept of software. In it problems are identified and then a possible solution is proposed.

3.1 Problem Analysis

The report reveals the functional requirements of the system as under:

- Application of high order Gaussian low pass filter to estimate the non-uniform illumination.
- Image enhancement in terms of removal of non-uniform illumination from a digital image by using illumination and reflectance model.
- Adding a constant illumination to the image obtained from previous step and stretches the contrast.
- For the recognition phase to continue work, selection of digit boundary is carried out. After that Minimum Enclosing rectangle is drawn to get the compact shape of digit.
- Profiles are to be obtained starting from left edge of the rectangle tightly covering the digit till the boundary of the actual character starts.
- Application of smoothing is helpful in characterizing the projection shape to ignore unimportant variations.
- Features are then extracted based on the depth profiles looking for variations in crest and trough obtained from it.
- Comparing the statistical features obtained for each digit shape does matching of the input charater and the closest match is the successful key.

Use case depicts a set of scenarios that describing an interaction between a user and a system. Use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors.

3.2.1 Use Case in Expanded Format

For each module of the project several use cases are identified and the description of each use case is as follows:

3.2.1.1 Load Bitmap Use Case

a) Name: Load Bitmap

b) Actor: User

c) Pre-Condition: None

d) Post Condition: Image gets loaded in memory.

e) Typical Course of Action:

Actor Action	System Response
1. User presses image load button.	2. System loads image in memory. 3. System displays image on screen.

f) Alternate Course of Action:

Actor Action	System Response
1a. Image is not loaded. 3a. Repeat step 1 to 3	2a. Display error message.

3.2.1.2 Apply Gaussian Filter Use Case

a) Name: Apply Gaussian Filter

b) Actor: User

c) Pre-Condition: Image is loaded and filter values are preset in convolution kernel.

d) Post Condition: Blurred image is obtained after filter application

e) Typical Course of Action:

Actor Action	System Response
1. User presses remove imbalanced illumination button.	2. An image with changed spatial values is obtained. 3. Actual size as that of original image is selected. 4. New storage space is allocated.

f) Alternate Course of Action:

Actor Action	System Response
None	

3.2.1.3 Find Reflectance and Add Illumination Use Case

a) Name: Find Reflectance and Add Illumination

b) Actor: User

c) Pre-Condition: Low pass of image should be available in memory.

d) Post Condition: Estimated reflectance part is obtained with uniform illumination.

e) Typical Course of Action:

Actor Action	System Response
	1. Low pass of image is used to find reflectance part of image. 2. Using reflectance of image it is multiplied with constant illumination.

f) Alternate Course of Action:

Actor Action	System Response
	1a. Division by zero. 2b. Ignore zero value and replace older value at that location.

3.2.1.4 Clip Noise and Adjust Contrast Use Case

a) Name: Clip Noise and Adjust Contrast

b) Actor: User

c) Pre-Condition: Uniformly illuminated image is present.

d) Post Condition: An enhanced image to be generated containing digits to be recognized

e) Typical Course of Action:

Actor Action	System Response
1. User gives lower and upper bounds for clipping.	2. Normalize pixels by clipping to eliminate noisy pixels introduced in the previous step. 3. Adjust the contrast to use full range of values.

f) Alternate Course of Action:

Actor Action	System Response
	3a. Image contrast is not enhanced. Repeat step 1.

3.2.1.5 Select Boundary and Its Minimum Enclosing Rectangle Use Case

a) Name: Select Boundary and Its Minimum Enclosing Rectangle

b) Actor: User

c) Pre-Condition: An enhanced image as input is available.

d) Post Condition: Digit is inscribed in a tight rectangle.

e) Typical Course of Action:

Actor Action	System Response
1. User presses select area button. 2. User clicks at top left corner of digit.	3. Red mark indicates that point selected as top left corner.
4. User clicks at bottom right side of digit.	5. Red mark indicates that point selected as top left corner.

	rectangle is reached. 8. Starting from top right find projection profile in column until the boundary of character is reached at bottom. 9. Repeat 8 for next line till last column in rectangle is reached.

f) Alternate Course of Action:

Actor Action	System Response
None	

3.2.1.7 Apply Smoothing on Projections Use Case

a) Name: Apply Smoothing on Projections

b) Actor: User

c) Pre-Condition: Depth projections of a digit along each side are available.

d) Post Condition: None.

e) Typical Course of Action:

Actor Action	System Response
	2. Take average of projection before and after the current to avoid smaller changes to be counted as crest or trough. 3. Apply step 2 for the projections in each side.

f) Alternate Course of Action:

Actor Action	System Response
None	

3.2.1.8 Extract Features for Matching Use Case

a) Name: Extract Features for Matching

b) Actor: User

c) Pre-Condition: Depth projections of a digit along each side are available.

d) Post Condition: None.

e) Typical Course of Action:

Actor Action	System Response
	1. Find the crest to trough or trough to crest transition. 2. For any transition assign binary pattern. 3. Repeat 1 and 2 for right, top and bottom projections. 4. Store features in a vector. 5. Read feature file to find the closer match from an existing stored features of each digit. 6. Display digit with the closer match.

f) Alternate Course of Action:

Actor Action	System Response
	5a. File not found. 5b. Open locate feature file dialogue to locate it. Perform step 5.

3.2Conceptual Model

Conceptual objects have been described those are found in the domain of the system. This helps in further understand the working of the system that is to be designed.

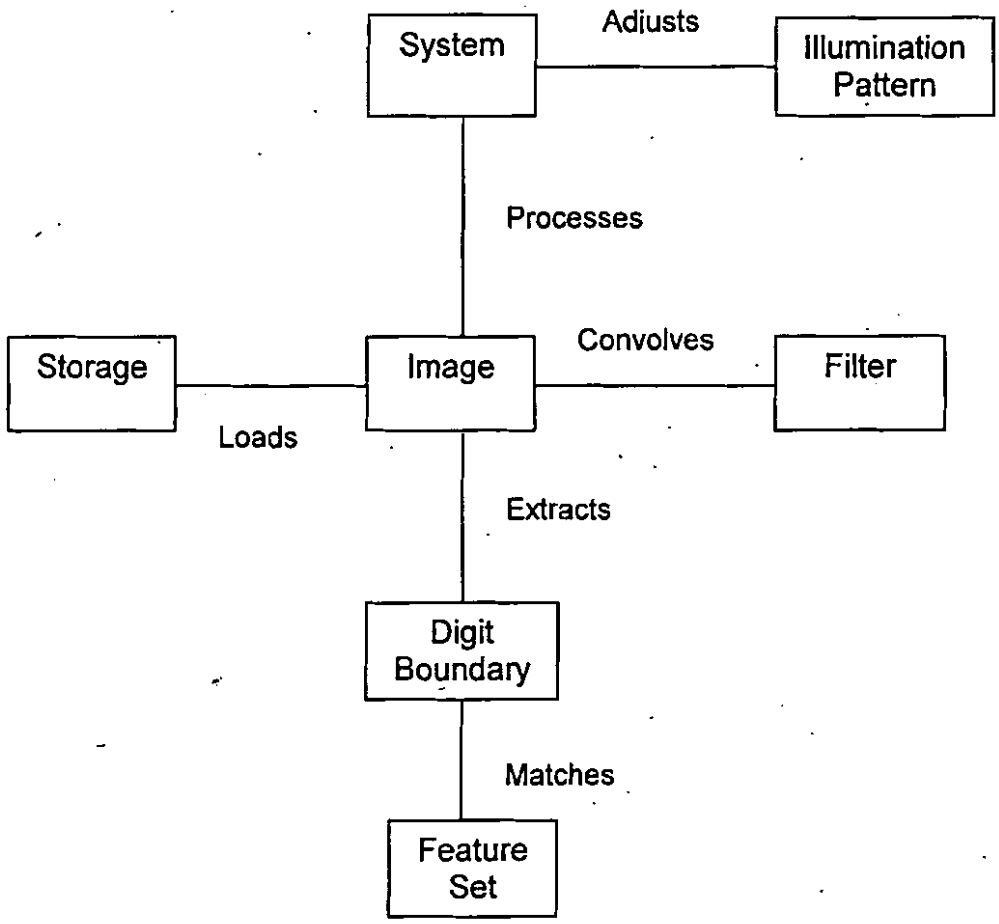

Figure 3.2 Conceptual Model of Handwritten Roll Number Recognition Followed by Non-Uniform Illumination Removal

CHAPTER 4

DESIGN

4. System Design

System design is the specification or construction of a technical, computer-based solution for the business requirements identified in the system analysis. It is the evaluation of alternative solutions and the specification of a detailed computer-based solution. The design phase is the first step towards moving from problem domain to the solution domain. System design develops the architectural detail required to build a system or product.

4.1 Object-Oriented Design Method

Object-Oriented design translates the Object Oriented Analysis (OOA) model of the real world into an implementation-specific model that can be realized in software. Object-oriented design transforms the analysis model, created using object-oriented analysis method, into a design model that serves as a blueprint for software construction. For the development of the system under consideration the same technique is used.
Object-oriented design (OOD) is concerned with developing an object-oriented model of a software system to implement the identified requirements.
Object Oriented Design builds on the products developed during Object-Oriented Analysis (OOA) by refining candidate objects into classes, defining message protocols for all objects, defining data structures and procedures, and mapping these into an object-oriented programming language (OOPL).

4.1.1 Sequence Diagrams

Once the use cases are specified, and some of the core objects in the system are prototyped, we can start designing the dynamic behavior of the system. Sequence diagrams demonstrate the behavior of objects in a use case by describing the objects and the messages they pass. Sequence diagrams emphasize the order in which things happen

TH-4473

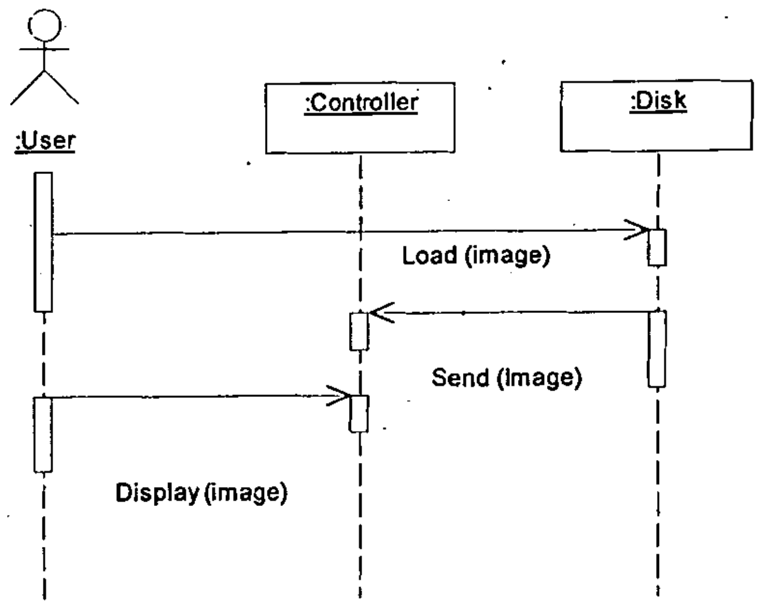

Figure 4.1 Sequence Diagram of Load Bitmap

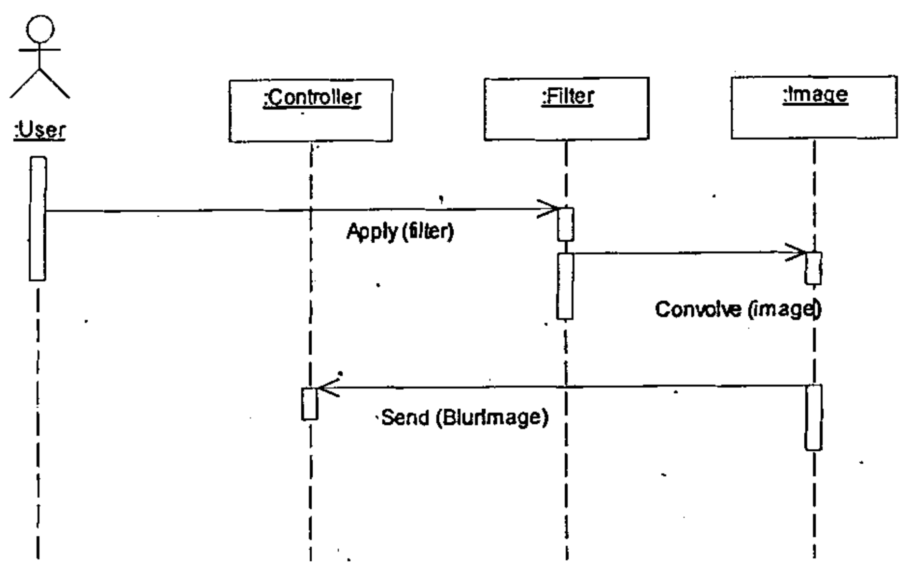

Figure 4.2 Sequence Diagram of Apply Gaussian Filter

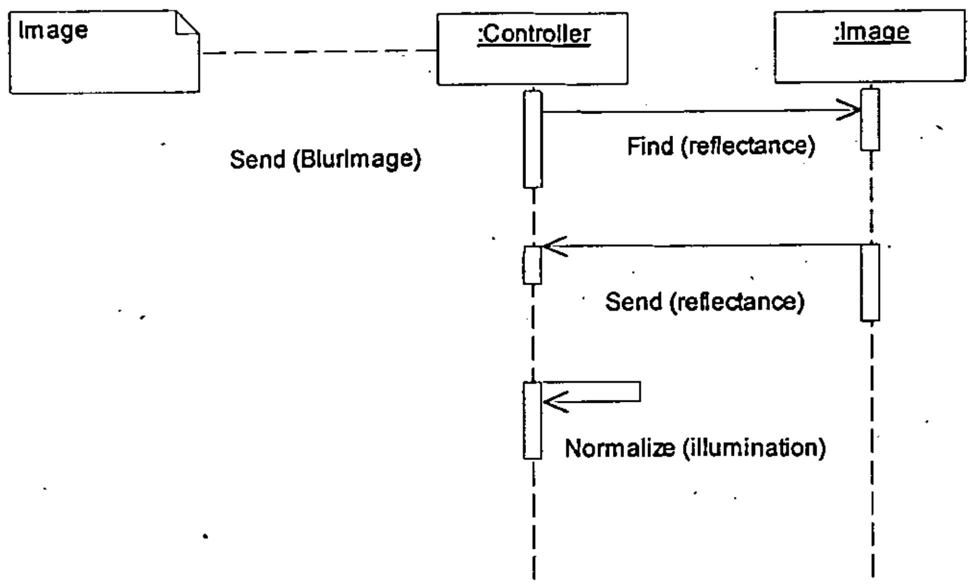

Figure 4.3 Sequence Diagram of Find Reflectance and Add Illumination

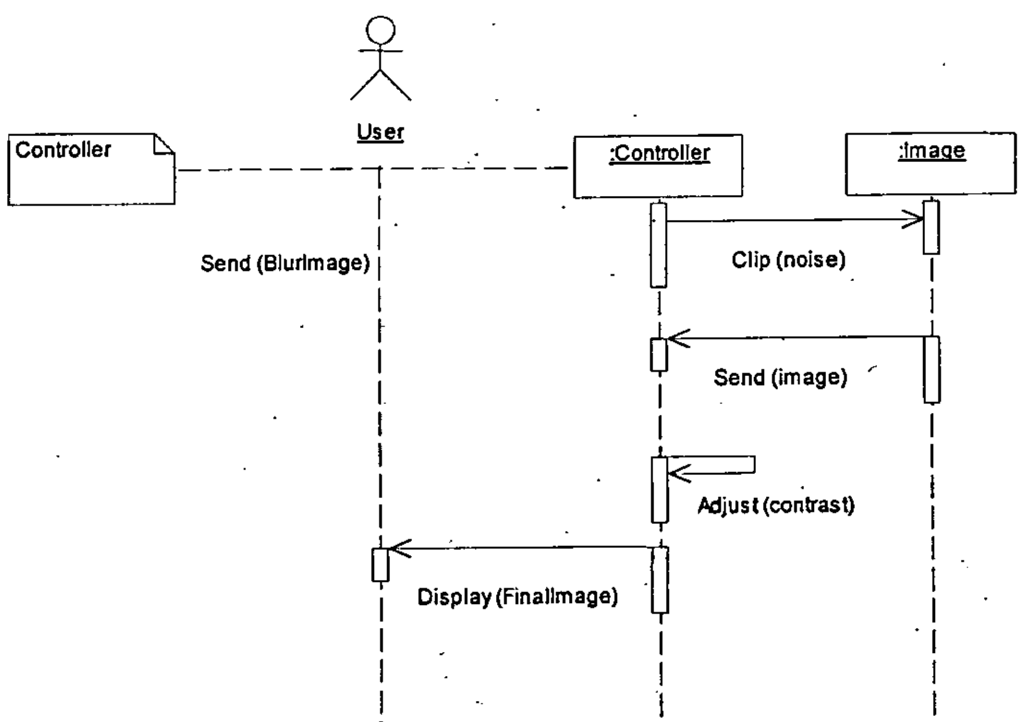

Figure 4.4 Sequence Diagram of Clip Noise and Adjust Contrast

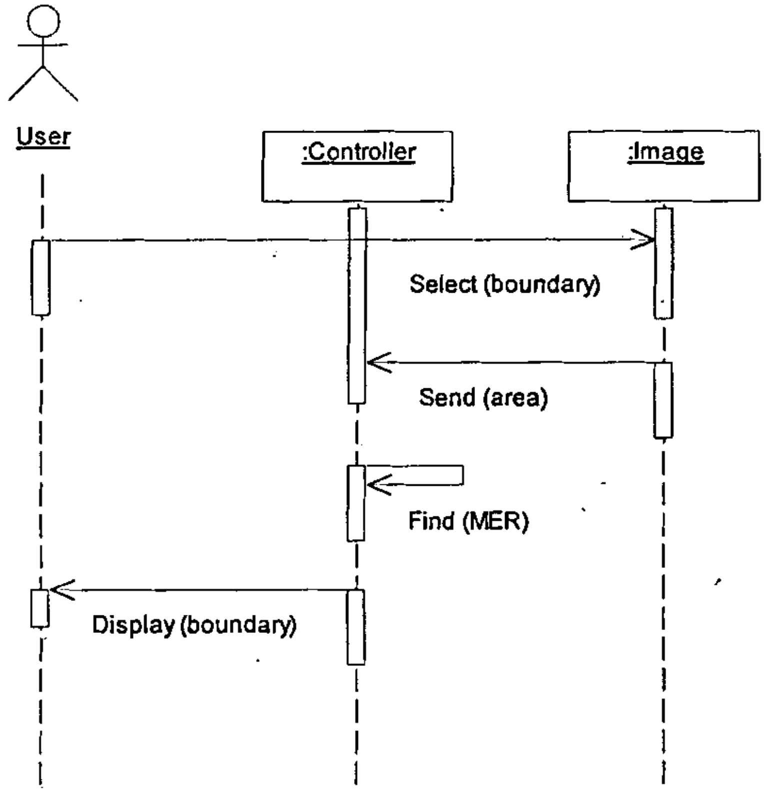

Figure 4.5 Sequence Diagram of Select Boundary and Its Minimum Enclosing Rectangle

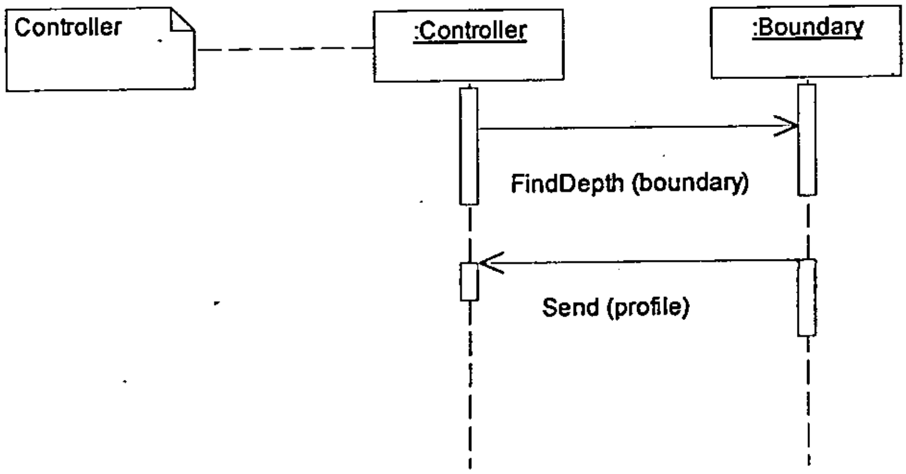

Figure 4.6 Sequence Diagram of Find Depth Projection Profile

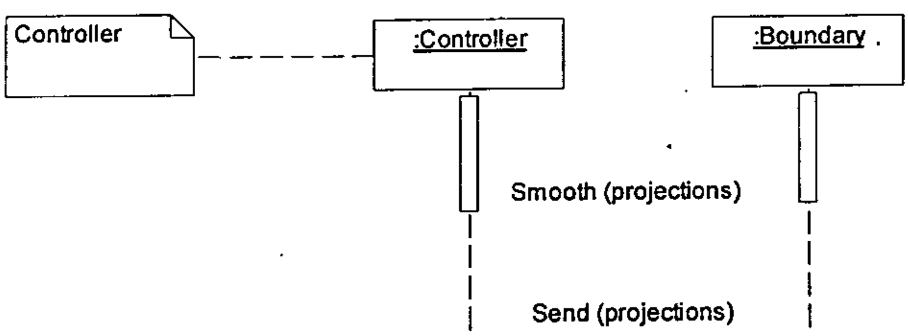

Figure 4.7 Sequence Diagram of Apply Smoothing on Projections

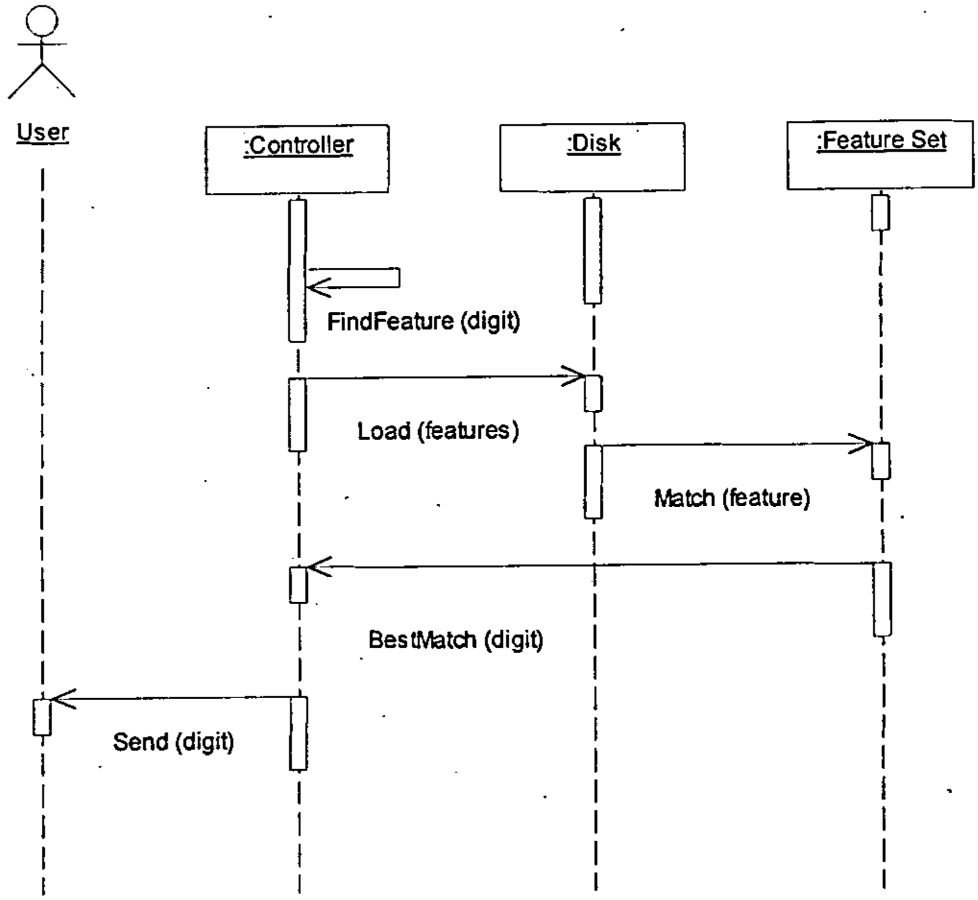

Figure 4.8 Sequence Diagram of Extract Features for Matching

4.1.2 Class Diagrams

Class diagrams are the backbone of almost every object-oriented method including UML. They describe the static structure of a system. It can also be said that class diagrams identify the class structure of a system, including the properties and methods of each class. Also depicted are the various relationships that can exist between classes, such as an inheritance relationship. The Class diagram is one of the most widely used diagrams from the *UML specification*.
Another purpose of class diagrams is to specify the class relationships and the attributes and behaviors associated with each class. Class diagrams are remarkable at illustrating *inheritance and composite* relationships. A class diagram consists of one major component and that is the various classes, along with these are the various relationships shown between the classes such as aggregation, association, composition, dependency, and generalization.

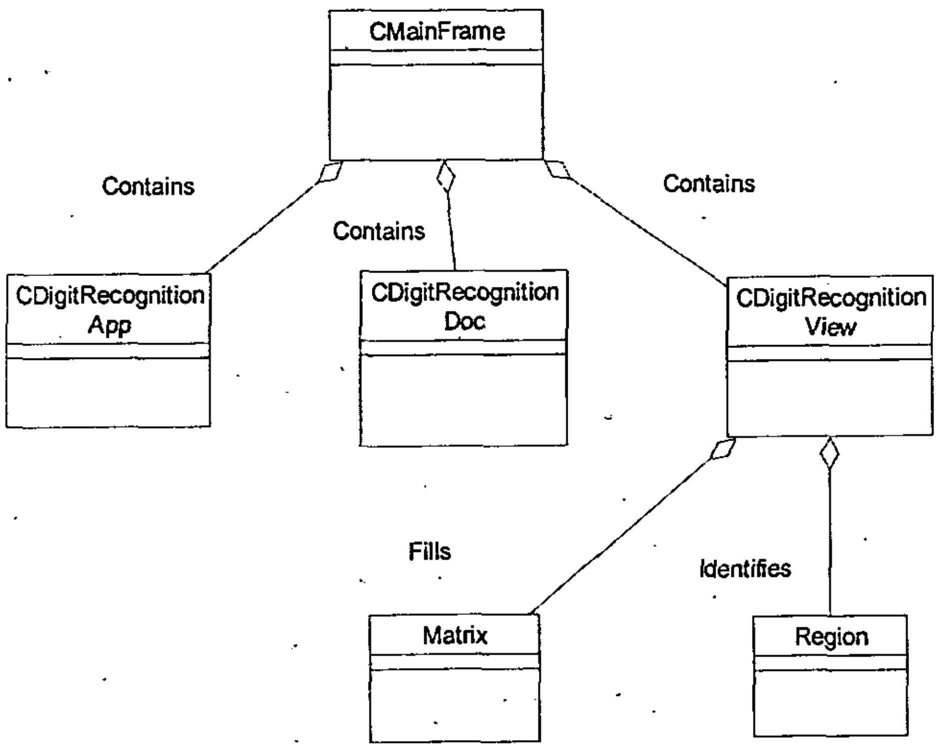

Figure 4.9 Class Diagram of Handwritten Roll Number Recognition Followed by Non-Uniform Illumination Removal

4.3.2.1 Classes Identified

Following classes are identified that contain the member data and the associated functions that operate on image to produce a processed image or to extract and match the features.

CMainFrame
m_wndStatusBar
AssertValid() CMainFrame() ~CMainFrame() Dump() OnCreate() PreCreateWindow()

Class 4.1 CMainFrame

CDigitRecognitionApp
CDigitRecognitionApp() InitInstance() OnAppAbout()

Class 4.2 CDigitRecognitionApp

CDigitRecognitionDoc
CBitmap m_Bmp BITMAP b
AssertValid() CDigitRecognitionDoc() ~CDigitRecognitionDoc() Dump() OnFileOpen() Serialize()

Class 4.3 CDigitRecognitionDoc

All the above mentioned classes provide the basic skeleton for an application developed for finding the features and thus the recognition of digits.

The following class contains the core processing functions that operate to find the features once the exact area of digit is specified by the user. Internal detail of these methods is provided in the next chapter.

CDigitRecognitionView

LeftFeatureVector
RightFeatureVector
TopFeatureVector
BottomFeatureVector
RegionPixCount, RegionMinX, RegionMaxX, RegionMinY, RegionMaxY
Width, Hight
Threshold
m_OrgBmp,m_ResBmp
m_OrgDC,m_ResDC

AssertValid()
CDigitRecognitionView()
~CDigitRecognitionView()
Dump()
GetBottomProjections()
GetDocument()
GetImagePart()
GetLeftProjections()
GetRegionExtents()
GetRightProjections()
GetTopProjections()
OnDraw()
OnLButtonDown()
OnRecognitionRecognize()
OnRecognitionSelectarea()
OnViewBilevelimage()
OnViewColorimage()
OnViewGrayimage()
PreCreateWindow()
PutInVector()
QuantizeProjections()
SameAsPrevInVector()
SmoothVector()
OnApplyFilter()
OnDraw()
OnLoadBitmap()
Clip()
Normalize()

Class 4.4 CDigitRecognitionView

4.1.3 State Transition Diagram

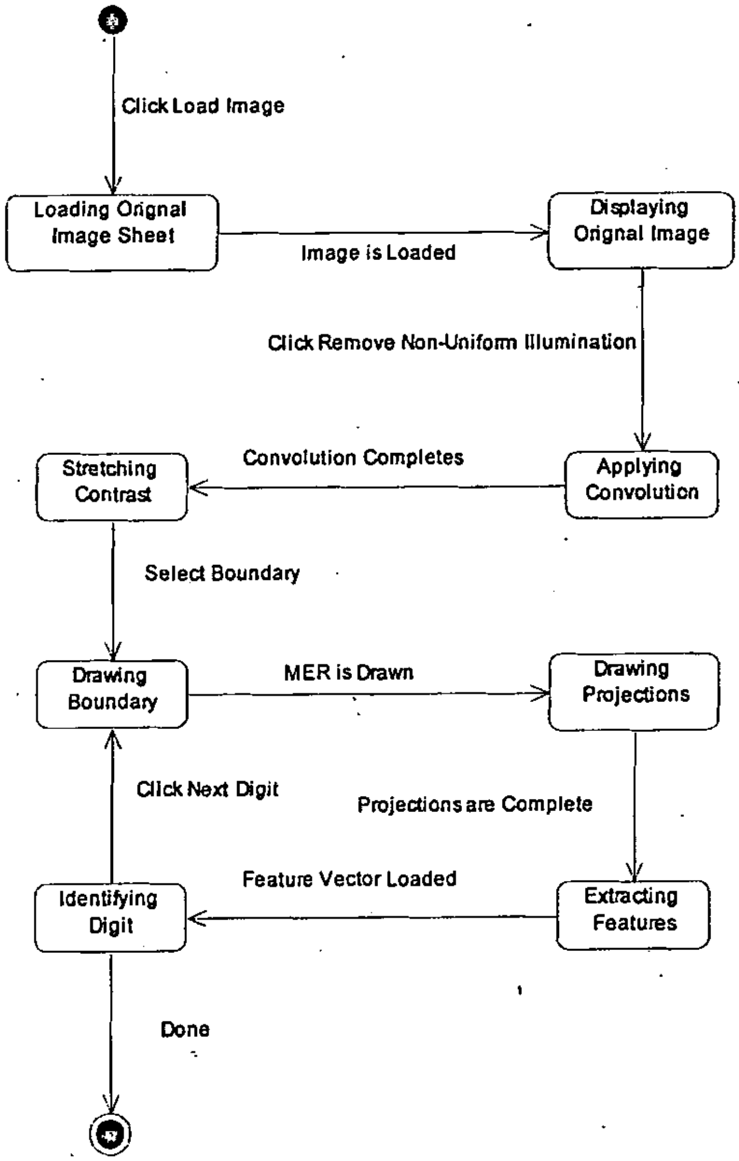

Figure 4.10 State Transition Diagram of Handwritten Roll Number Recognition Followed by Non-Uniform Illumination Removal System

CHAPTER 5

IMPLEMENTATION

5. Implementation

First module for the design of Gaussian filter and removal of non-uniform illuminationis coded in Matlab 6.0 where as the other modules performing the successive operations are developed in Visual C++ 6.0.

5.1 Gaussian filter design

A normalized Gaussian filter is generated as a 21 by 21 elements two-dimensional array that is used as convolution mask to operate on the entire image to estimate the illumination pattern of the image.

```
% Gaussian Spatial Domain Lowpass

r = 5;
sum=0;
gfil = zeros (21,21);
for m = 1:21,
   for n = 1:21,
      D = sqrt ( (m - 11)^2 + (n - 11)^2 );
      E = (D^2) / (2*(r^2));
      gfil(m, n) = 1 / exp (E);
      sum = sum + gfil(m,n);
   end
end

gfil=gfil./sum;
```

5.2 Finding the reflectance part and Addition of Uniform Illumination

Reflectance part of the image is obtained once the illumination pattern of the digital image is available. This reflection part is then multiplied with a constant illumination factor.

```
% Findout the reflectance part of image
x1 = xx(11:551,11:375);
out1 = out(11:551,11:375);
ref=x1 ./ out1;

maxv=max(max(ref));
const=255 ./ maxv;
ref2=ref .* const;

restore = clip (ref2,0,255);
ref3 = restore;
```

5.3 Gray View of Image

This function converts the image into grayish shade. In this function gets the current RGB value and multiply them by some specific float value to convert them into RGB which makes the grayish shade.

```
void CDigitRecognitionView::OnViewGrayimage()
{
	//Works on data loaded in Document class
	CDC memDC;
	CClientDC dc(this);
	unsigned char Gray;
	CDigitRecognitionDoc* pDoc = GetDocument();

	memDC.CreateCompatibleDC(&dc);
	memDC.SelectObject(&pDoc->m_Bmp);

	for(int j=0;j<=pDoc->b.bmHeight-1;j++)
		for(int i=0;i<pDoc->b.bmWidth;i++)
		{
			Gray=0.299*GetRValue(memDC.GetPixel(i,j))+
				0.587*GetGValue(memDC.GetPixel(i,j))+
				0.114*GetBValue(memDC.GetPixel(i,j));
			dc.SetPixel(i,j, RGB(Gray,Gray,Gray));
		}

	memDC.DeleteDC();
}
```

5.4 Bilevel View of Image

This function converts the image into black and white shade. In this function gets the current RGB value and multiply them by some specific float value to convert them into RGB which makes the black and white shade.

```
void CDigitRecognitionView::OnViewBilevelimage()
{
	CDC memDC;
	CClientDC dc(this);
	unsigned char Gray, Bi;
	CDigitRecognitionDoc* pDoc = GetDocument();

	memDC.CreateCompatibleDC(&dc);
	memDC.SelectObject(&pDoc->m_Bmp);


for(int j=0;j<=pDoc->b.bmHeight-1;j++)
```

```
            for(int i=0;i<pDoc->b.bmWidth;i++)
            {

        Gray=0.299*GetRValue(memDC.GetPixel(i,j))+0.587*GetGValue(memDC.GetP
ixel(i,j))+0.114*GetBValue(memDC.GetPixel(i,j));
                if (Gray<Threshold)
                        Bi= 0;
                else
                        Bi = 255;
                dc.SetPixel(i,j, RGB(Bi,Bi,Bi));
            }

        memDC.DeleteDC();

}
```

5.5 Get the Image Part of Digit

In this Function we get the image part of the selected digit on mouse click. We get the RGB value and convert it into grayish value by following the same procedure as in above function. Then we compare this gray value with some specific threshold value and if this value is less than threshold then we get that this pixel is not a part of digit and when its value is greater than it is a part of digit.

```
void CDigitRecognitionView::GetImagePart()
{
        CDC memDC;
        CClientDC dc(this);
        unsigned char Gray;

        CDigitRecognitionDoc* pDoc = GetDocument();

        memDC.CreateCompatibleDC(&dc);
        memDC.SelectObject(&pDoc->m_Bmp);
        //Initializing Image Part
        for(int j=0;j<250;j++)
                for(int i=0;i<250;i++)
                        ImagePart[i][j] = 255;

        //Getting the selected image part into memory
        int x=0,y=0;
        for(j=LtCorner.y;j<RtCorner.y;j++)
        {
                for(int i=LtCorner.x;i<RtCorner.x;i++)
                {
```

```
        Gray=0.299*GetRValue(memDC.GetPixel(i,j))+0.587*GetGValue(memDC.GetP
ixel(i,j))+0.114*GetBValue(memDC.GetPixel(i,j));
                        if (Gray<Threshold)
                                ImagePart[x][y]= 0;
                        else
                                ImagePart[x][y] = 255;
                        x++;
                }
                y++;
                x=0;
        }
        //Putting selected part on blanked rectangle by OnLButtonDown
        x=1;y=1;
        for(j=LtCorner.y+1;j<RtCorner.y-1;j++)
        {
                for(int i=LtCorner.x+1;i<RtCorner.x-1;i++)
                {
                        dc.SetPixel(i, j, RGB(ImagePart[x][y], ImagePart[x][y],
ImagePart[x][y]));
                        x++;
                }
                y++;
                x=1;
        }
        Hight=abs(RtCorner.y-LtCorner.y)+1;
        Width=abs(RtCorner.x-LtCorner.x)+1;

        memDC.DeleteDC();
}
```

5.6 Get Left Projection

Through the Image Part of digit Left projection of the digit is found. In the left projection we find vectors on the left hand side of the digit. These vectors are calculated from Image Part.

```
void CDigitRecognitionView::GetLeftProjections()
{
        int x, y;
        CClientDC dc(this);

        for(y=RegionMinY; y<RegionMaxY; y++)
        {
                x=RegionMinX;
                while(x<RegionMaxX)
```

```
                {
                        dc.SetPixel(LtCorner.x+x, LtCorner.y+y, RGB(0,255,0));
                        if(ImagePart[x][y]==0)
                        {
                                LeftProjectionVector[y]=x;  //Record for distance
travelled in x for first black pixel
                                break;                              //for each line.
Stop when found
                        }
                        x++;

                }
        }
}
```

5.7 Get Right Projection

Through the Image Part of digit Right projection of the digit is found. In the right projection we find vectors on the right hand side of the digit. These vectors are calculated from Image Part.

```
void CDigitRecognitionView::GetRightProjections()
{
        int x, y;
        CClientDC dc(this);
        int Distance;
        for(y=RegionMinY; y<RegionMaxY; y++)
        {
                x=RegionMaxX;
                Distance=0;
                while(x>RegionMinX)
                {
                        dc.SetPixel(LtCorner.x+x, LtCorner.y+y, RGB(0,0,255));
                        if(ImagePart[x][y]==0)
                        {
                                RightProjectionVector[y]=Distance; //Record for distance
travelled in x for first black pixel
                                break;                              //for each line.
Stop when found
                        }
                        x--;
                        Distance++;
                }
        }
}
```

5.8 Get Top Projection

Through the Image Part of digit Top projection of the digit is found. In the top projection we find vectors on the top hand side of the digit. These vectors are calculated from Image Part.

```
void CDigitRecognitionView::GetTopProjections()
{
        int x, y;
        CClientDC dc(this);

        for(x=RegionMinX; x<RegionMaxX; x++)
        {
                y=RegionMinY;
                while(y<RegionMaxY)
                {
                        dc.SetPixel(LtCorner.x+x, LtCorner.y+y, RGB(255,0,0));
                        if(ImagePart[x][y]==0)
                        {
                                TopProjectionVector[x]=y;     //Record for distance
travelled in y for first black pixel
                                break;                                        //for each line.
Stop when found
                        }
                        y++;

                }
        }
}
```

5.9 Get Bottom Projection

Through the Image Part of digit Bottom projection of the digit is found. In the bottom projection we find vectors on the bottom hand side of the digit. These vectors are calculated from Image Part.

```
void CDigitRecognitionView::GetBottomProjections()
{
        int x, y;
        CClientDC dc(this);
        int Distance;
        for(x=RegionMinX; x<RegionMaxX; x++)
        {
                y=RegionMaxY;
                Distance=0;
                while(y>RegionMinY)
```

```
            {
                    dc.SetPixel(LtCorner.x+x, LtCorner.y+y, RGB(255,0,0));
                    if(ImagePart[x][y]==0)
                    {
                            BottomProjectionVector[x]=Distance;         //Record for
distance travelled in y for first black pixel
                            break;                                      //for each line.
Stop when found
                    }
                    y--;
                    Distance++;
            }
        }
}
```

5.10 Smoothing the Vectors

After getting Vectors on all sides of the digit first of we smooth them. In this function we smooth the Vectors by getting their averages. First of all we check that if middle vector is less than other two Vectors (Before and After that) then we set that middle vector equal to their average. After that We check that if middle vector is greater than other two Vectors (Before and After that) then we set that middle vector equal to their average.

```
void CDigitRecognitionView::SmoothVector(int Vector[], int VectorDirection)
{
        int i,Min, Max;

        if(VectorDirection==0) //Horizontal Projections
        {
                Min=RegionMinY;
                Max=RegionMaxY;
        }
        else //Vertical Projections
        {
                Min=RegionMinX;
                Max=RegionMaxX;
        }
        //Smoothing by averaging
        for(i=Min+1;i<Max-1;i++)
        {
                //Vector[i]=float(Vector[i-1]+Vector[i]+Vector[i+1])/3.0;
                if(Vector[i-1] > Vector[i] && Vector[i+1] > Vector[i] )
                        Vector[i]=double(Vector[i-1]+Vector[i]+Vector[i+1])/3.0;
                if(Vector[i-1] < Vector[i] && Vector[i+1] < Vector[i] )
                        Vector[i]=double(Vector[i-1]+Vector[i]+Vector[i+1])/3.0;
        }
}
```

5.11 Quantize the Projections

After Getting All the Projection Vectors on each side of the digit we quantize them. Quantize means that we draw the All the features of that digit on each side of the digit on the basis of these projections. For drawing the features we use the concept of peak and valley. Peak means from lower to higher and valley means from higher to lower. For example if we take the left hand side Projection vectors of digit, we see the length of each of the vectors from starting to end. If we found that vector length is decreasing, we conclude that it is valley and if it is increasing it is peak. In the case of peak we generate the feature 1 and in the case of valley we generate the feature 0. Hence we found All the features of the digit on each side of the digit in the form of 1 and 0.

```
  void CDigitRecognitionView::QuantizeProjections(int Vector[], int VectorID, int
VectorDirection)
{
        int Position;
        CClientDC dc(this);
        CPoint p;
        int Min, Max;
        int Dx, Dy;
        int i, Index;

        Index=0;

        if(VectorDirection==0) //Horizontal Projections
        {
                Min=RegionMinY;
                Max=RegionMaxY;
        }
        else //Vertical Projections
        {
                Min=RegionMinX;
                Max=RegionMaxX;
        }

        switch(VectorID)
        {
        case 1: Dx=500; Dy=250;break;
        case 2: Dx=750; Dy=250;break;
        case 3: Dx=500; Dy=500;break;
        case 4: Dx=750; Dy=500;
        }

i=Min;
bool eqflag=false;
int count=0;
while(i<Max-1)
```

```
{
        if(Vector[i] == Vector[i+1])
        {
                // May be Equal
                count=i;
                while(Vector[i] == Vector[i+1])
                        i++;
                count=i-count;
                if(count>4)
                        eqflag=true;
        }
        else if((Vector[i]*3/5) <= Vector[i+1] && Vector[i]-Vector[i+1] >=0)
        {
                // May be Vally
                ////////starting at the peak///////
                if(!SameAsPrevInVector(VectorID, Index,'1'))
                {
                        PutInVector(VectorID, Index, '1');
                        Index++;
                        dc.SetPixel(i+Dx, Dy-Vector[i], RGB(255,0,0));
                }
                ////////////////////////////////////
                count=i;
                while((Vector[i]*3/5) <= Vector[i+1] && Vector[i]-Vector[i+1] >=0)

                        i++;
                count=i-count;
                if(count>3)
                {
                        //Vally confirmed
                        if(eqflag)
                        {
                        // Insert extra zero as velly
                                eqflag=false;
                        if(!SameAsPrevInVector(VectorID, Index,'0'))
                                {
                                        PutInVector(VectorID, Index, '0');
                                        Index++;
                                        dc.SetPixel(i+Dx, Dy-Vector[i], RGB(255,0,0));
                                }
                        }

                        if(!SameAsPrevInVector(VectorID, Index,'0'))
                        {
                                PutInVector(VectorID, Index, '0');
                                Index++;
```

```
                        dc.SetPixel(i+Dx, Dy-Vector[i], RGB(255,0,0));
                }
        }
}
else if((Vector[i+1]*3/5) <= Vector[i] && Vector[i+1]-Vector[i] >=0)
{
        // May be peak
        ////////starting at the valley///////
        if(!SameAsPrevInVector(VectorID, Index,'0'))
        {
                PutInVector(VectorID, Index, '0');
                Index++;
                dc.SetPixel(i+Dx, Dy-Vector[i], RGB(255,0,0));
        }
        ///////////////////////////////////

        count=i;
        while((Vector[i+1]*3/5) <= Vector[i] && Vector[i+1]-Vector[i] >=0)
                i++;
        count=i-count;
        if(count>3)
        {
                //peak confirmed
                if(eqflag)
                {
                // Insert extra 1 as peak
                        eqflag=false;
                        if(!SameAsPrevInVector(VectorID, Index,'1'))
                        {
                                PutInVector(VectorID, Index, '1');
                                Index++;
                                dc.SetPixel(i+Dx, Dy-Vector[i], RGB(255,0,0));
                        }
                }

        if(!SameAsPrevInVector(VectorID, Index,'1'))
                {
                PutInVector(VectorID, Index, '1');
                Index++;
                dc.SetPixel(i+Dx, Dy-Vector[i], RGB(255,0,0));
                }
        }
}
else
        i++;
```

CHAPTER 6

RESULTS

6. Results

Every image needs preprocessing to make it suitable for subsequent processing that may include segmentation, feature extraction of region found in image and finally recognition.

6.1 Non-Uniform Illumination Removal Results

The technique of non-uniform illumination is applied on different images taken by different acquisition devices to produce an image with balanced illumination.

Figure 6.1 Original Angiogram (Image1)

Figure 6.2 Illumination Pattern of Angiogram (Image1)

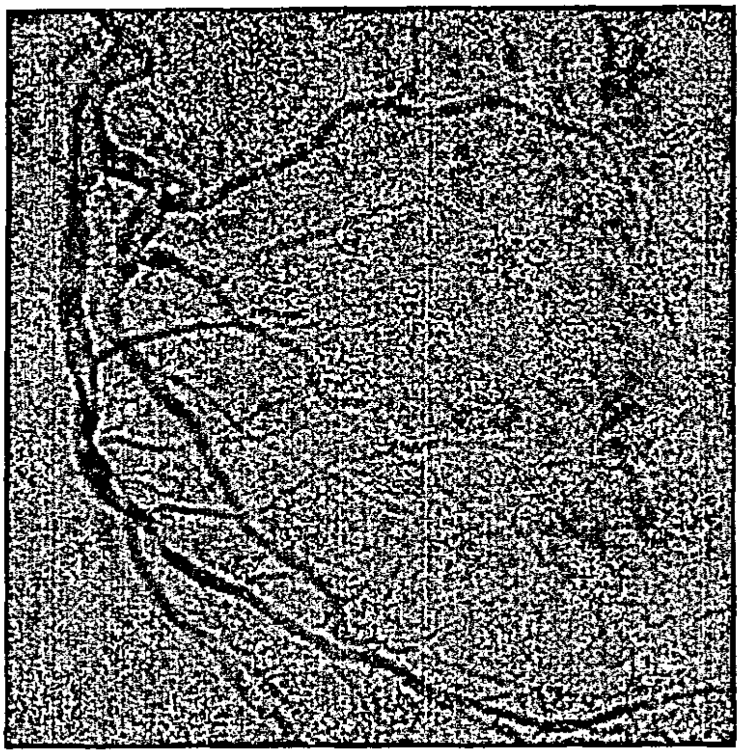

Figure 6.3 Angiogram Image After Adding Illumination (Image1)

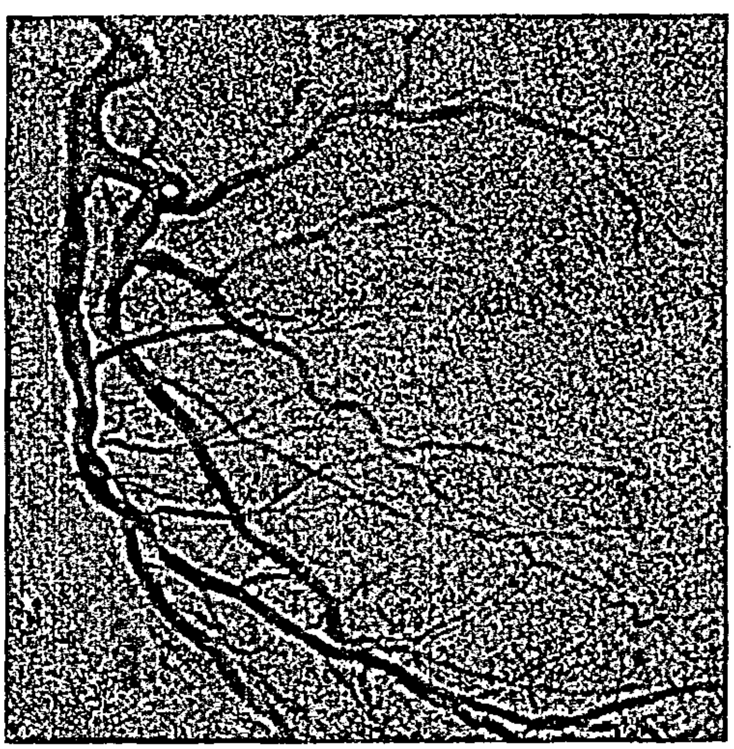

Figure 6.4 Image After Contrast Stretching (Image1)

Figure 6.5 Original Angiogram (Image2)

Figure 6.6 Illumination Pattern of Angiogram (Image2)

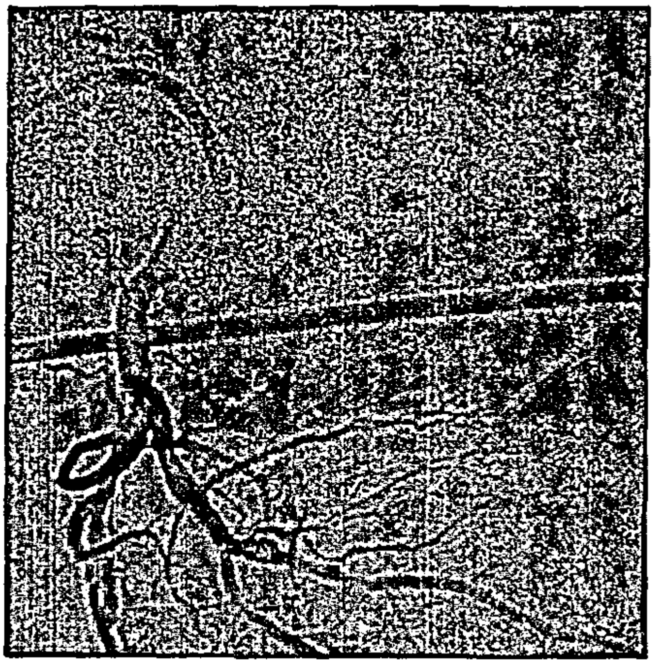

Figure 6.7 Angiogram Image After Adding Illumination (Image1)

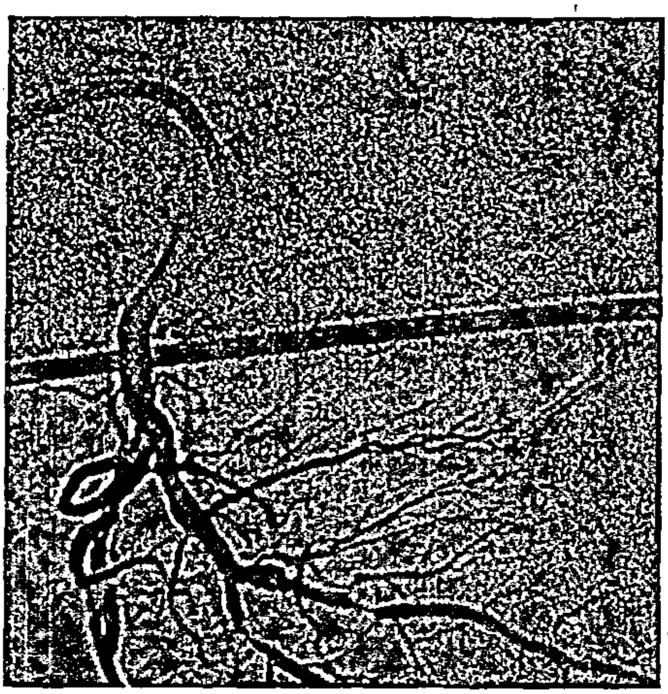

Figure 6.8 Image After Contrast Stretching (Image1)

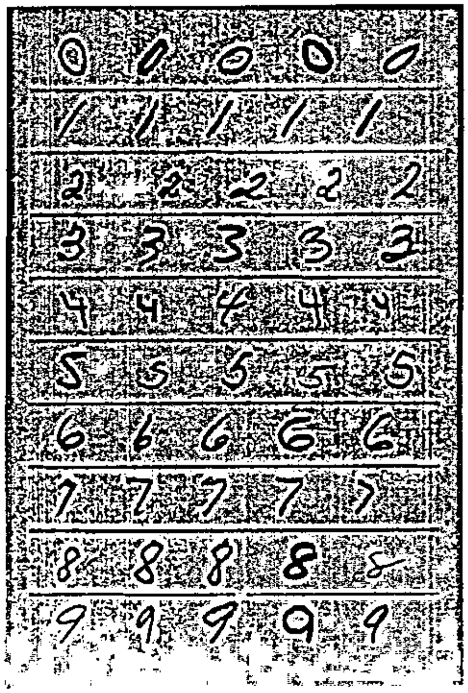

Figure 6.9 Digit Sheet with Non-uniform Illumination

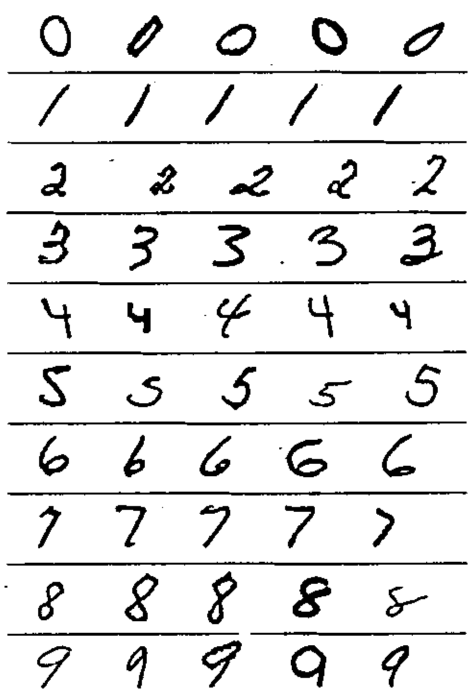

Figure 6.10 Processed Image of Digit Sheet

6.2 Digit Recognition Results

Process of recognition is comprises the drawing of minimum enclosing rectangle containing the digit inside the area, then finding the depth projection profiles around all four directions to find out the binary features. Using the prototype structural information the binary features obtained of any digit, are compared to declare it belonging to any class or may be rejected.

Figure 6.11 Minimum Enclosing Rectangle Around Digit '0'

(a) (b) (c) (d)

Figure 6.12 Depth Projections for digit '0'. (a) Left (b) Right (c) Bottom (d) Top

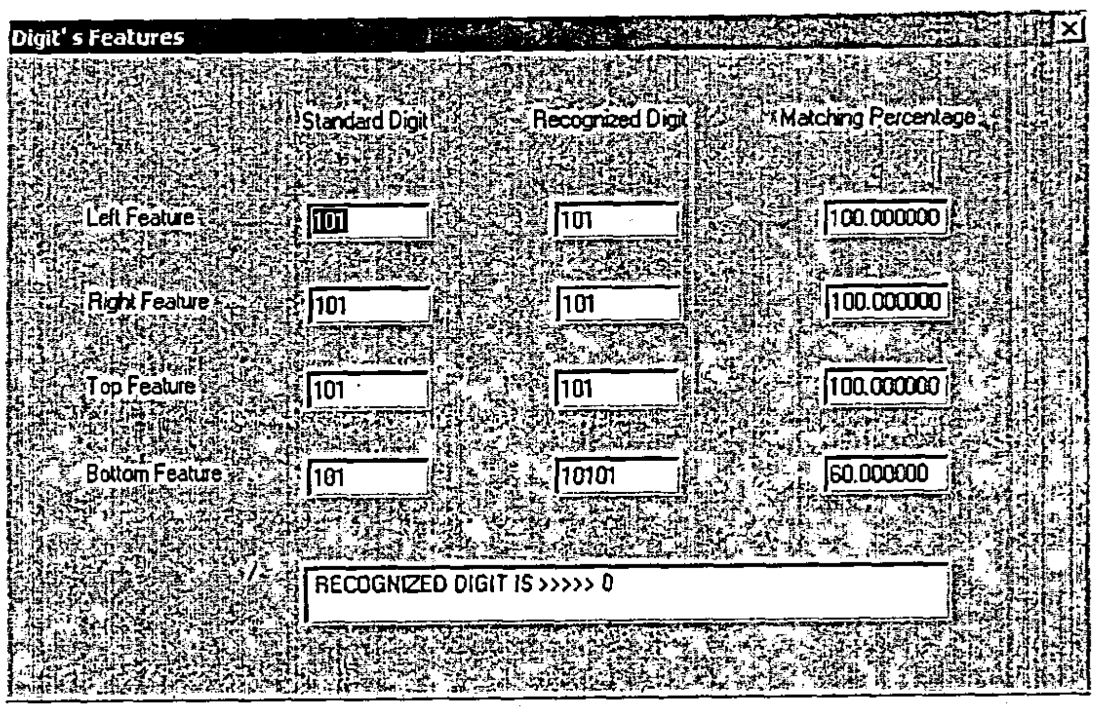

Figure 6.13 Recognition of Digit '0' Based on Structural Prototype

Figure 6.14 Minimum Enclosing Rectangle Around Digit '3'

(a) (b) (c) (d)

Figure 6.15 Depth Projections for digit '3'. (a) Left (b) Right (c) Bottom (d) Top

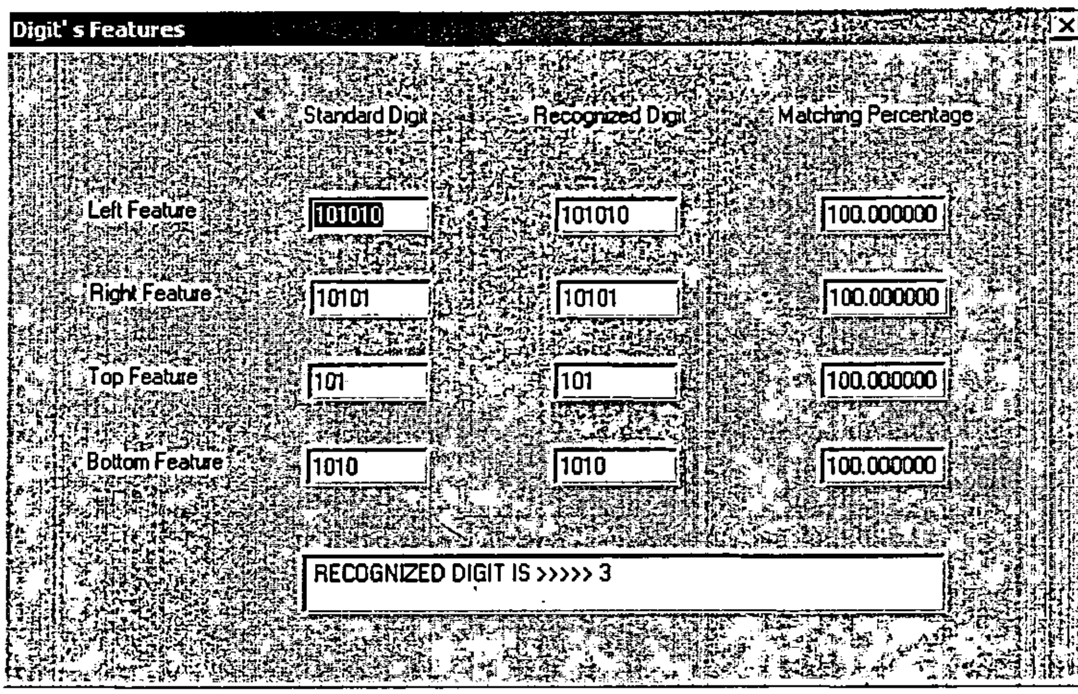

Figure 6.16 Recognition of Digit '3' Based on Structural Prototype

Figure 6.17 Minimum Enclosing Rectangle Around Digit '9'

(a) (b) (c) (d)

Figure 6.18 Depth Projections for digit '9'. (a) Left (b) Right (c) Bottom (d) Top

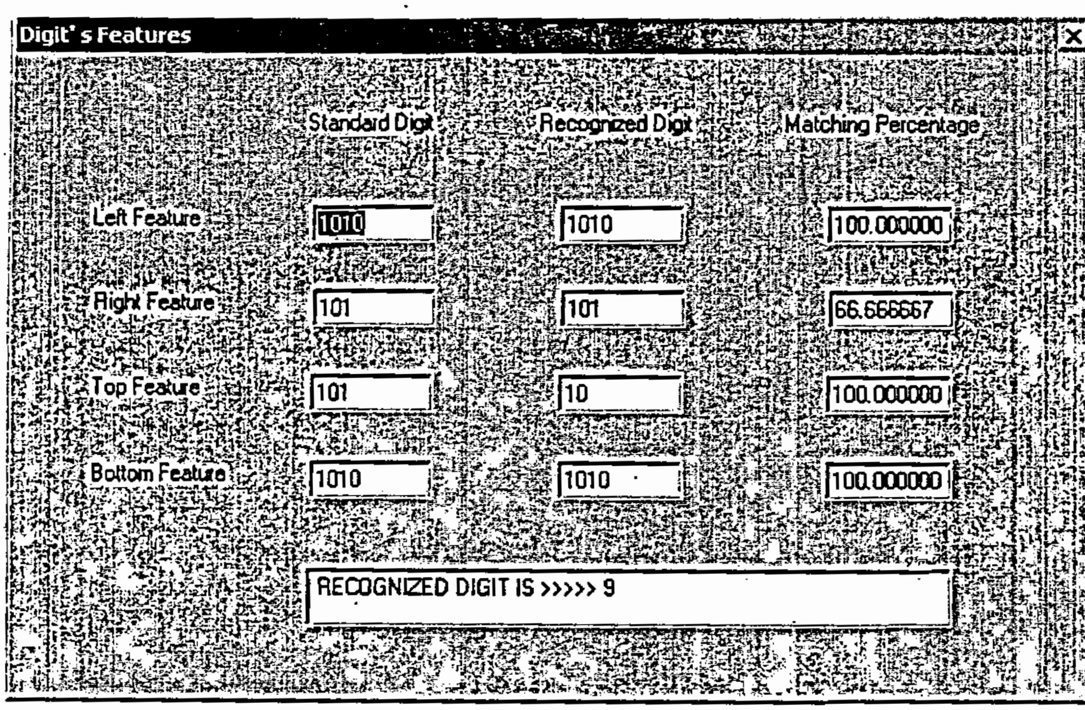

Figure 6.19 Recognition of Digit '9' Based on Structural Prototype

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7. Conclusion and Future Enhancements

During the acquisition process of images, improper source of illumination may cause variable light effect on different parts of digital image. This may happen in an image acquired by camera, X-ray or ultrasound machine. Another cause of improper light could be the error introduced by the sensors of acquisition device. The image thus acquired may loose the contents of interest that should be the outcome of later processes like segmentation where the separation of region from the background is necessary. Also recognition can be carried out successfully only if the image exhibits linear illumination. A significant improvement has been made in this way in the appearance of the images. This sustains the actual objects as well making it worthier to be used by the subsequent processing than that of non-uniform illuminated actual image.

Recognition of handwritten text has been a problem that is attempted to solve by applying a variety of methods. In this report the structure of digits is obtained using depth profile projections around the four sides of any digit by drawing minimum enclosing rectangle around it.

7.1 Conclusion

Every image up to some extent is non-uniformly illuminated during the process of acquisition as shown in Figure 6.1 Figure 6.10. It is a best approach to remove non-uniform illumination from the image before applying enhancement technique. Enhancement comes under the pre processing method for many digital image applications that include segmentation, noise removal etc. Segmentation method works better, if the image foreground is separable from background with certain difference in pixel values. So the application of this method can significantly improve the segmentation and separation of varying intensity valued regions such as veins in angiogram etc.

This report applies the scheme of non-uniform illumination removal from digital image containing hand-written characters that makes the segmentation of image simpler and a single threshold is sufficient to produce a binarized image. Recognition is the then performed by obtaining depth projection profile around four sides of digit. Using these projections, binary features of left, right, top and bottom structure are obtained. These are then matched with the prototype features with 10 digit domains to find a closest match.

7.2 Future Enhancements

Exhaustive applications using digit recognition system and removal of non-uniform illumination are needed. Character recognition has been the subject of intensive research during the last decades. It is a very challenging scientific problem but provides a solution for processing large volumes of data automatically. Now a days character recognition, even of handwritten script has numerous applications such as address and zip code recognition, car number plate recognition, writer identification, etc.

Unique properties of signature allow identifying a person uniquely; the process may be manual or automated, in the same way hand-written text may also help to identify through the careful selection of features. The approach could be combine use of scalable vector temples instead of fixed and depth projection profile.

Enhancement plays an important role and background separation from the object to be recognized is also crucial. A care should be taken that reflectance component is not removed when illumination is tackled.

APPENDIX A

BIBLIOGRAPHY AND REFERENCES

Bibliography and References

Books & Thesis

1. "Digital Image Processing"; Rafael Gonzalez, Richards Woods; Third Edition; Published by Pearson Education, Singapore.

2. "Practical Computer Vision Using C", J. R.. Parker, ISBN 0-471-59262-5.

3. "Pattern Recognition: Statistical, Structural and Neural Approaches"; Robert J. Schalkoff; First Edition; Published by John Wiley & Sons Inc.; 1992.

4. "On Chromatic and Geometrical Calibration"; Jørgen Folm-Hansen Ph.D. in engineering from the Technical University of Denmark.; Lyngby 1999; IMM-PHD-1999-61; Ph.D. Thesis

5. "Fundamentals of Computer Vision"; Mubarak Shah; University of Central Florida; 1997.

6. "Applying UML and Patterns"; Craig Larman; Prentice Hall; 2002

Publications

6. Rapantzikos, K. Zervakis, M., 2001. "Nonlinear enhancement and Segmentation algorithm for the detection of age-related macular degeneration (AMD) in the eye's retina". Proceedings of ICIP 2001, Thessaloniki, Greece, October 2001.

7. Oppenheim, Schafer and Stockham. "Nonlinear Filtering of Multiplied and Convolved Signals". IEEE Proceedings and Transactions on Audio and Electroacoustics.

8. Norman Haas, Nalini K. Ratha, Ruud M. Bolle, "Pre-Enhancing Non-Uniformly Illuminated Fingerprint Images". Auto ID 2002 – Tarrytown, NY, IEEE Conference on Automatic Identification Advanced Technologies, March 2002.

9. Roberto J. Rodrigues, Antonio C. G. Thomé, Cursive character recognition – a character segmentation method using projection profile-based technique, The 4th World Multiconference on Systemics, Cybernetics and Informatics SCI 2000 and The 6th International Conference on Information Systems, Analysis and Synthesis ISAS 2000 – Orlando, USA – August 2000

APPENDIX B

PUBLICATION