**Software Requirements Specification**

**on**

**Devanagari Optical Character**

**Recognition**

Submited by:

Ashay Churi

Mayur Navarkar

Durgesh Waghmare

**1 Introduction**

**1.1 Purpose :**

Currently there are many systems are available for performing the optical character Recognition .

All these systems are build for standard foreign languages like English , French , Spanish and German . They does not support the Recognition of regional languages like Marathi , Sanskrit , Hindi etc . We are going to build a Software which is capable of recognizing Devanagari Script Characters .

Our main aim is to recognize these characters and to store or save them into a standard text file so that the user can easily go through them and can perform various text related operations like Copy , Paste , Find , Replace etc .

**1.2 Intended Audience :**

Devanagari OCR System is mainly intended for the audience who wish to extract the Devanagari Script Characters from the image . It can came into use in variety of sectors like

* Effective pattern Recognition
* Forensic Science
* Processing Checks
* Documenting Library Materials

**1.3 Product Scope :**

This system can be used by multiple users. We can do this by improving our software for recognizing the handwriting of more than one user. Also if we can take the stroke information and give it to our system, then it will be possible to recognize even cursive script also.

The recognized characters are stored in the text file. We can add words to the sound files and invoke them through the program, so that the recognized words can be read aloud. Thus we can make the computer read the handwritten document.

* 1. **References :**

[1] Optical Character Recognition by Open Source OCR Tool Tesseract: A Case Study - International Journal of Computer Applications (0975 – 8887) Volume 55– No.10, October 2012.

[2] A Survey on Offline Recognition of Handwritten Devanagari Script by Ashwin S Ramteke, Milind E Rane International Journal of Scientific & Engineering Research Volume 3, Issue 5, May-2012 1 ISSN 2229-5518

[3] A Bilingual OCR for Hindi-Telugu Documents and its Applications

By C. V. Jawahar, M. N. S. S. K. Pavan Kumar, S. S. Ravi Kiran

Centre for Visual Information Technology,International Institute of Information Technology,Gachibowli, Hyderabad, India - 500 019

[4] Tesseract OCR Engine : <https://code.google.com/p/tesseract-ocr/>

[6] B.B. Chaudhuri and U. Pal — Skew Angle Detection of Digitized Indian Script Documents

[7] Huanfeng Ma, David Doermann — Adaptive Hindi OCR Using Generalized Hausdorff Image Comparison

[8] Vijay Kumar, Pankaj K. Sengar — Segmentation of Printed Text in Devanagari Script and Gurmukhi Script

[9] Mudit Agrawal, M. N. S. S. K. Pavan Kumar, C. V. Jawahar — Indexing and Retrieval of Devanagari Text in Printed Documents

[10] R. Jayadevan, Satish R. Kolhe, Pradeep M. Patil, and Umapada Pal — Offline Recognition of Devanagari Script: A Survey

1. **Overall Description**

**2.1 Product perspective :**

Machine replicating human functions, like reading, is an old dream. However, over the last five decades, machine reading has grown from a dream to reality. Machine reading uses the principles of Optical Character Recognition (OCR). OCR has also become one of the most successful applications of technology in the field of pattern recognition and artificial intelligence. Since the mid 1950s, OCR has been a very active field of research and development. While the OCR technology for some scripts like Latin is fairly mature and commercial OCR systems like Nuance OmniPage Pro or ABBYY FineReader are available which can perform with high accuracy, it is still under development for other scripts like Chinese and Devangari.

Although a great deal of research has been done for OCR applications for Latin script, even theses OCR based machines are still not able to compete with human reading capabilities. This problem is more prominent for other scripts for which OCR technology is relatively newer. Typefaces are very important in determining the performance of the OCR technology. Hence in order to improve the accuracy of the OCR system, typefaces which are specially designed for OCR are   
required. For Latin script, quite a few typefaces have been designed which are optimized for OCR. These specially designed typefaces have a unique and well defined character set which allows for greater accuracy in recognition. This in turn helps in building low cost systems which can recognize characters using simple algorithms. However, no Devanagari script font is available which is designed specifically for machine reading and we address this problem in this report.

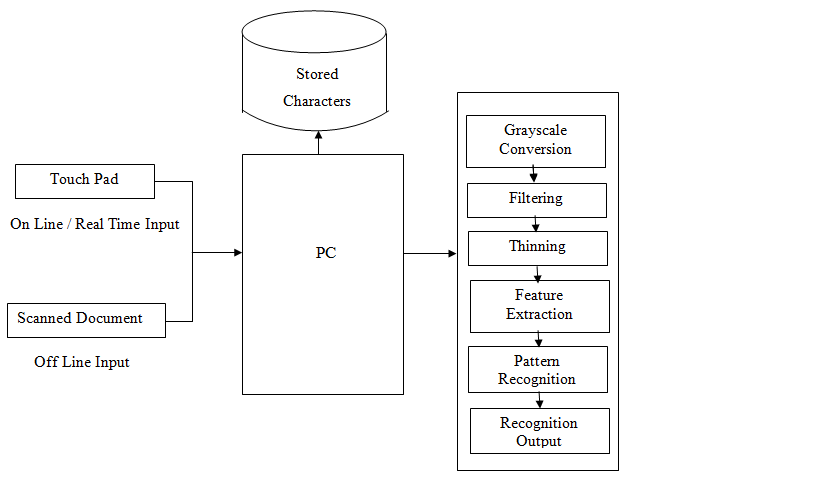
In general, documents contain text, graphics, and images. The procedure of reading the text component in such a document can be divided into three steps:

1. Document layout analysis in which the text component of the document is extracted.

2. Segmentation, i.e. extraction of characters from the text component of the document.

3. Recognition of the segmented characters.

The basic building blocks of OCR System are as follows



*(figure 2.1.1)*

**Recognition of Devanagari Script**

The most important principle of automatic pattern recognition is training the machine what kind of pattern may be present and what they look like. In OCR the patterns are letters, numbers and punctuations. Machine is trained to recognize the pattern by showing it all the kind of characters present in the script. This period is referred as the training period. On the basis of these examples

the machine builts a prototype of all the characters. Then during recognition the machine compares the unknown character to the prototype and assigns the character which is the closest match. The four steps in recognition shown in figure 3.2 are as follow:

1. Preprocessing

2. Segmentation

3. Recognition

4. Post Processing

Preprocessing

The text document is generally scanned at 300 or 400 DPI. Preprocessing is also done to improve the accuracy of the recognition algorithm. Main steps in preprocessing are noise removal, binarization

and skew correction.

**2.1.1 Image Preprocessing (Noise Removal or De-Noising) :**

The main sources of noise in the input image are as follows:

• Noise due to the quality of paper on which the printing is done.

• Noise induced due to printing on both sides of paper or the quality of printing

• Noise added due to the scanner source brightness and sensors.

All this noise results in reduction of accuracy of OCR system. As a result of this having a noise correction routine in place becomes inevitable. To reduce the amount of noise, image is passed through a mean filter; in this filter the intensity of the each pixel is replaced by the average intensity of pixels surrounding it. After de-noising the image is subjected to binarization and skew (or tilt) correction.

Binarization

Printed documents generally are black text on white background. Hence most of the OCR algorithms

are designed to interpret bi-level images (an image that has only two possible value of pixel i.e. black and white). This process of converting colored or grayscale images to bi-level image is often known as binarization or thresholding.

*(Figure 2.1.1a ) (Figure2.1.1 b)*

*Original Image Binarized Image*

**2.1.2 Segmentation :**

Segmentation is the process of the dividing the page into its constituent element. The aim of segmentation is to extract out all the character from the text in the image. This is needed to recognize these characters.

Segmentation phase is a very crucial stage since this is where most of the errors occur. Even in good quality documents, sometimes adjacent characters touch each other due to inappropriate scanning resolution or the design of characters. This can create problems in segmentation. Incorrect segmentation leads to incorrect recognition. Segmentation phase includes line, word and character segmentation. Segmentation in OCR occurs in three steps: line segmentation, word segmentation and character segmentation. an overview of segmentation process is given below.

**2.1.3 Line Segmentation :**

In line segmentation our aim is to separate out the line of text from the image. For this global horizontal projection profile method is used which constructs a histogram of all the black pixels in every row as shown in figure 3.4. Based on the peak/valley points of the histogram, individual lines are separated. The steps for line segmentation are as follow:

1. Horizontal projection profile for the image is created.

2. Using the projection profile, the points from which the line starts and ends are found.

3. For a line of text, upper line is drawn at a point where we start finding black pixels and lower line is drawn where we start finding absence of black pixels. And the process continues for next line and so on. Character Segmentation

Once the words are segmented, the next step is to extract out the characters form these words. A word in Devanagari script is further divided into three parts: as shown in figure 3.6:

1. Top

2. Core (or Middle)

3. Bottom

The top strip and the core part are separated by the Header Line or the Shiro Rekha. But there is no separation between the core strip and the bottom strip. The top strip contains the top matras and the bottom strip contains the bottom matras or the descenders of some on the characters. The Shiro Rekha is a unique feature of Devanagari script and helps to identify Devanagari in multi-lingual document. It also helps in the identification of the baseline of the text.

The steps of character segmentation shown in figure 3.7 are as follows:

1. Shiro Rekha is identified and the top strip is seperated from the core and bottom strip. So now the text is divided in two parts a.) The Shiro Rekha and the top mantra and b.) The core-bottom part of the text

2. Core strip and bottom strip from the core-bottom part of the text, is identified and lower matras are extracted.

3. Core strip is segmented into different letters or characters which may include conjuncts, punctuation

or numerals.

4. Conjuncts are segmented into single characters.

5. Shiro Rekha is removed form the extracted top strip and top matras are extracted.

6. Once the segmentation of the core character is done, Shiro Rekha is put back on the top of individual

characters.



*(Figure 2.1.3)*

**2.1.1.3 Recognition :**

Segmentation is followed by recognition of the characters. The two main methods used for recognizing

characters are as follows:

• Template Matching

• Feature Based Recognition

Template Matching

In this method a matrix containing the image of the input character is matched with the set of prototypes created in the training period. The distance between the pattern and each prototype is computed and the character which is the best match to the pattern is assigned to the pattern.

The technique is simple and easy to implement in hardware. However, this technique is sensitive to noise and style variations and has no way of handling rotated characters.

Feature Based Recognition

In this method significant features of the pattern are measured and examined. These features are then compared to the prototypes developed in training phase. The description which provide the closest match provides the recognition. These features can be like presence of vertical bar or the number of conjunctions.

**2.1.1.4 Post Processing :**

The result of recognition is set of some characters. However these characters doesn't contain the complete information. We would like to combine these individual characters to form strings. This process is called grouping. Grouping of string depends on the location of string in the document. Strings which are close to each other are grouped together to form a word, since the distance between two words is more than the distance between the letters of the word.

**2.2 Product Functions :**

The main function of Devanagari OCR System is

* To recognize the Devanagari Script Characters from the image and to save or the recognized characters into a standard text file.

Additionaly we are also performing the image enhancement operations to reduce the background noise from the image in order to increase the efficiency of recognizing process.

**2.3 Operating Environment :**

Currently we are developing the DevanagariOCR software only for the Windows operating system . Normal system requirements for running this software are as follows

* + 1. Software Requirements :

(1) Windows O.S. (Either Windows XP or Windows 7)

(2) MATLAB R2012b

2.3.2 Hardware Requirements:

(1) Intel core i3 Processor

(2) Minimum of 2GB RAM

(3) Minimum of 2GB of HDD Space

(4) Image Scanner (Optional)

**2.4 Design And Implementation Constraints**

Recognition of Characters is much more complicated Devanagari Script than any other languages because of the graphical similarities in the letters .

Some of the letters have just a difference of a stroke like ष just has an additional diagonal stroke as compared to प. While there are others which differ from each other only because of the presence of vertical line like न and म. Also unlike Latin script, Devanagari has letters which are disjoint horizontally. This should be avoided in the characters in which this can be avoided for example रव can also be designed as ख. This results in inaccurate recognition.

Also the open counters in the letters should be designed carefully. Open counter is the curved part of the character that encloses curved parts (counter) of some letters as shown in figure 2.4



*Figure 2.4 Example of open counter (grey)*

While designing the counters, special care need to be taken so that the strokes forming these curves don't get connected because of noise or smudging. This results in the algorithm to confuse between two letters. For example if the strokes of ल connects together they can be recognized as न.

* 1. **Assumptions and Dependencies :**

Users who want to use the DevanagariOCR Software must have a primary knowledge of how to use the Software. To avoid any kind of difficulties while using the system user must read the User Manual which is provided along with the Software

DevanagariOCR Software is totally depended on -

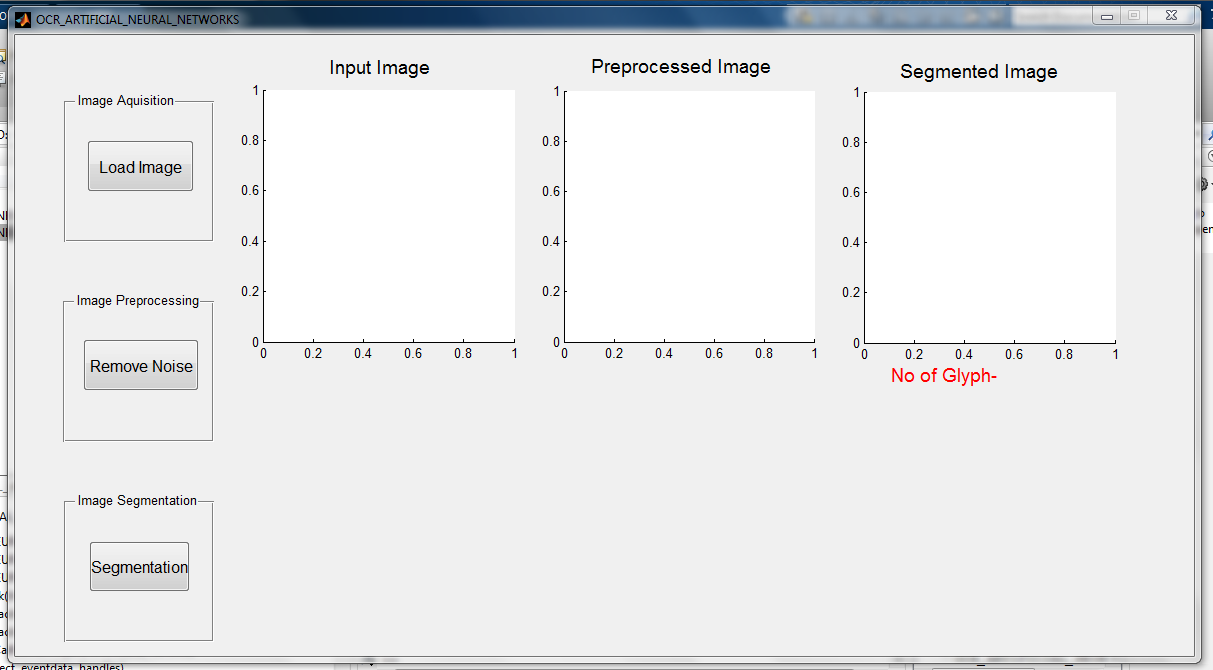
* Tesseract Framework
* Microsoft .Net 4.0 Framework

If these Framework will get disturb due to some reasons the overall system will not be of any use . Users who are willing to use our software will make sure that they won’t cause any kind of harm to these two Frameworks.

1. **External Interface Required :**

**3.1 User Interfaces :**

The simple interface that we are presenting to user is as follows :



*(Figure 3.1.1)*

Functions of buttons are as follows :

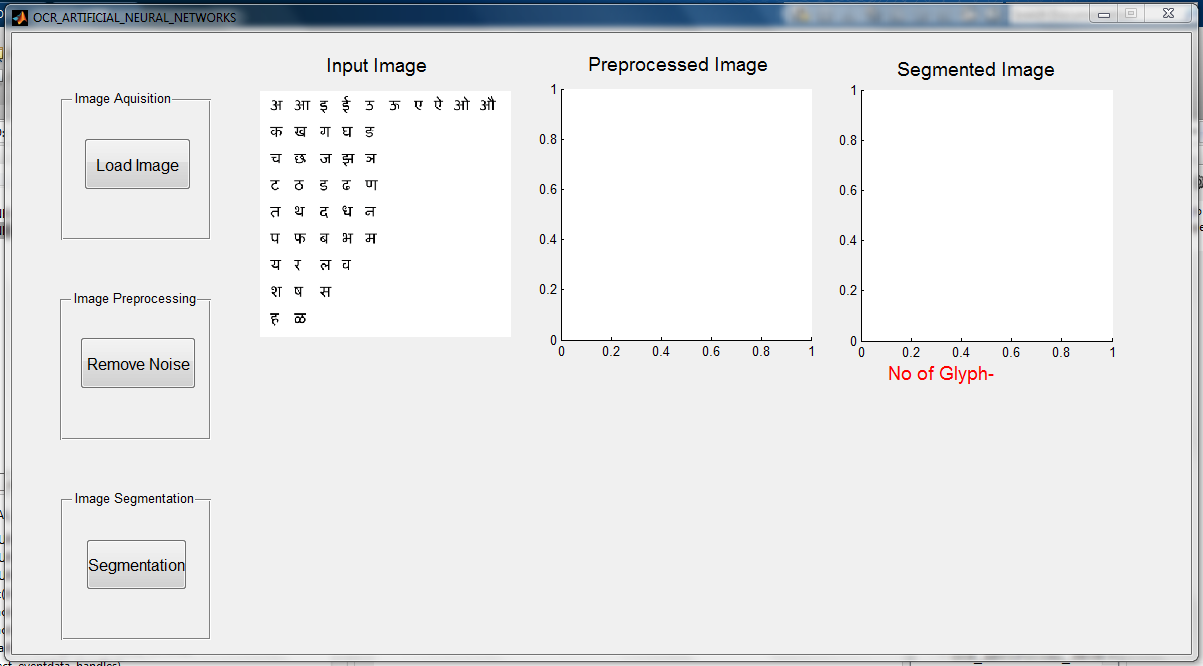
Load : Allows user to browse the image for Optical Character Recognition. Selected image will be then displayed in the image box.

Remove Noise : Functions performed by this function are as follows-

1. Convert image into Grey Scale Image
2. Convert image into Binary Format
3. Remove Salt and Paper noise from the image
4. Display modified image into the second picture box

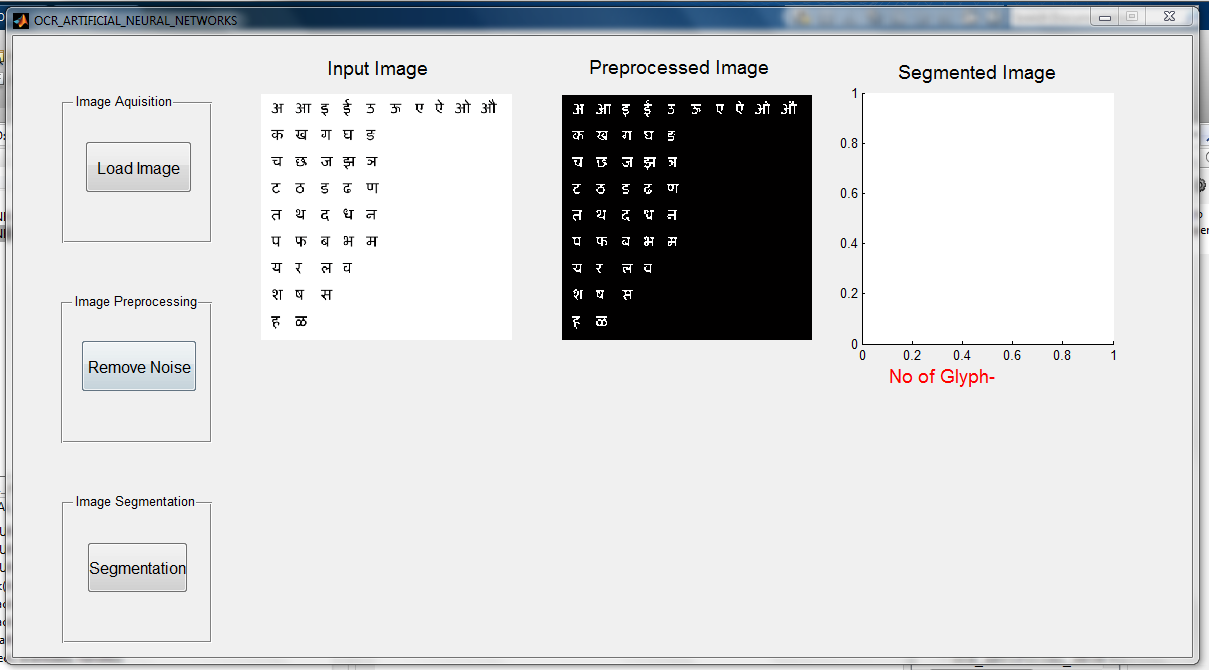
Segmentation: It performs the segmentation of image and calculates no of glyphs present in the modified image(i.e. the image without noise).

After selection of image:



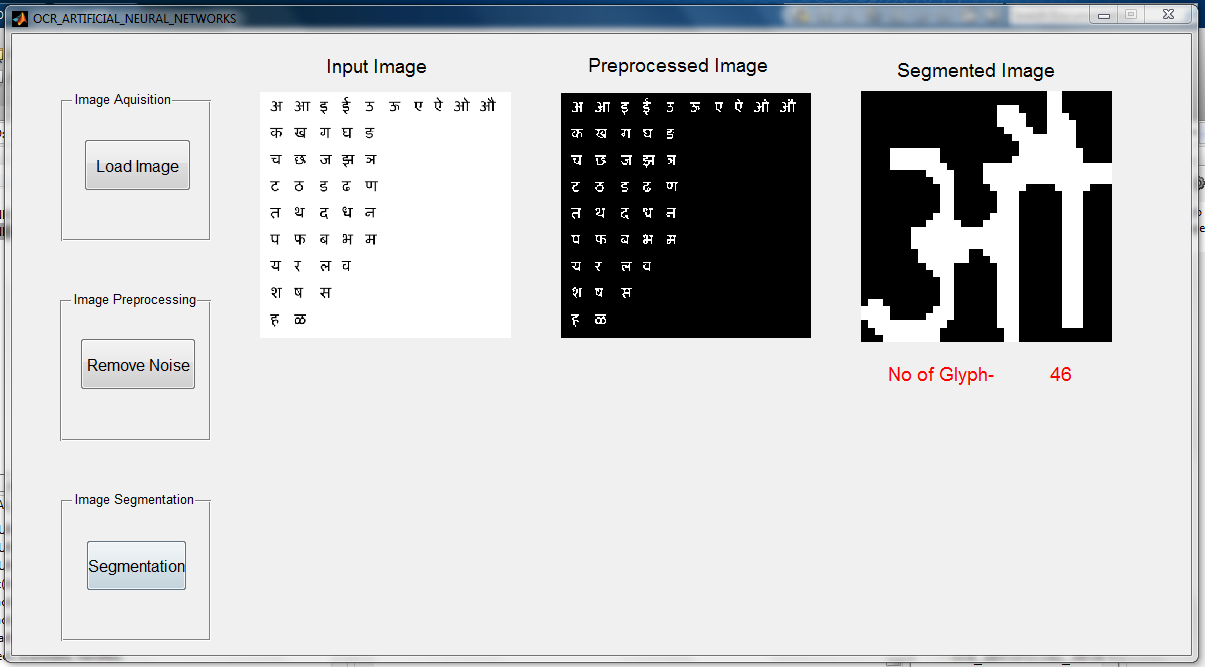
*(figure 3.1.2)*

After Noise Reduction :



*(figure 3.1.3)*

After Segmentation:



*(figure 3.1.4)*

1. **System Features :**

Usually, OCR uses a modular architecture that is open, scalable and workflow controlled. It includes forms definition, scanning, image pre-processing, and recognition capabilities.

*(figure 4.1.1)*

**4.1 Load Scanned Images :**

For static OCR, software should provide a way to load scanned document for recognition purpose.

**4.2 Image Modification (Enhancing the Image) :**

The input image might contains some kind of noise, to remove such noise we are applying various image processing filters like-

1. Grey Scale Conversion
2. Image Binarization
3. Salt and Pepper noise reduction

**4.3 OCR :**

Software should process the image and extract characters.

**4.4 Save Result :**

User should have facility to save extracted data in format of his interest.

**4.5 Recognize characters dynamically :**

For dynamic OCR, the software should recognize characters drawn by user simultaneously.

**5 Non-Functional Requirements :**

* The input image is to be in the bitmap file format
* In case of scanned image, a high quality scanner as well as good paper quality is required. The resolution of the scanner should be set to a minimum of 300 dots per inch (dpi).
* During scanning a maximum tilt of up to 20º can be corrected.
* In case of discontinuities in the hand written characters a maximum gap of up to 3 pixel wide thickness is tolerable.

*(figure 5.1)*

**5.1 Performance Requirements :**

**5.1.1 Accuracy :**

The extent to which a program satisfies its specification and fulfils the customer mission objective.

**5.1.2 Reliability :**

The extent to which a program can be expected to perform its intended function with require precision.

**5.1.3 Speed :**

The time require for a program to perform the given task.

**5.1.4 Maintainability :**

The efforts required to locate and fix an error in the program.

**5.1.5 Portability :**

The efforts required to transform a program from one hardware and/or software system environment to another.

**5.1.6 Availability :**

The system is expected to be available around the clock as it will be further used to analyze blood slides at the installed site**.**

## 5.2 Safety Requirements

## OCRs also have huge application for the blind. This was one of the earliest thought applications of OCRs. Combined with text to speech conversion OCRs would enable blind people to read the printed documents. It can also be used for automated license plate reading and can also help in reading specially designed forms automatically. Once the text is computerized it can be used for machine processes like text to speech conversion, language translation and text mining. This can also be considered as a safety measurement for blind People.

## 5.3 Security Requirements

As per concerning the security requirements, OCR has been used to computerize data for dissemination and processing. The first major use of OCR was in the banking industry where it was first used to read credit card numbers. Nowadays OCRs are widely used for automated data entry especially in banks where it is used to read account number, customer identification, amount of money etc.It is also used for text entry i.e. extracting text out of a scanned document. The reading machine is used to process large amount of text, which can then be used of several other purposes like for searching within the document.

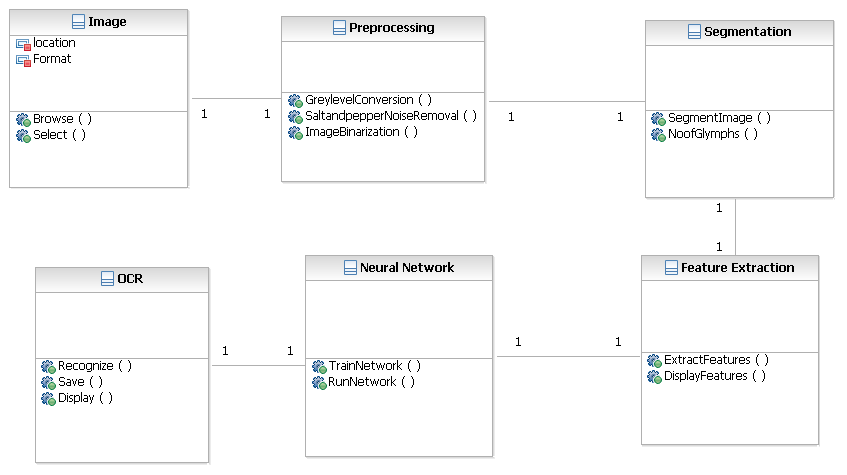
**6 Appendix A: Glossary**

**6.1 Abbreviations:**

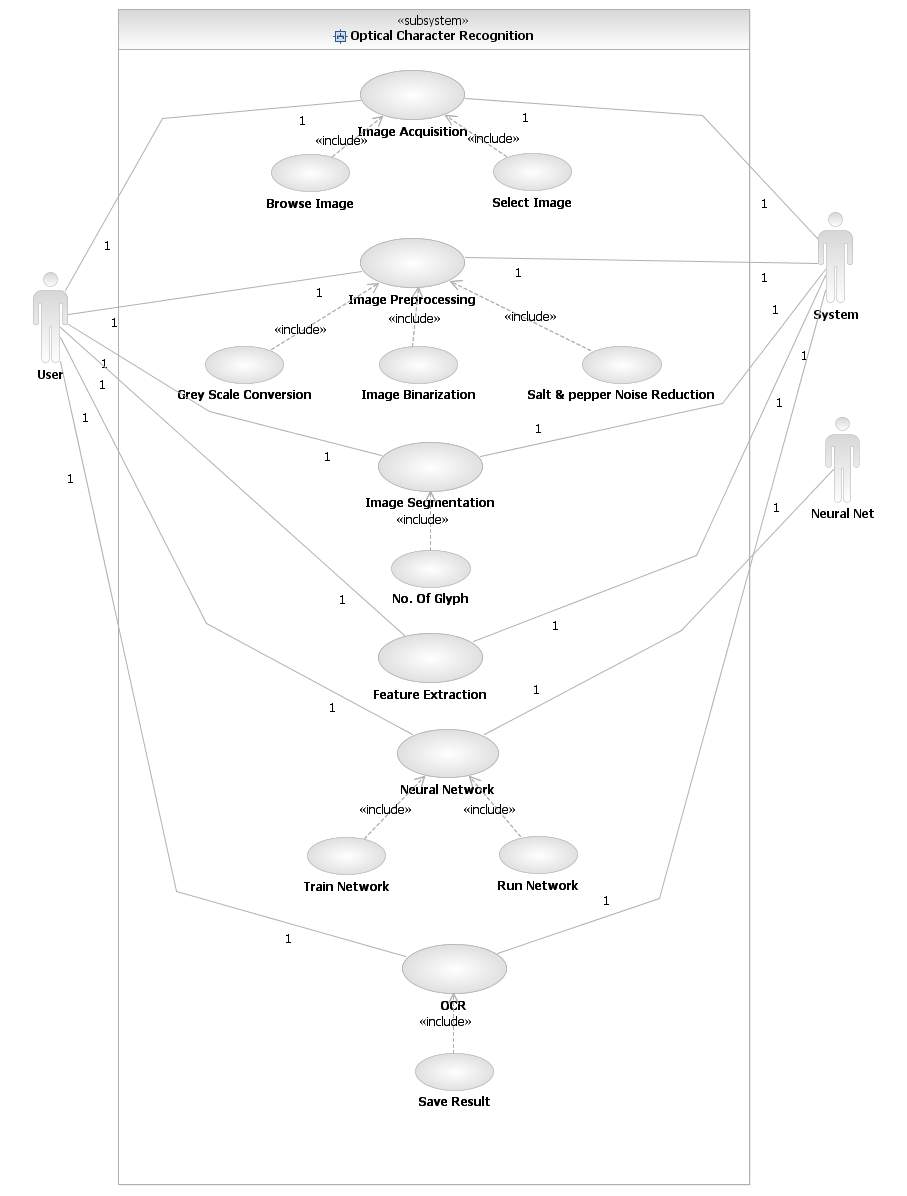
|  |  |
| --- | --- |
| ANN | Artificial Neural Networks |
| ANSI | American National Standards Institute |
| BPM  DPI | Bitmap  Dots Per Inch |
| ECMA | European Computer Manufacturers Association |
| FFNN | Feedforward Neural Network |
| Gb | Giga byte |
| GHZ | Giga Hertz |
| GIF  ISO | Graphic Interchange Format  International Standards Organization |
| JPEG/JPG  MATLAB | Joint Photographic Expert Group  Matrix Laboratory |
| M-files | Matlab Files |
| NN | Neural Networks |
| NNT | Neural Networks Toolbox |
| OCR  OS | Optical Character Recognition  Operating System |
| PDF | Portable Document  Format |
| RTF | Rich Text Format |
| TIFF | Tagged Image File Format |
|  |  |
|  |  |

**7 Appendix B : Analysis Model**

**7.1 Class Diagram :**

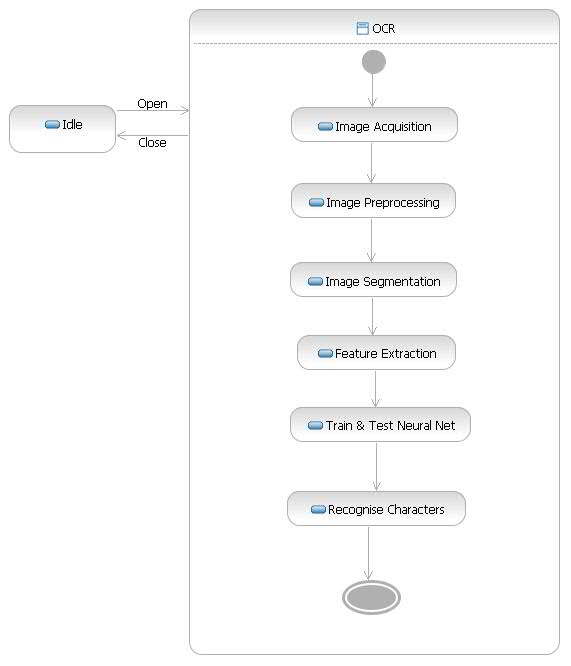
****

*(figure 7.1.1)*

**7.2 Use Case Diagram : **

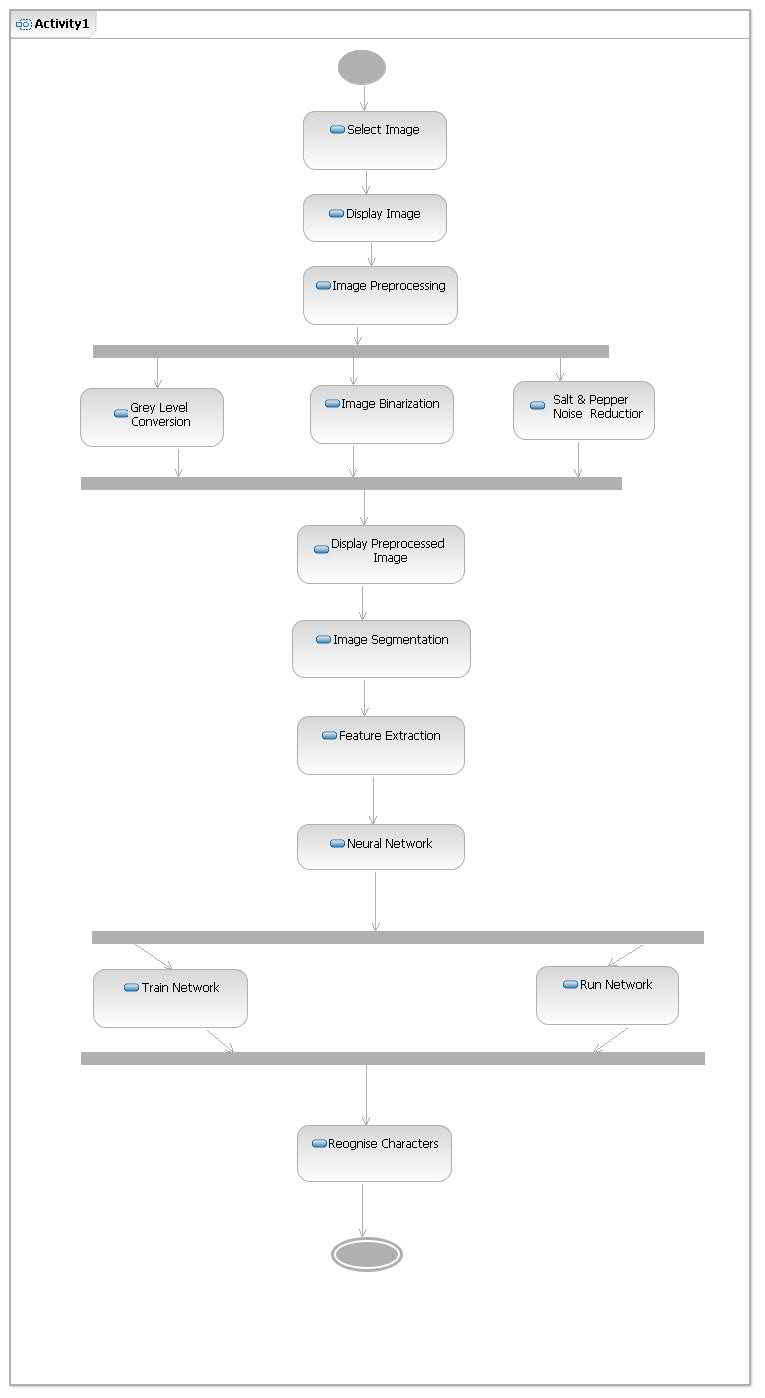
*(figure 7.1.2)*

**7.3 State Diagram :**

****

*(figure 7.1.3)*

**7.4 Activity Diagram :**

*****(Figure 7.1.4)*