Handwritten Hindi Text Segmentation Techniques for Lines and Characters

Saiprakash Palakollu Renu Dhir Rajneesh Rani

Abstract—**This paper mainly deals with the new methods for line segmentation and character segmentation of overlapping characters of Handwritten Hindi text. The text is segmented into lines, lines into words and then from lines words header lines are detected and converted as straight lines. Each word is divided into three parts upper modifier, consonant and lower, so that character segmentation becomes easy. Algorithm is finding the header lines and base lines by estimating the average line height and based on it. This algorithm works efficiently on overlapped characters for different text sizes and different resolutions images.**

Index Terms—**Line, word and character segmentation, projection, average line height, header line, base line detection.**

I. INTRODUCTION

Handwritten character recognition is difficult task compared to machine printed character recognition in the area of Optical Character Recognition. Devanagari is a script for Hindi text. A lot of research is done on the printed Hindi text, but less work has been done on the handwritten Hindi text recognition. It has three major steps such as pre-processing, segmentation and recognition. Segmentation is the important step. Segmentation also contains three major steps such as line segmentation, word segmentation and character segmentation. If we fail in doing line segmentation then entire segmentation process goes wrong. A lot of research has been done in the past on line segmentation of handwritten text. A wide variety of line segmentation methods for handwritten documents are reported in the literature. Some of the important methods for line segmentation are projection based method [1] and [2], Hough transform based method [3], smearing method [4], grouping method [5], graph based method [6], CTM (Cut text Minimum) approach [7], Block covering method [8], linear programming method [9] and curve based [15].

Saiprakash Palakollu has completed his M. Tech from the Department of Computer Science and Engineering, NIT Jalandhar, Punjab in June, 2011. (E-mail: saiprakash258gm@gmail.com)
Renu Dhir is associate Professor in the Department of Computer Science and Engineering, NIT Jalandhar, Punjab. (E-mail: dhirr@nitj.ac.in)
Rajneesh rani is assistant Professor in the Department of Computer Science and Engineering, NIT Jalandhar, Punjab. (E-mail: ranir@nitj.ac.in)

The projection based methods are successful in the case of straight and easily separable lines only. A lot of research is going on how to detect overlapped lines and characters. The method which is based on header line and base lines detection and average line height is assumed as 30 pixels gives good results in case of fixed resolution images. The main purpose of this paper is estimating the average line height and based on it, finding the header lines and base lines.

Word segmentation is very easy because gap between words is generally more, so we can separate them easily. But in Character segmentation most of the researchers use vertical projection method [11]. Some researchers use Hidden Markov model [13][14]. But these methods do not work well for the overlapping characters.

In next Section, we have discussed the characteristics of Hindi language. In Section 2, segmentation techniques used for segmenting the handwritten Hindi text have been discussed. Finally, Section 3-4 contains experimental results and discussions.

II. CHARACTERISTICS OF HINDI LANGUAGE

Devnagari is most popular script to write Hindi as well as Sanskrit, Marathi, Sindhi, and Nepali language with minor modifications. The alphabets of Devanagari script consists of 33 consonants and 14 vowels. There is no concept of lower or upper case in Hindi language. In Devnagari script, 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 may contain where some vowel and consonant modifiers can reside. For a long number of characters (basic as well as compound) there exists a horizontal line at the upper part called "*shirorekha*" or headline in Hindi. All the characteristics with detailed explanation and images are given by Raghuraj Singh [10].

The imaginary line separating the middle and lower zone may be called the base line. In Hindi language characters also have a half form which increases the language complexity for recognition. The half characters may touch with full characters to make the characters called conjuncts. Two consecutive lines touches or overlap each other due to these modifiers. This makes the segmentation of handwritten Hindi text very complex.

III. SEGMENTATION

Segmentation is one of the most important phases in character recognition process. Segmentation is the process of segmenting the whole document image into recognizable units.

The segmentation process is divided into four major parts.

i. Line segmentation
ii. Word segmentation
iii. Zone segmentation
iv. Character segmentation

A Line Segmentation

First step of segmentation process is segmenting the text region into lines, also called as line segmentation.

First we need to calculate header lines and base lines for the Line segmentation as shown in the figure 1a, 1b. Header lines are rows with maximum number of black pixels and base lines are rows with minimum number of black pixels. Finding header line is a challenge because of skew in headline. Till now most of the researchers are detecting the header line by finding the row with maximum pixel density, but it cannot work for skew variable text. The algorithm proposed by M. K. Jindal helps to find header line and base line [16] [17].

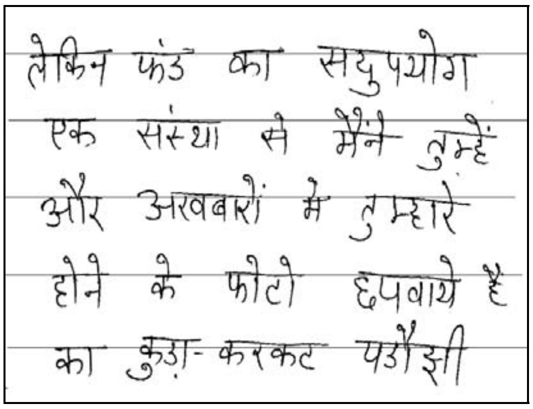

लेकिन फंड का सदुपयोग
एक संस्था से मैंने तुम्हें
और अखबरों में तुम्हारे
होने के फोटो छपवाये हैं
का कुड़ा-करकट पडौझी

Figure 1: Header lines

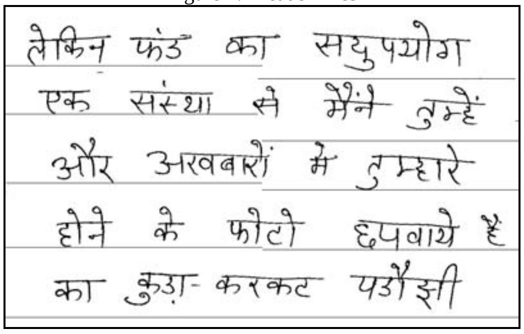

लेकिन फंड का सदुपयोग
एक संस्था से मैंने तुम्हें
और अखबरों में तुम्हारे
होने के फोटो छपवाये हैं
का कुड़ा-करकट पडौझी

Figure 2: Base lines

This method gives good results for uniform and non-uniform skewed lines. But average line height is assumed to be 30 pixels. As different users handwritings are different, so average line height of users may be greater than 30 pixel or less than 30 pixels. Average line height of high resolution images is greater than low resolution images. If 450×540 image has average line height as 30 pixels, 900×1080 image average line height will be 60 pixels, 225×270 image average line height will be 15 pixels. In the above two situations this algorithm fails to detect the lines accurately.

The new algorithm can work for the images having different average line height. In this algorithm we need to find average line height using horizontal projection based method. Divide the Image into three equal halves (stripes), because the skew in entire line may be high, as compared to skew in the first half. Perform the following steps in first half.

1) Find out the rows with minimum number of pixels and replace that row pixels with black pixels .So that we can separate text lines with black rows.
2) Find out the height of the text lines using those black rows.
3) Store the heights of lines in array and use sorting technique to sort the elements and take median as average height of the lines.
4) By using average line height calculate minimum height of the consonant in a line.
   Minimum height of the consonant = Average line height / 4.

B Word Segmentation

Word segmentation is easier than line segmentation and character segmentation. Space between two words is generally more than three pixels. Words are segmented by the projection based method.

C Zone Segmentation

Main problem in zone segmentation is separating header line from word. In handwritten text header line is not straight line as shown in the figure 3. To solve this problem we trace the header line and convert it into straight line by using following algorithm. In this algorithm we estimate header line row as shown in the figure 4 and compare the difference between actual header line and estimated header line. After that actual header line pixels will be shifted towards estimated one. After applying the following algorithm curved line becomes straight line as shown in figure 5.

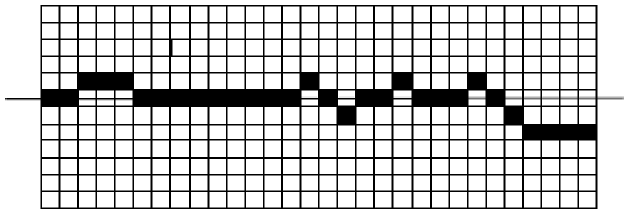


Figure 3: Curved header line

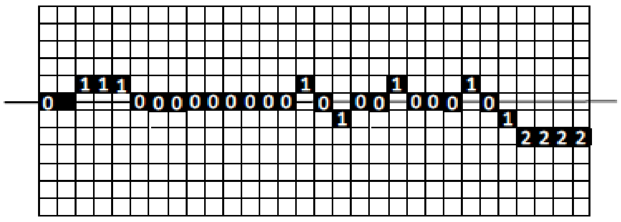


Figure 4: Representation of difference between estimated header line and actual curved header line

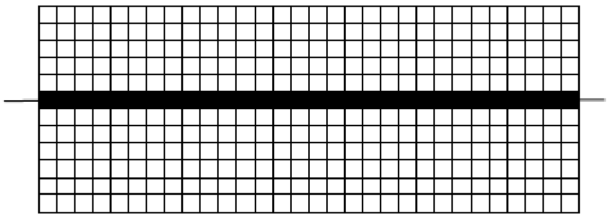


Figure 5: Straightened header line after applying the algorithm.

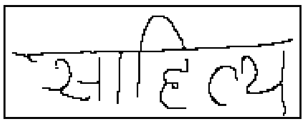


Figure 6: Image of word before applying algorithm 3.4

Algorithm for straitening header line of a word after performing thinning operation on binary image.

1. Identify the most visible line in first 50% area and consider it as expected header line.
2. Find out a point where expected header line and actual header line meets.
3. From that point trace the actual header line and find out the differences between actual header line and expected header line on both sides.
    a. $Diff_j$= expected header line –actual header $line_j$, i.e., j – column
4. If $diff_j$ =0 column remains same.
5. If $diff_j$>0 move the entire column upside according to $diff_j$ value.
6. If $diff_j$>0 move the entire column downside according to $diff_j$ value.

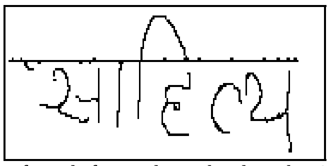


Figure 7: Image of word after applying the algorithm 2.3.1 on figure 6

After straightening the header line we separate upper modifier, consonant and lower modifier. To separate consonant and lower modifier we use horizontal projection profile and finds the row with maximum number of black pixels (header line).

D Character Segmentation

Main problem in character segmentation is separating header line from word. In handwritten text header line is not straight line. To solve this problem we trace the header line and convert it into straight line by using following algorithm.

1. Perform thinning operation on binary image.
2. Identify the most visible line in first 50% area and consider it as expected header line.
3. Find out a point where expected header line and actual header line meets.
4. From that point trace the actual header line and find out the differences between actual header line and expected header line on both sides.
5. $Diff_j$= expected header line –actual header $line_j$, i.e., j – column
6. If $diff_j$ =0 column remains same.
7. If $diff_j$>0 move the entire column upside according to $diff_j$ value.
8. If $diff_j$>0 move the entire column downside according to $diff_j$ value.

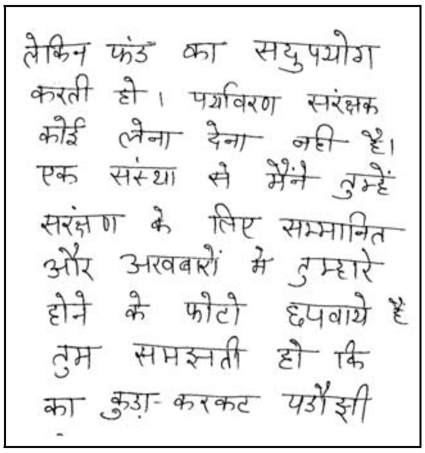
After straightening the header line we separate upper modifier, consonant and lower modifier. To separate consonant and lower modifier we use horizontal projection profile.

Following algorithm for character segmentation is useful in the case of overlapping characters also.

1. Read all the pixels intensities (0 or 255) in the binary image and store them in a 2 dimensional matrix.
2. Assume i and j represents the row and column position of the pixel. $P_{i,j}$ represents intensity of pixel from i^{th} row and j^{th} column. Where i=1, j=5;
3. If $P_{i+1, j}$ equals to 255 then save it as next position in path. $P_{i, j}$= $P_{i+1, j}$ repeat step 3. Otherwise go to step 4.
4. If $P_{i, j+1}$ equals to 255 then save it as next position in path. $P_{i, j}$= $P_{i+1, j}$ go to step 3. Otherwise go to step 5.
5. If $P_{i, j-1}$ equals to 255 then save it as next position in path. $P_{i, j}$= $P_{i+1, j}$ go to step 3. Otherwise go to step 6.
6. Stop the process and check whether the path leads to the end row or not. If yes, draw the path. Increment j by 5 (j=j+5), i=1 and start from step 3.

IV. EXPERIMENTAL RESULTS

Experiments for line segmentation are carried out with images which are scanned from the handwritten Dataset. Handwritten data is collected from the 30 different writers from various backgrounds such as students, teachers, doctors, lawyers and business men. Data of different sizes, i.e. writer's handwriting is different so the size of characters varies from 15-60 pixels and different resolutions. As data is collected from the people from various backgrounds, so the database works like real database. This database contains 29 pages and 500 lines. Number of lines per page varies from page to page. Figure 8 contain part of handwritten Hindi database.

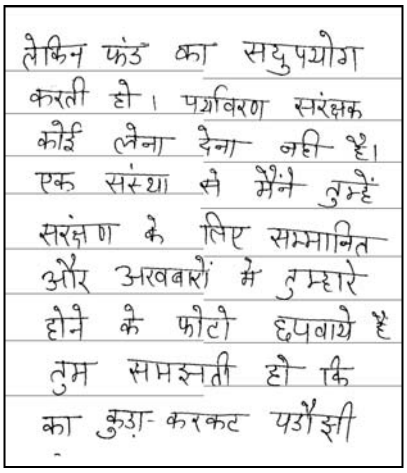


लेकिन फंड का सदुपयोग
करती हो। पर्यावरण संरक्षक
कोई लेना देना नहीं है।
एक संस्था से मैंने तुम्हें
संरक्षण के लिए सम्मानित
और अखबारों में तुम्हारे
होने के फोटो छपवाये हैं
तुम समझती हो कि
का कूड़ा-करकट पड़ौसी

Figure 8: Part of Database

The accuracy of line segmentation depends upon the accuracy of header line and base line detection. If we assume average line height as 30 pixels, then in high resolution images lines are not detected accurately, because in high resolution images number of pixels used per line is high.

To solve this problem we are using the new technique for estimating average line height and lines are detected correctly as shown in figure 9. It works for high resolution images and different size characters.



लेकिन फंड का सदुपयोग
करती हो। पर्यावरण संरक्षक
कोई लेना देना नहीं है।
एक संस्था से मैंने तुम्हें
संरक्षण के लिए सम्मानित
और अखबारों में तुम्हारे
होने के फोटो छपवाये हैं
तुम समझती हो कि
का कूड़ा-करकट पड़ौसी

Figure 9: Result of segmentation (Average line height is estimated using algorithm).

For word segmentation we have used vertical projection method because gap between any two words is naturally good enough to use projection method, it gives good results as shown in the following figure 10.

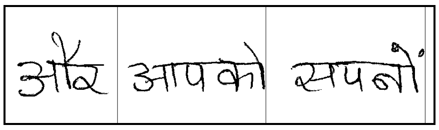


Figure 10: Correctly segmented words.

This method for straitening the header line works accurately and after applying the algorithm header line becomes as shown in the figure 11.

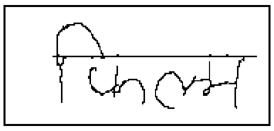


Figure 11: Word image after straighten header line.

After straightening header line we can separate upper modifier easily and generate three images consisting consonant, header line, upper modifier separately shown in the following figure 12.

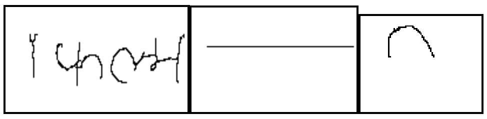


Figure 12: Images after separating upper modifier and consonant

After generating word which has only consonant part we apply the proposed algorithm for character segmentation. It generates stripes after every 5 pixels, if any character is in path it converts its path as shown in the figure. So that we can segment overlapped characters also, an example of segmentation of overlapped characters is shown in the following figure 13.

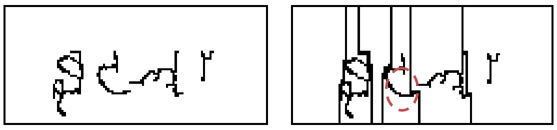


Figure 13: Image before segmentation and correctly segmented overlapping characters.

Table I Accuracy of Line segmentation

Number of lines	Lines correctly segmented	% of Accuracy
500	468	93.6

Table II Accuracy of Word segmentation

Number of words	Words correctly segmented	% of Accuracy
1500	1480	98.6

Table III Accuracy of Consonants

Number of Consonants	Consonants correctly segmented	% of Accuracy
4478	4026	89.90

V. DISCUSSIONS

From the above tables I, II and III we can say that new methods are giving good results. The new method for line segmentation is working efficiently in the cases of different text sizes and different resolution images. Second method for character segmentation is also working efficiently in the case of overlapping characters as shown in the figure 7. The method which is used for straightening header line is working fine. These methods are also applicable for printed Hindi text. The lines which have broken parts in upper modifiers are not correctly recognized as shown in figure 8. The lines with thick parts in upper modifiers also not correctly recognized. Touching lines are not correctly recognized. For applying contour technique we need estimate header lines and base lines correctly, this method gives good results in finding those.

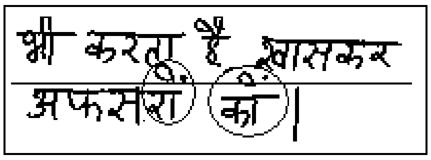


Figure 14: Wrongly segmented lines with broken parts in upper modifiers.

In future the study may be carried out on following direction:

- The above method can be applied on other languages.
- Some other technique may be tried for segmentation of overlapped or touching lines within a stripe.
- Segmentation of lines with broken parts is not done yet. It may be carried out in the future.
- Character segmentation technique mentioned above does not work for touching characters. So character segmentation method can be changed to improve accuracy.

REFERENCES

[1] B. M. Sagar, G. Shobha, P. Ramakanth kumar, “Character Segmentation Algorithms for Kannada optical character recognition”, Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition, 2008.

[2] Seong-whan Lee, Dong-June Lee, Hee-Seon Park, “A new Methodology for Gray-Scale Character Segmentation and Recognition”, IEEE transactions on pattern analysis and machine intelligence, 1996.

[3] Satadal Saha, Subhadip Basu, Mita Nasipuri and Dipak Kr. Basu, “A Hough Transform based Technique for Text Segmentation”, journal of computing, 2010.

[4] Y. Li, Y. Zheng, D. Doermann, and S. Jaeger, “A new algorithm for detecting text line in handwritten document”, in the proceedings of International Workshop on Frontiers in Handwriting Recognition, 2006.

[5] L. Likforman-Sulem and C. Faure, “Extracting text lines in handwritten documents by perceptual grouping”, Advances in handwriting and drawing : a multidisciplinary approach, C. Faure, P. Keuss, G. Lorette and A. Winter Eds, Europia, Paris, 1994.

[7] I.S.I. Abuhaiba, S. Datta and M. J. J. Holt, “Line Extraction and Stroke Ordering of Text Pages”, in the Proceedings of Third International Conference on Document Analysis and Recognition, Montreal, Canada, 1995.

[8] C. Weliwitage, A. L. Harvey and A. B. Jennings, “Handwritten Document Offline Text Line Segmentation”, in the Proceedings of Digital Imaging Computing: Techniques and Applications, 2005.

[9] A. Zahour, B. Taconet, L. Likforman-Sulem and Wafa Boussellaa, “Overlapping and multi-touching text-line segmentation by Block Covering analysis”, Pattern analysis and applications, 2008.

[10] Raghuraj Singh, C. S. Yadav, Prabhat Verma, “Optical Character Recognition (OCR) for Printed Devnagari Script Using Artificial Neural Network”, International Journal of Computer Science & Communication, 2010.

[11] Naresh Kumar Garg, Lakhwinder Kaur, M. K. Jindal, “Segmentation of Handwritten Hindi Text”, in the International Journal of Computer Applications, (0975 – 8887), 2010.

[12] Naresh Kumar Garg, Lakhwinder Kaur, MK. Jindal, “A New Method for Line Segmentation of Handwritten Hindi Text”, Seventh International Conference on Information Technology, 2010.

[13] Bikash Shaw, Swapna kumar parui, malayappan, “A Segmentation Based Approach to Offline Handwritten Devanagari word Recognition.”, in the International Conference on Information Technology, 2008.

[14] Supriya Deshmukh, Leena Ragha, “Analysis of Directional Features - Stroke and Contour for Handwritten Character Recognition", 2009.

[15] Bidyut B. Chaudhuri, Sumedha Bera, “Handwritten Text Line Identification In Indian Scripts”, in the 10th International Conference on Document Analysis and Recognition, 2009.