Segmentation of Touching Characters in Hand-written Devanagari Script

Mr. Dipak V. Koshti
Student, ME[COMP]
Pillai's Institute of Information Technology
New Panvel
Navi Mumbai, India
dipak.koshti@ymail.com

Mrs. Sharvari Govilkar
Assistant Professor, Computer Dept.
Pillai's Institute of Information Technology
New Panvel
Navi Mumbai, India
g_sharvari@rediffmail.com

Abstract- **In this paper, we propose the method of segmentation only for touching characters for Handwritten Marathi Text that is the Devanagari script. There may be some touching characters in the word. Sometimes, interline space and noise makes line segmentation a difficult task. The main purpose of this paper is to provide the new segmentation technique based on joint point algorithm. Foreground and background information are used here for accurate segmentation. Our method can take care of this situation accurately. Segmentation is one of the important step of character recognition systems. . It is an important step because inaccurately segmented text lines will cause errors in the recognition stage.**

Keywords**—joint point algorithm, touching characters, handwritten, devanagari script.**

I. INTRODUCTION

Handwritten character recognition is an important field of Optical Character Recognition (OCR). The objective of OCR is automatic reading of optically sensed document text materials to translate human-readable characters to machine-readable codes. A good survey about the OCR is given in [1] Research in OCR is popular for its various application potentials in banks, library automation post-offices and defense organizations.

Segmentation is a technique which partitions handwritten Marathi words into individual characters. Since recognition heavily relies on isolated characters, segmentation is a critical step for character recognition because better is the segmentation, lesser is the ambiguity encountered in recognition of candidate characters of word pieces. Handwritten character segmentation is difficult task because of different writing style of person. The handwritten characters do not have a fixed size and shape. So they are quite different from the printed characters. In the case of printed characters, vertical bar occupies a single column whereas handwritten characters might occupy more than one column. Moreover the header line is never straight. Position of the header line in all the Devanagari word is not same. So it can be removed from all the characters at the time of pre-processing. The position of the header line in printed words of Indian script is found using the horizontal pixel projection profiles. This technique does not work for the handwritten words in which the header line covers multiple rows instead of a single row as in the printed words. This is true of handwritten characters. A selective algorithm is developed for the identification and removal of header line and for further segmentation from the handwritten Devanagari characters. The algorithm takes care of slant in the header line as well as end bar lines. Proposed system gives promising results for printed as well as handwritten text.

II. CHARACTERISTICS OF MARATHI LANGUAGE

Marathi, Hindi, Sanskrit and Nepali languages comes under Devanagari script. It is the most popular script in India. It has 12 vowels and 35 consonants as shown in fig.1. Vowels can be written as independent letters, or by using a variety of diacritical marks which are written above, below, before or after the consonant they belong to. When vowels are written in this way they are known as *modifiers* and the characters so formed are called *conjuncts* as shown in fig 2. Sometimes two or more consonants can combine and take new shapes. These new shape clusters are known as *compound characters* as shown in fig 3. These types of characters namely basic characters, compound characters and modifiers are present not only in Devanagari but also in other scripts. All the characters have a horizontal line in the upper part, known as *Shirorekha* or header line. No English character has such a characteristic and so it can be taken as a distinguishable feature to extract English from these scripts. In such scripts, a text word may be partitioned into three zones. The upper zone denotes the portion above the headline, the middle zone covers the portion of basic and compound characters below the headline and the lower zone that may contain some vowel and consonant modifiers. The imaginary line separating the middle and lower zone may be called the base line.

A typical zoning is shown in Fig.4.
In Marathi we have three types of characters as follows.
1.END-BAR Characters
2.MIDDLE-BAR Characters
3.NO-BAR Characters
All three types are shown in Fig 5a, Fig 5b and Fig 5c respectively.

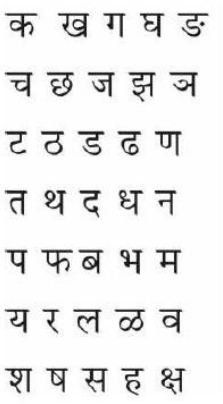


Fig 1. Consonants.

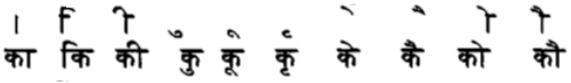


Fig 2.Modifiers(Ascenders and descenders)

क्य ख्य घ्य च्य ज्य त्य थ्य ध्य न्य प्य भ्य म्य य्य ल्य व्य

Fig 3.Compound Characters

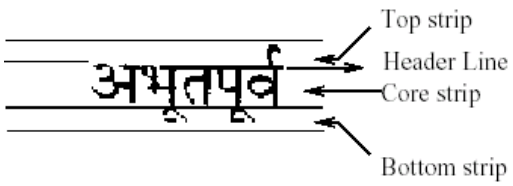


Fig 4.Three strips of a Devanagari word.

अ स ष च ज झ म त थ

Fig 5a. END-BAR Characters

ऋ क फ

Fig 5b. MIDDLE-BAR Characters

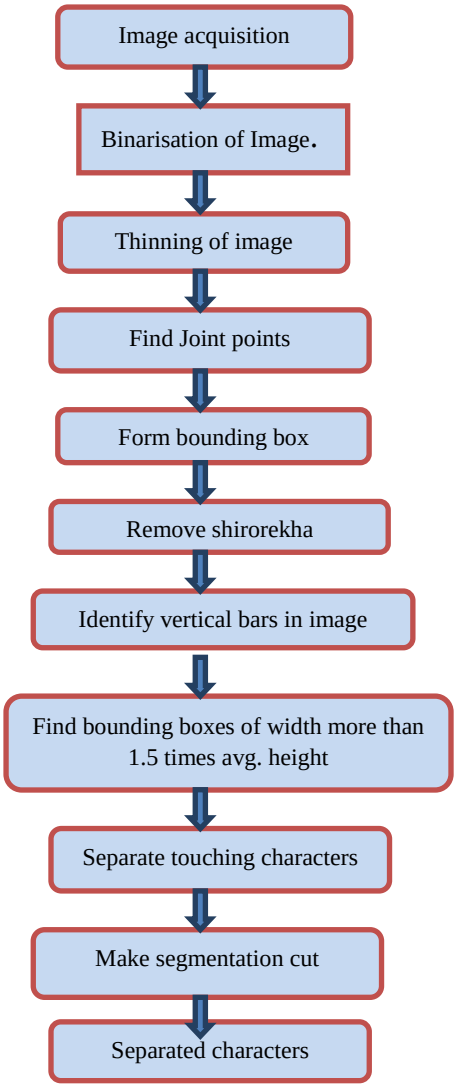
इ उ ऊ ए छ ट ठ द ड ढ

Fig 5c. NO-BAR Characters

II. Segmentation

The basic character set of Devanagari script is very large comprising 12 vowels and 35 consonants. Characters of handwritten words do not have a fixed size and shape. So they are quite different from the printed characters. In the case of printed words, vertical bar of a character occupies a single column whereas that of handwritten words might occupy more than one column. For segmentation of printed words, statistical information is used. The statistical information is also used for classifying the printed conjunct characters into consonants and half consonants. The methods shown in [2-5] of printed words fail to work on the handwritten words.

III. Proposed Algorithm



Structural Analysis to Find the Cut Position

It is found that on the basis of structural properties of the devanagari script, various touching characters can be classified among few categories. Some characters also fall in multiple categories. For each pair of touching characters, these categories are defined on the basis of left character of the pair. These categories are now briefly described.

1) Category 1:

Touching characters containing sidebars at right end. By carefully analyzing, it is found that many of the total pair of touching characters contains these characters at left side, which have sidebars at their right end.

2) *Category 2:*

Half character touching to full characters containing sidebars at right end.

3) Category 3:

Find pattern between two vertical bar touching characters. One convex and one concave touching pattern.

1. Image acquisition:

This involves scanning a document and storing it as an image. This is used as input to the program. Higher resolution produces better results. The resolution (number of dots per inch, dpi) affects the accuracy of process of recognition.

Database creation is an important task for testing of the proposed system. Since the aim is to handle handwriting, it is important to ensure that the system handles variations in the handwriting at various levels like style, quality and paper used. Hence the need to create multiple samples with all kinds of variations at all levels. 5 persons were asked to write 5 words each. They were encouraged to write samples in varying styles. Then the handwritten samples are scanned.

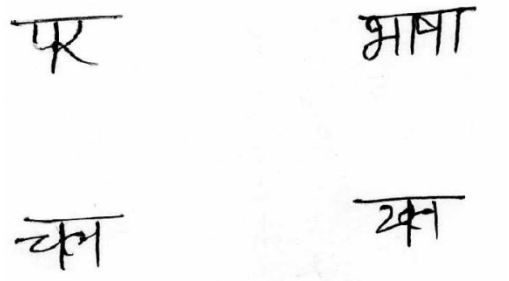


Fig.6 shows the samples of handwritten Devanagari characters

2. Binarization of Image:

In a binary image, each pixel assumes one of only two discrete values: 1 or 0. We binarize the image to separate the background and foreground. The handwriting to be recognized is present in the foreground. Also the foreground is represented with 1 as most functions in the programming language consider it to be so. A binary image is stored as a logical array which needs less number of storage space. By convention, this documentation uses the variable name BW to refer to binary images.

The input image is first binarized and complemented so that the background is black and foreground i.e. the words are white and are represented as 1's. Connected components are found in the binary image to isolate each word separately and segmentation process has to concentrate on a smaller regions.

Dilated Image: The words are dilated to allow formation of bounding boxes only around words. In this way words can be isolated for further processing.

Morphology is a broad set of image processing operations that process images based on shapes. Morphological operations apply a structuring element to an input image, creating an output image of the same size. In a morphological operation, the value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbours. By choosing the size and shape of the neighbourhood, we can construct a morphological operation that is sensitive to specific shapes in the input image.

The most basic morphological operations are dilation and erosion. Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. The number of pixels added or removed from the objects in an image depends on the size and shape of the *structuring element* used to process the image.

To dilate the image, pass the image BW and the structuring element SE to the imdilate function. Dilation adds a rank of 1's to all sides of the foreground object.

BW2 = imdilate(BW,SE)

The bounding boxes of each word is found after dilating the binarized image and finding its bounding

The heights of the characters is first found by finding the heights of the bounding boxes. The height is the only dimension we can use at this stage as it varies less than the width from word to word.

Fig 7.Dialated Image.

Properties of image regions to be computed.

'BoundingBox' — The smallest rectangle containing the region, a 1-by-Q *2 vector, where Q is the number of image dimensions: ndims(L), ndims(BW), or numel(CC.ImageSize). BoundingBox is [ul_corner width], where:

ul_corner is in the form [x y z ...] and specifies the upper-left corner of the bounding box

Width is in the form [x_width y_width ...] and specifies the width of the bounding box along each dimension

Fig 8. Binarized Image

Small black patches are removed that present in the binarized image. This is done because small black patches create

circles in the thinning process which can further confuse the system. The system may wrongly consider the circles as structural parts of the characters. Bounding boxes are formed around small black patches and converted into white.

Protruding points

1.Protruding points are bulging pixels which unnecessarily generate joint points after thinning where no joint actually exists. These must be removed to generate accurate joints points where joints between lines actually exist.
2.The sum of three by three matrix (minus centre pixel) around each pixel is found. If this sum is less than or equal to 3 then the protruding pixel is removed. This process is visualized in the figure.
3.Image with protruding points highlighted with red.

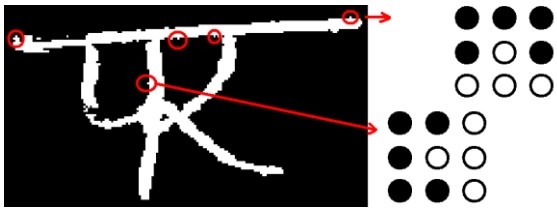


Fig 9.Removal of Protruding points eliminate useless joint points in the image.

3. *Thinned Image*

Thinning removes unnecessary pixels and makes further processing possible.

Thinning is a morphological operation that is used to remove selected foreground pixels from binary images, It reduces all lines to single pixel thickness. Thinning is normally only applied to binary images, and produces another binary image as output. The image is thinned so that the thickness of the lines and curves in the characters becomes one pixel. This reduces the number of pixels we need to handle and process. The thinning process does not remove the structural features of the words and characters but represents them with minimum number of pixels. The joint points can be found only in the thinned image as the lines around joints are of 1 pixel thickness. With n = Inf, the thinning happens to the greatest possible extent. It removes pixels so that an object without holes shrinks to a minimally connected stroke, and an object with holes shrinks to a connected ring halfway between each hole and the outer boundary.

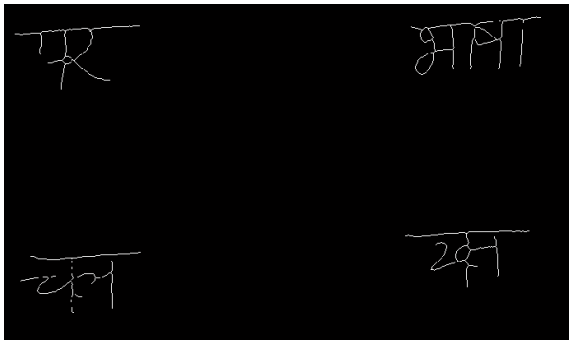


Fig 10. Thinned Image.

4. The joint points

1. The joint points are points where two or more lines meet. The lines can be straight or curved. Using these points we can dissect parts of character into separate lines depending on whether they are curved or straight. However here we use them only to retrieve vertical bars and header lines.
2. The joint points are found by analysing the pattern of pixels in 3 x 3 Matrix surrounding each pixel.
3. Joints points are important as they are the points where more than one line meet. Its possible to dissect each and every character into its individual curves and straight lines for further processing. Also the joint points have to be precise as less or more of them can confuse further stages of the system. To find vertical bars these joint points are removed to allow slim bounding boxes to be formed around the vertical bars.
4. The sum of 3 x 3 matrix (minus centre pixel) around each pixel is found. If this sum is equal to or more than 3 then the pixel is considered as a joint point. This is so because the sum is 3 only when the joint point has three lines meeting at it.
5. Also the terminating points are found which have only 1 pixel in the surrounding. This pixel comes from the only line joining the point.

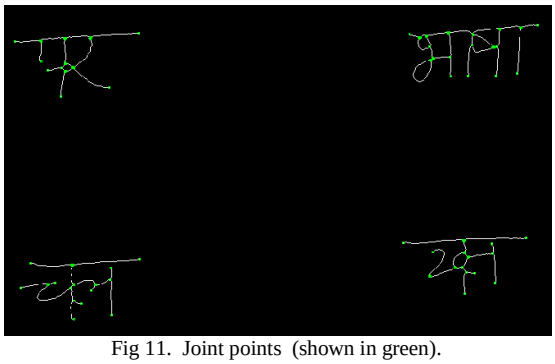


Fig 11. Joint points (shown in green).

Next the joint points are removed from the thinned image to allow for bounding boxes to be developed around smaller lines. Of these slim vertical rectangles are mostly parts of the vertical bars and slim horizontal lines are parts of the header lines.

5. Form Bounding Box

Bounding boxes are formed to find segments of each character in each word to make way for identification of header line and vertical bars. Within each word horizontal rectangles are identified. The rectangles are obtained by finding the bounding boxes of words in the thinned image after removing the joint points.

6. Removal of Shirorekha

1. The first aim is to find and remove shirorekha.
2. Rectangles are obtained by finding the bounding boxes of words in the thinned image minus joint points.
3. Within each word horizontal rectangles i.e. bounding boxes are identified.
4. Horizontal rectangles are rectangles having width-height ratio more than or equal to 3:1.

5. Heights of all pixels within these rectangles are averaged and rectangles having most number of pixels on one side of this averaged line are considered as parts of shirorekha.
6. These candidate rectangles are removed to remove the shirorekha**.**

The location of shirorekha is stored for future use in detection of vertical bars. To identify the vertical bars the vertical rectangles are Identified. These vertical rectangles are extended up to the position of the shirorekha at top and up to the bottom of the bounding box at bottom.

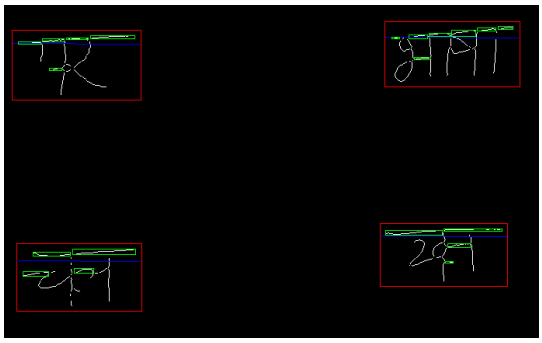


Fig 12.Horizontal Bounding Boxes.

7. The vertical bar detection

1. To identify the vertical bars, the vertical rectangles are identified.
2. Vertical rectangles are rectangles having width-height ratio less than or equal to 1:3.
3. The vertical rectangles are extended up and down and percentage of white pixels (which are parts of vertical bars) is found. If this percentage is more than 60% then the extended rectangle contains a vertical bar.
4. A new image is taken and the rectangles are extended vertically at up to the level of shirorekha at the top and up to a level at the bottom.
5. Bounding boxes are formed in the new image.
6. Bounding boxes having more than 60% rows of at least one white pixel are identified as bounding box of a vertical bar.

8. Cutting of Joint Characters

1. Characters within a word may be joined due to imperfect handwriting. These can be separated using the vertical bar in case of side bar characters. For middle bar characters further processing is required for dissection.
2. The vertical bar serves as the cut line that can cut joint characters.
3. A cut is made after each vertical bar.
4. The algorithm shows good results as compared to other methods

IV. Conclusion & Future work

We have proposed an algorithm specially for touching characters and hope it gives promising results as compared to other techniques used for segmentation of handwritten as well as printed document images. In handwritten documents, segmentation of touching characters is very difficult task, our algorithm works well to segment touching characters. The segmentation problem occurs when there is overlapping of ascenders and descenders on two consecutive lines. The same work can be utilized to work with other types of documents such as Hindi, Marathi and other Devanagari languages.

The algorithm can be modified to deal with overlapping ascenders and descenders of two consecutive lines as well as two characters overlapping on each other in the same line.

Acknowledgment

I remain immensely obliged to Prof. Mrs. Sharvari Govilkar, for providing me with the idea of this topic, and for her invaluable support in garnering resources for me.

References

[1] S. Mori, C. Y. Suen and K. Yamamoto, "Historical review of OCR Research and development", Proceedings of the IEEE,Vol. 80(7), pp. 1029-1058, 1992.

[2] Segmentation of Printed Text in Devanagari Script and Gurmukhi Script Vijay Kumar Pankaj K. Sengar International Journal of Computer Applications (0975 – 8887)Volume 3 – No.8, June 2010

[3] Segmentation of Touching Characters in Printed Devanagari and Bangla Scripts Using Fuzzy Multifactorial Analysis Utpal Garain and Bidyut B. Chaudhuri

[4] An Improved Method for Segmentation of Touching Symbols in Printed Mathematical Expressions Dong-Yu Zhang1, 2, Xue-Dong Tian1, 2, and Xin-Fu Li1, 2

[5] SEGMENTATION OF TOUCHING AND FUSED DEVANAGARI CHARACTERS -Veena Bansal and R. M. K. Sinha

[6] An Iterative Algorithm for Segmentation of Isolated Handwritten Words in Gurmukhi Script.

[7] N.Madan , Sunil Madan and H.Singh, " Hybrid Approach to character segmentation of Gurmukhi Script characters", Proceedings of the 32nd applied imagery pattern recognition workshop,2003

[8] M. K. Jindal, G.S. Lehal and R.K. Sharma, "A Study of Touching Characters in degraded Gurmukhi Script", International Conference on Pattern Recognition and Computer Vision, PRCV 2005, 25-27 February 2005, Istanbul, Turkey.