International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 1, ISSN 2249 –071X

Devnagari Script Segmentation Based on JNI (Java Native Interface)

Pinal K. Shah and Preeti K. Dave
pinal_cp@yahoo.com, preetidave@gmail.com.

***Abstract* — Optical Character Recognition (OCR) is a form of computer vision that extracts alphanumeric characters from a digital image. The technology can be used for digitizing printed text, handwriting recognition, and making digital images searchable for text. Following the survey, an implementation of OCR Segmentation in JAVA will be presented and analyzed.**

OCR system converts scanned input document into editable text document. This paper presents the detailed description about the characteristics of Devnagari Script. How it is different from the other roman scripts. And what makes the Segmentation for any roman script different from the Segmentation for Devnagari script. The various stages of an OCR system are: upload a scanned image from the computer, segmentation process in which we extract the text zone from the image, recognition of the text and the last which is post processing process in which the output of the previous stage goes through the error detection and correction phase. This paper explains about the user interface provided with the OCR with the help of which a user can very easily add or modify the segmentation done by the OCR system.

***Index Terms* — JNI, CLP, XCLP, SVM**

I. INTRODUCTION

Optical Character Recognition is an important and practical technology in the computer age. More people than ever before are using personal computers, laptop, tablets, and e-readers to read documents and books. This means that old print media must be scanned and converted to a digital format in order to be accessed from these devices. Optical Character Recognition (OCR) programs are used to read scanned images and convert them into a digital character-based format.

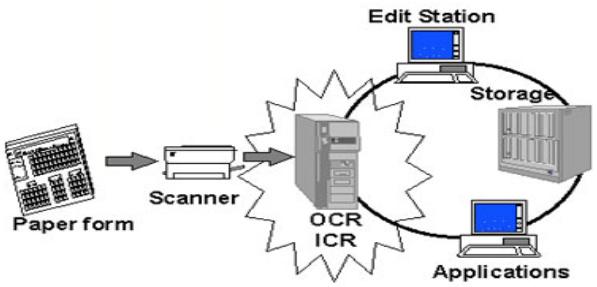


Fig. 1 General Diagram to represent OCR

Devnagari is used in many Indian languages like Hindi, Nepali, Marathi, Sindhi etc . Devnagari script is written in left to right and top to bottom format. [1]It consists of 11 vowels and 33 basic consonants. Each vowel except the first one have corresponding modifier that is used to modify a consonant. All words in Devnagari script have a continuous line of black pixels for whole word. This line is called "**Shirolekha**". Devnagari owes its complexity to its rich set of conjuncts. [2] Optical Character Recognition for Devanagari is fairly complex given its rich set of conjuncts. The language is partly phonetic in that a word written in Devanagari can only be pronounced in one way, but not all possible pronunciations can be written perfectly. A syllable ("akshar") is formed by a vowel alone or any combination of consonants with a vowel.

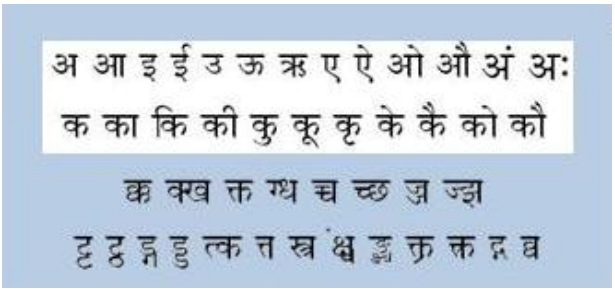


Fig. 2 some of the vowels and consonants with modifiers and compound characters

II. PROPOSED METHOD

In this section, we first present a brief overview of the basic Devnagari Segmentation technique and then will give details for embedding and retrieval of characters with example.

A word of Devnagari script is first of all segmented into composite characters and then each character is decomposed into set of symbols. A symbol may represent a composite Devnagari character, a modifier symbol – upper or lower, or a Devnagari alphabet.

[3]These decomposed symbols are recognized using the prototypes (explained later) and are composed for obtaining valid words. The symbols that can not be recognized as the valid symbols are rejection and substitution errors. During the training phase, we provide OCR with image and corresponding text. The OCR segments the image and extracts the prototype for the decomposed symbols for the recognition stage.

Devnagari word is written into the three strips namely: a core strip, a top strip, and a bottom strip as shown in Fig.ure 3.

The core strip and top strip are differentiated by the header, while the lower modifier is attached to the core character. We use height of the core characters to locate lower modifiers.

Fig. 3 Three strips of Devnagari word

OCR for Devnagari script becomes even more difficult when compound character and modifier characteristics are combined in 'noisy' situations. The image below illustrates a Devnagari document with background noise. We can clearly see that compound characters and modifiers are difficult to detect in this image because the image background is not uniform in color, and marks are present that must be distinguished from characters.

A. Swing Technology

Swing is a GUI toolkit for Java. Swing is one part of the Java Foundation Classes (JFC). Swing includes graphical user interface (GUI) widgets such as text boxes, buttons, split-panes, and tables.

Swing widgets provide more sophisticated GUI components than the earlier Abstract Windowing Toolkit. Since they are written in pure Java, they run the same on all platforms, unlike the AWT which is tied to the underlying platform's windowing system. Swing supports pluggable look and feel– not by using the native platform's facilities, but by roughly emulating them. This means we can get any supported look and feel on any platform. The disadvantage of lightweight components is possibly slower execution. The advantage is uniform behavior on all platforms.

B. JNI (Java Native Interface)

The Java Native Interface (JNI) is a powerful feature of the Java platform. Applications that use the JNI can incorporate native code written in programming languages such as C and C++, as well as code written in the Java programming language. The JNI allows programmers to take advantage of the power of the Java platform, without having to abandon their investments in legacy code.

Because the JNI is a part of the Java platform, programmers can address interoperability issues once, and expect their solution to work The JNI is a powerful feature that allows us to take advantage of the Java platform, but still utilize code written in other languages. As a part of the Java virtual machine implementation, the JNI is a two-way interface that allows Java applications to invoke native code and vice versa.

During the solution development the following softwares were used:

a) Microsoft Visual Studio
b) JDK1.6
c) Swings
d) JNI-Java Native Interface
e) Eclipse

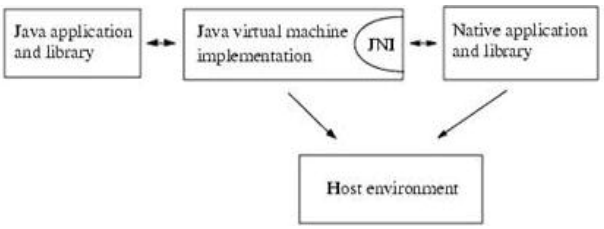


Fig. 4 Java Native Interface

C. Explanation with Steps

a) Binarization

Binarization (thresholding) refers to the conversion of a gray-scale image into a binary image. Image is converted into 1s and 0s form after binarization next step is segmentation.

b) Segmentation

In the segmentation, the input image is segmented into individual characters and then, each character is resized into *m* * *n* pixels towards the extracting the features. Segmentation in the context of character recognition can be defined as the process of extracting from the preprocessed image the smallest possible character units which are suitable for recognition. It consists of the following steps:

i. Locate the header line

An image is stored in the form of a two dimensional array in computer. A black pixel is represented by 1 and a white pixel by a 0. The array is scanned row by row and the number of black pixels is recorded for each row resulting in horizontal histogram. The row with the maximum number of black pixels is the position of the header line called as Shirolekha. This position is identified as *hLinePos.*

ii. Separate the character box

Characters are present below the header line. To identify the character boxes, we make a vertical histogram of the image starting from the *hLinePos* to boundary of the word i.e. the row where there are no black pixels. The boundaries for characters are identified as the columns that have no black pixels.

iii. Separate the upper modifier symbols

To identify the upper modifier symbols, we make a vertical histogram of the image starting from the top row of the image to the *hLinePos.*

iv. Separate the lower modifiers

c) Feature Extraction

Feature extraction refers to the process of characterizing the images generated from the segmentation procedure based on certain specific parameters.

d) Classification
Classification involves labeling each of the symbols as one of the known characters, based on the characteristics of that symbol. Thus, each character image is mapped to a textual representation.

e) Post Processing
The output of the classification process goes through an error detection and correction phase. This phase consists of the following three steps:

1) Select an appropriate partition of the dictionary based on the characteristics of the input word; select the candidate words from the selected partition to match the input word with.
2) Match the input word with the selected words.
3) In case the input word is found in the dictionary, no more processing is done and the word is assumed to be correct. If the word is not found, there are two options available. We can generate aliases for the input word or restrict to an exact match.

Finally put the whole scenario in terms of Fig.ure like,

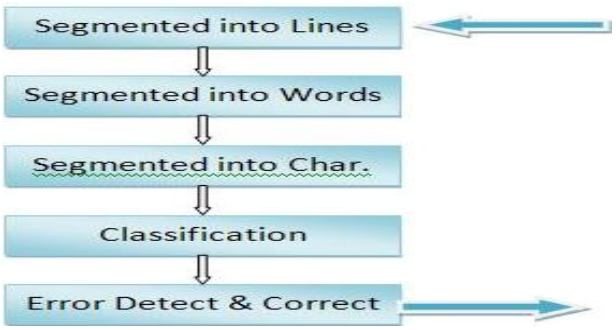


Fig. 5 Segmentation Steps

D. Design Approach

Modular approach has been taken into consideration. Designs are the determination of the modules and inter modular interfaces that satisfy a specified set of requirements. A design module is a functional entity with a well-defined set of inputs and outputs. Therefore, each module can be viewed as a component of the whole system, just as each room is a component of a house. A module is well defined if all the inputs to the module are essential to the function of the module and all outputs are produced by some action of the module. [6][7]Thus if one input will be left out, the module will not perform its full function. There are no unnecessary inputs; every input is used in generating the output. Finally, the module is well defined only when each output is a result of the functioning of the module and when no input becomes an output without having the transformed in some way by the module.

Modularity: Modularity is a characteristic of good system design. High level modules give us the opportunity to view the problem as whole and hide details that may distract us. [4]By being able to reach down to a lower level for more detail when we want to, modularity provides the flexibility , trace the flow of data through the system, and target the pockets of complexity.

These all are interrelated with each other and also self sufficient among themselves and help in running the system in an efficient and complete manner.

Level of Abstraction: Abstraction an information hiding allows us to examine the way in which modules are related to one another in the overall design the degree to which the modules are independent of one another is a measure of how good the system design is. Independence is desirable for two reasons.

[5]First it is easier to understand how a module works if its function is not tied to others. It is much easier to modify a module if it is independent of others. Often a change in requirements or in a design decision means that certain modules must be modified. Each change affects data or function or both. If the modules depend heavily on each other, a change to one module may mean changes module that are affected by the change.

Coupling: Coupling is a measure of how modules depend on each other. Two modules are highly coupled if there is a great deal of dependence between them. Loosely couple modules have no interconnection at all. Coupling depends on several things:

- ③ The references made from one module to another.
- ③ The amount of data passed from one module to another.
- ③ The amount of control one module has over the other.
- ③ The degree of complexity in the interface between one module and another.

Thus, coupling really represents a range of dependence, from complete dependence to complete independence. We want to minimize the dependence among modules for several reasons. First, if an element is affected by a system action, we always want to know which module causes an effect at a given time. Second, modularity helps in tracking the cause of the system errors. If an error occurs during the performance of particular function, independence of modules allows us to isolate the defective module more easily.

Cohesion: cohesion refers to the internal “glue” with which a module is constructed. The more cohesive a module, the more related are the internal parts of the module to each other and to the functionality of the module. In other words, a module is cohesive if all elements of the module are directed towards and essential for performing the same function.
For example the various triggers written for the Subscription entry form are performing the functionality of the module like querying the old data, saving the new data, updating records etc. So it’s a highly cohesive system.[8]

Scope of control and effect: Finally we want to be sure that the modules in our design do not affect other modules over which they have the control. The modules controlled by the given module are collectively referred to as the scope of effect. No module should be in scope of effect if it not in scope control.

Thus in order to make the system easier to construct, test, correct, and maintain our goals had been:

- ③ Low coupling of modules
- ③ High cohesive modules
- ③ Scope of effect of a module limited to its scope of control

It was decided to store data in different tables in SQL Server. The tables were normalized and various modules identified so as to store data properly create designed reports and on screen queries were written. A menu driven (user friendly) package has been designed containing understandable and presentable menus. Table structures are enclosed. Input and output details were made which are enclosed herewith.

The specifications in our design include

- ③ User interface Design screens and their description
- ③ Entity Relationship Diagrams

III. RESULTS AND DISCUSSION

In this section we will present some experimental results to demonstrate the effectiveness of our proposed technique. And we will also demonstrate various java packages and methods which must be required to perform the segmentation on Devnagari script. Different jpeg, gif, and png images are allowed by performing filtration on JNI.

A. Important packages

Import java.awt.*;// this package is a abstract window toolkit for applets design for interaction with user.

Import java.awt.event.*; //This package is supporting handled event are those generated by mouse, keyboard and other control such as push button etc.

Import javax.swing.*; //swing is a set of class that provide a more powerful and flexible component than in AWT.

Import javax.swing.JOptionPane; //It is a sub package of swing class which contain option panel.

Import java.io.*;//This package is used for INPUT from user and OUTPUT by program or console stream.

Import java.util.*; //This package contain some of the most exciting enhancement like : collection and t contain a wide assortment of classes and interface that support broad range of functionality.

Import java.awt.image.*; //This package use to support graphic images pictures.

B. Important Methods

The important methods which we have developed for performing segmentation on Devnagari script are:

Public int wordseg (int lineno, int w, int h, int vHisto [])
This above method is used for word by word segmentation

Public int lineseg (int w, int h, int hHisto [])
This above method is used for Line by Line segmentation horizontally

Public int hline (int ln, int wn, int w, int h, int hHisto [])
This above method is used for Line by Line selection horizontally

Public void ccharseg (int ln, int wn, int w, int h, int vHist)
This above method is used for vertically selecting single character segmentation.

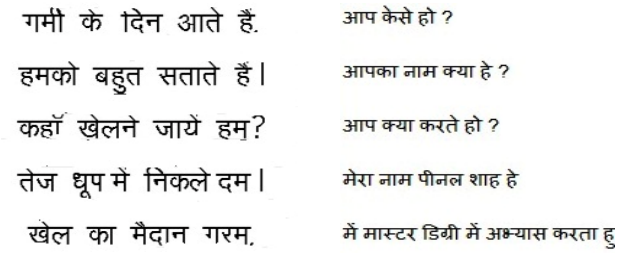
Some internal methods are also used to develop this segmentation software like:

Public Boolean accept (File f)
This function is internally used for the Filtering action

Public String getDescription ()
This function is internally used for the Filter Option drop down menu

C. Visual analysis

When you run the segmentation engine you will find very



effective user interface. We have tried this on some images and will show you result for one of the image which we have shown in Fig. 6.

Fig.. 6 Different sample images with different heights

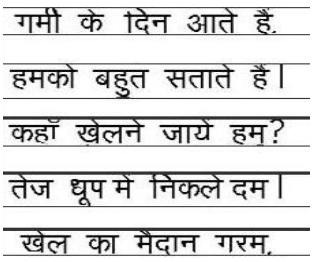


Fig. 7 Line Segmentation

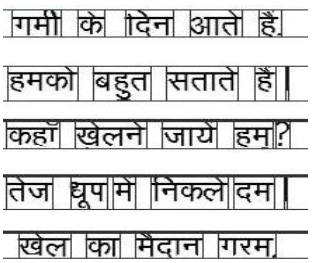


Fig. 8 Word Segmentation

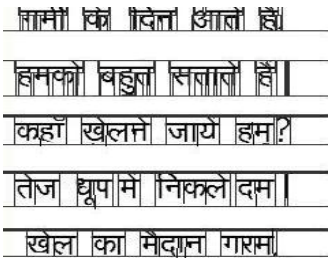


Fig.. 9 Character Segmentation

In the case of line segmentation and word segmentation this proposed method works perfectly even for printed text, English letters and digit numbers also.

For Example:

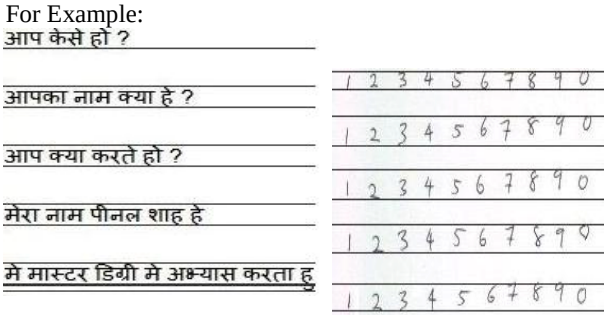


Fig. 10 Line Segmentation for different images

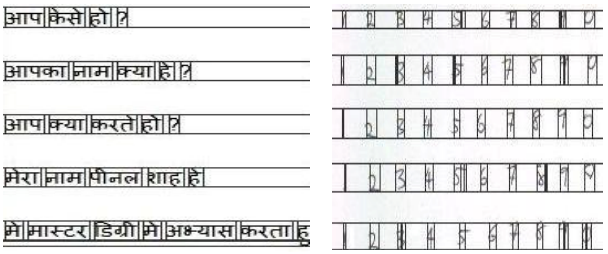


Fig. 11 Word Segmentation for different images

IV. CONCLUSION

A Devnagari document system has been developed which uses various knowledge sources to improve the performance. The composite characters are first segmented into its constituent symbols which help in reducing the size of the set, in addition to being a natural way of dealing with Devnagari script. The automated trainer makes two passes over the text image to learn the features of all the symbols of the script. A character pair expert resolves confusion between two candidate characters. The composition processor puts the symbols back together to get the words which are then passed through the dictionary. The dictionary corrects only those characters which cause a mismatch and have been recognized with low confidence.

The preliminary results on testing of the system show performance of more than 95% on printed texts on individual fonts. Further testing is currently underway for multi-font and hand printed texts. Most of the errors are due to inaccurate segmentation of symbols within a word. We are using only up to word level knowledge in our system. The domain knowledge and sentence level knowledge could be integrated to further enhance the performance in addition to making it more robust. The method utilizes an initial stage in which successive columns (vertical strips) of the scanned array are stored in groups of one pitch width to yield a coarse line pattern (CLP) that crudely shows the distribution of white and black along the line.

The CLP is analyzed to estimate baseline and line skew parameters by transforming the CLP by different trial line skews within a specified range. For every transformed CLP (XCLP), the number of black elements in each row is counted and the row-to-row change in this count is also calculated.

The XCLP giving the maximum negative change (decrease) is assumed to have zero skew. The skew corrected row that gives the maximum gradient serves as the estimated baseline. Successive pattern fields of the scanned array having unit pitch width are superposed (after skew correction) and summed. The resulting sum matrix tends to be sparse in the inter-character area. Thus, the column having minimum sum is recorded as an "average", or coarse, X-direction segmentation position. Each character pattern is examined individually, with the known baseline (corrected for skew) and average segmentation column as references.

A number of neighboring columns (3 columns, for example) to the left and right of the average segmentation columns are included in the view that is analyzed for full segmentation by conventional algorithm.

REFERENCES

[1] Paul and B.B. Chaudhuri, "Indian script character recognition: A survey," Pattern Recognition, vol. 37, no.9, pp. 1887-1899, 2004.
[2] Yi-Kai Chen and Jhing-Fa Wang, "Segmentation of Single-or Multiple-Touching Handwritten Numeral String Using Background and Foreground Analysis", IEEE PAMI vol.22, 1304-1317, 2000.
[3] J.Pradeep, E.Srinivasan, S.Himavathi "Diagonal Feature Extraction Based Handwritten Character System Using Neural Network" International Journal of Computer Applications (0975 – 8887) Volume 8– No.9, October 2010.
[4] H. Izakian, S. A. Monadjemi, B. Tork Ladani, and K. Zamanifar "Multi-Font Farsi/Arabic Isolated Character Recognition Using Chain Codes", World Academy of Science, Engineering and Technology 43 2008.
[5] Miguel Po-Hsien Wu, "Hand Written Character Recognition" Thesis for the degree of Bachelor of Engineering (Honors) In the division of Electrical Engineering October 29, 2003.
[6] Miguel Po-Hsiem wu, "Hand Written Character Recognition" Thesis for the degree of Engineering (Honors) In the division of electrical engineering October 29,2003.
[7] Robert Howard Kasse "A comparision approach for handwritten character Recognition System", Manchest Institute of Technology,, June 1995.
[8] Steven C. Elliot "Contest Sensitive Optical Character Recognition using neural network", Rochester Institute of Technology , Computer science department. February 13, 1992.

AUTHOR' S PROFILE

Pinal Shah received the B.E. in Computer Engineering degree from Sardar Patel University, India in 2009.

He is currently a Master of Engineering student in Shantilal Shah Engineering College, Bhavnagar, Gujarat, India. His research interests include digital forensics, cryptography, Image Processing (Optical Character Recognition and Face detection Technology).

Prof. Preeti K Dave received the B.E. in Electronics and Communication degree from Bhavnagar University, India in 2000 and M.E in Electrical (Microprocessor application and System) from M.S University, Vadodara, India in 2009.

Her research interests include Image Processing, Remote Sensing, Digital Image Processing.