International Journal of Advanced Science and Technology
Vol. 10, September, 2009

Coarse Classification of Handwritten Hindi Characters

Pooja Agrawal, M. Hanmandlu, Brejesh Lall

Department of Electrical Engineering, I.I.I. Delhi, Hauz Khas,
New Delhi-110016, India
Pooja.iitd@gmail.com, mhmandlu@ee.iitd.ac.in, Brejesh@ee.iitd.ac.in

Abstract

This paper describes a system to classify the off-line handwritten Hindi characters into several groups based on some similarity measure. A novel method is proposed for finding the header line, based on end points and pixels positions in the top half part of the character image. The algorithm works in the presence of slant of the header line. After the identification and removal of header line, all the characters are coarse classified. A new algorithm is designed for the identification of presence and position of vertical bar in the handwritten Hindi characters. A coarse classification rate of 97.25% has been achieved in the simulation study.

Keywords: *Header line, Vertical bar, Handwritten Hindi characters, End points.*

1. Introduction

Off-line handwritten Hindi character recognition is one of the most difficult tasks of optical character recognition because of complex patterns, large number of classes involving basic characters and matras, different writing styles and sizes etc. It also requires a large amount of time as recognition module has three stages viz., pre-processing, feature extraction and matching with the database. We adopt the divide-and-conquer policy wherein a major category is divided into sub-categories thus making the classification process simpler. Accordingly we are inclined to develop a technique to divide a complete set of characters into some sub sets using a similarity measures.

It may be noted that the classification of handwritten Hindi characters into sub-groups has been a challenging problem since 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 Hindi characters is 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 in [3] 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 novel algorithm is developed for the identification and removal of header line from the handwritten Hindi characters. The algorithm takes care of slant in the header line. Finally, the coarse classification of the pre-processed characters is attended by devising two algorithms which identify the vertical bar (presence and position) and presence of close loops in the characters.

The rest of this paper is organized as follows: Section 2 describes the pre-processing of the characters. The algorithms for the coarse classification of characters are dealt with in Section 3. Section 4 presents the experimental results. Finally, conclusions are drawn in Section 5.

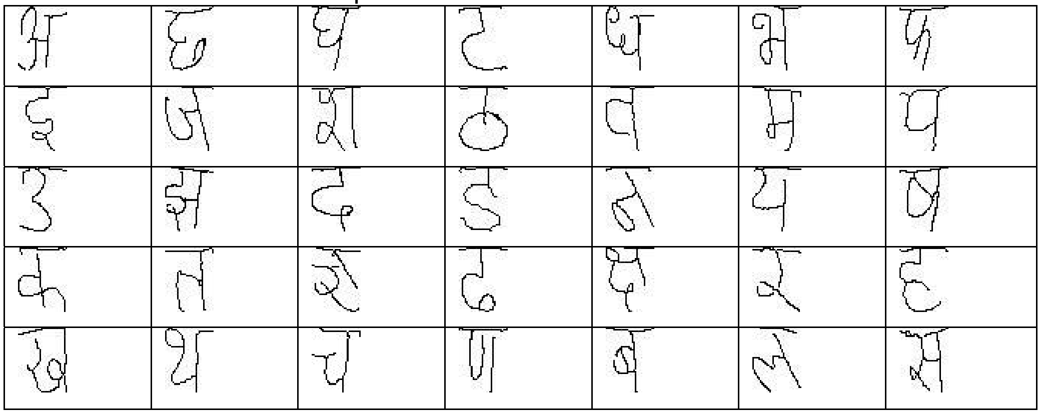
2. Preprocessing

The goals of preprocessing include the database collection, binarization and thinning of character images, and removal of header line from the characters.

2.1 Database Collection

Database creation is an important task for which we had used the services of undergraduate students over several lab sessions during their spare times. Each student is asked to 5 samples of a character set. Students are encouraged to write samples in varying styles. Table.1 shows the samples of handwritten Hindi characters collected from the students.

Table 1. Samples of handwritten Hindi characters



2.2 Identification and Removal of Header Line

The input to the Pre-processor which pertains to the preprocessing stage is the handwritten character image. First of all binarization [4] of the image is undertaken. This operation results in an image containing two gray values: one for the background and the other for the character.

To bring out the shape of the character image thinning is performed in [5]. Fig. 1 shows a binarized and thinned character images. Next the header line of the thinned characters is removed using the Algorithm 1.

Figure 1. (a) Original character image, (b) Binarized and thinned character image

Algorithm 1: *Identification and Removal of Header Line*

The algorithm consists of the following steps:

Step 1: Check whether the header line is connected to as follows.

a) Pass the character image to the Contour Trace algorithm [6] to find the independent contours.

b) If we get only one contour image as an outcome, it means that the header line is touching the character then exits from Step 1, else continue to Step 1c.

c) In the case of more than one contour, we need to check the position, curvature and the number of end points [7] of the observed contour images.

d) If the position of the contour lies in the top half of the image and the two end points indicate that the contour under consideration is a header line, then go to Step 2.

Step 2: Find the number of end points in the top half part of the character image.

Step 3: If there is only one end point then we scan the top half of the image vertically (top to bottom) for each column from the identified end point to the left side in order to capture the first black pixel (foreground pixel) positions (row and column) and go to Step 6.

Step 4: In the case of more than one end point encountered, the right most and the left most end point positions in the top half part of the image have to be found.

Step 5: Scan the top half part of the image vertically (top to bottom) for each column from the position of the right most end point to the left most end point in order to capture the first black pixel positions (row and column). The row positions of all the captured pixels are saved in an array.

Step 6: Find the pixels belonging to the header line as follows:

a) Take the pixels found in previous step and calculate the differences between the two adjacent pixels rows.

b) If the difference is more than a threshold, it means that the pixels at the left side of the left side pixel under consideration belong to the character.

c) The rest of the pixels (or the right side pixels of the right side pixel under consideration) belong to the header line if the condition in Step 6b is true.

d) The left side and right side pixels under consideration in Step 6a will be considered as a character and the header line pixels if the conditions in 6b and 6c are true.

e) The intensities of the header line pixels are converted into the background intensity as shown in Tables 2-5.

Step 7: In some cases, certain pixels are common to both the character and the header line as shown in Table 5. If we remove the header line these common pixels of the character will also be removed. In this case the following operations are needed to reconstruct the character:

a) Apply the Contour Trace algorithm on the resultant character image (after applying Steps 1 to 6).

b) Check the number of contours.

c) If the number of contour is the same as before the removal of header line then exit. Else continue to the next step.

d) Find the positions of the pixels (from the segmented header line) that are part of the character and convert them into black as in Table 5.

3. Coarse Classification

At this stage of classification, we are concerned with the separation of pattern classes into several groups based on the similarity of characters as shown in Fig. 2. These groups can be classified according to the space between the components of the same character, presence and position of the vertical bar, presence of one or more than one closed loops.

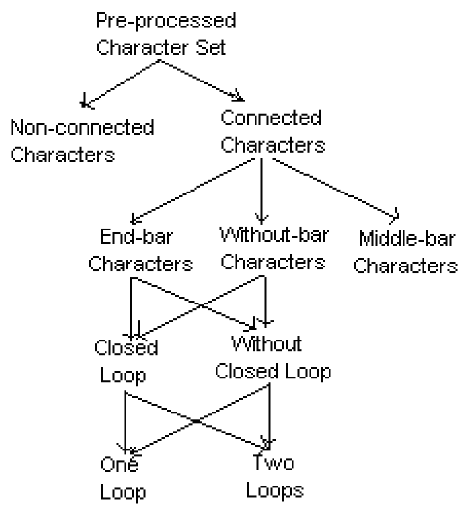


Figure 2. Coarse classification sequence of Hindi handwritten characters

3. 1 Identification of Non-Connected Characters

To detect whether the character is connected or not, the position of the first black pixel (from the top left corner) of the character is passed onto the Contour Trace algorithm. The algorithm returns the first contour of the character as shown in Fig. 3(c). Then the total number of black pixels of the contour image is compared with that of the character image without the header line (shown in Fig. 3(b)). If the total number of pixels of the character image is greater than the total number of pixels of the contour then the components of the character under test are non-connected; otherwise the character is connected.

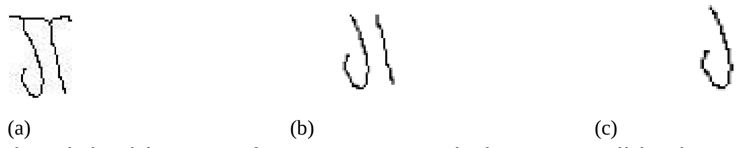


Figure 3. (a) Original image of non-connected character, (b) Character image after removing header line   (c) Contour image

3.2 Identification of End-bar, Middle-bar and Without-bar Characters

An algorithm is developed to identify the vertical bar from the handwritten characters. The algorithm does not require prior slant correction and normalization. It works effectively in the presence of slant (in the vertical line) in a 3x3 window [6]. The approach given in [6] is not able to identify the exact pixel positions of a vertical bar whereas the proposed algorithm identifies the exact pixel positions of a vertical bar in the end-bar and middle-bar characters.

Algorithm 2:

The steps of the algorithm are out lined here:

Step 1: Take a connected character image and divide it into two equal parts (Top half and Bottom half) from the middle row as shown in Fig. 4.

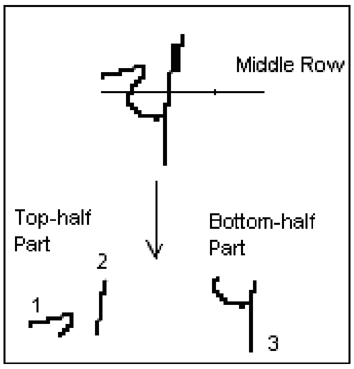


Figure 4. Separation of character image into top and bottom half parts

Step 2: Represent the end points of both parts by numbers 1, 2 and 3 as shown in Fig.4.

Step 3: If any part does not have end points, it indicates that character under test is without a bar then end else continue to Step 4.

Step 4: Find the end points at the right most position in both parts. Fig. 5 shows the right most end points for the top-half and bottom-half parts represented by 2 and 3.

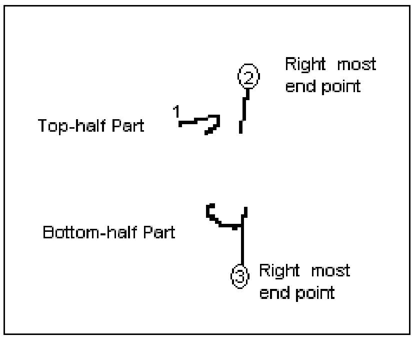


Figure 5. 2 and 3 points indicate the right most end points in the top and bottom half mage

Step 5: Identify whether the end point lies at the left side (as shown in Fig. 6) from the end points (2 and 3) found in Step 4.

I

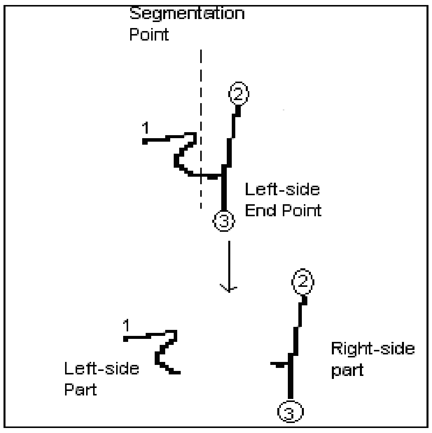


Figure 6. Partition of preprocessed character image (vertically) into two parts (Left-side and Right-side)

Step 6: Divide the character image vertically at a position a few columns before the left side end point (segmentation point) as shown in Fig. 6.

Step 7: Scan the right side part horizontally (right to left) row wise in between the end points (2 and 3 shown in Fig. 6) from top to bottom and capture the first black pixel. The column positions for all the captured pixels are saved in an array.

Step 8: Calculate the difference between the two adjacent pixel columns from the saved pixel columns from the previous step.

Step 9: If all the scanned rows have black pixels and calculated difference in Step 8 is not more than a threshold; it means that the character under consideration has a vertical bar at the end position. Then the character is identified as an end-bar character and end else continue to the next step.

Step 10: If all the scanned rows have black pixels and the calculated difference in Step 8 is more than a threshold; it means that the character under consideration has a vertical bar at the middle position. The character is identified as a middle-bar character and end else continue to the next step.

Step 11: If all the scanned rows do not have black pixels then it means that the character under consideration is without a bar.

3.3 Identification of Closed loop

Algorithm 3:

The algorithm consists of the following steps:

Step 1: Take the character with no header line and end bar (if present) and fill the closed area by black as shown in Fig. 7.

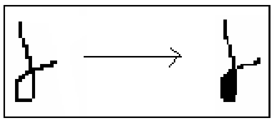


Figure 7. Character image after filling the closed area

Step 2: Apply the OR operator for the region between the filled image (shown in Fig. 8(b)) and the inverted preprocessed image (shown in Fig. 8(a)) to find the filled area as shown in Fig. 8(c).

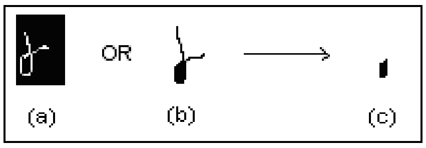


Figure 8. (a) Inverted character image, (b) Image after filling closed area and (c) Filtered image after applying OR operation between image (a) and (b)

Step 3: Apply the Contour Trace algorithm onto the filtered image (shown in Fig. 8 (b)) and check whether the filtered image has any contour or not.

Step 4: If there is no contour it means that the character under consideration has no closed loop and end; else it has a closed loop.

Step 5: Check for the close loops and look for the independent contours.

Step 6: If there is only one contour, it means that there is only one closed loop in the character; other wise there are two closed loops.

4. Experimental Results

Table 2. Header line not connected to the character

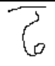
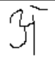
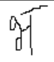
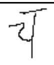
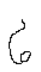
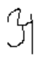
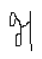
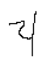
Character with Header Line				
Character after applying Algorithm 1				

Table 3. Header line not covering the whole character

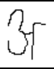
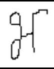
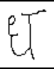
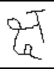
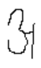
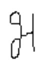
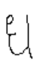
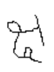
Character with Header Line				
Character after applying Algorithm 1				

Table 4. Header line covering the whole character

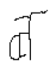
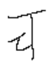
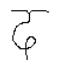
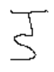
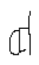
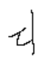
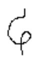
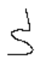
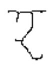
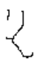
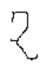
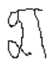
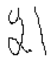
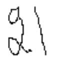
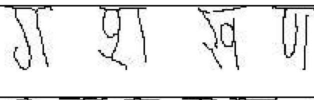
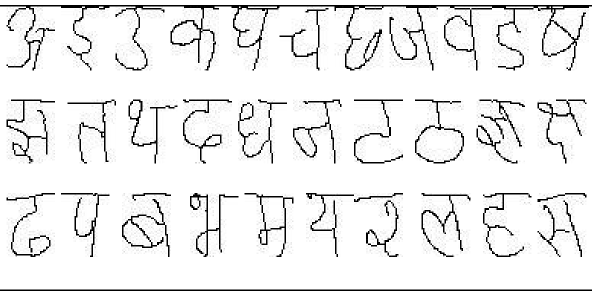
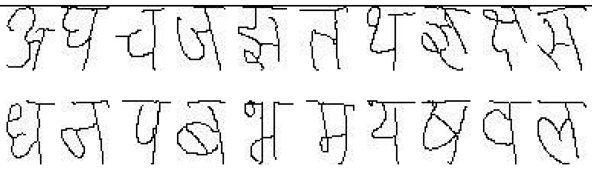
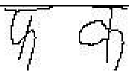
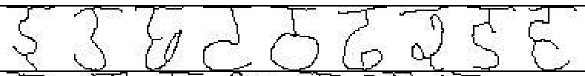
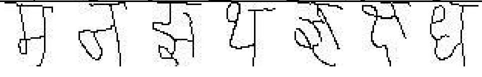
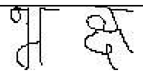
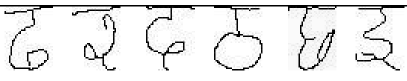
Character with Header Line				
Character after applying Algorithm 1				

Table 5. Header line part of the character

S. No.	Character with Header Line	Character after Applying Algorithm 1 up to step 6	Character after Applying Step 7 of Algorithm 1
1			
2			

The database consists of 100 samples of each character written in different styles for ascertaining the effectiveness of the proposed coarse classification approach. Different fonts are considered while classifying the handwritten characters. The proposed algorithm achieves 99% in the removal of header line. The coarse classification rate 98.5% is accomplished. The misclassification percentage for the set of handwritten Hindi characters is 2.5%. Samples of coarse classified characters are given in Table 6. Results of Algorithm 1 are in Table: 2-5.

Table 6. Samples of coarse classifies characters

Connected Characters	
Non-Connected Characters	
End-bar Characters	
Middle-bar Characters	
Without-bar Characters	
End-bar Characters with One Close Loop	
End-bar Characters with Two Close Loops	
Without-bar Characters with Loop	

Distortion cases:

A few cases of distortion are discussed now.

Algorithm 1:

(i) The case of a header line with no end points is shown in Fig. 9. Only one end point is present in the top half part of the character image in Fig. (a). The proposed algorithm for the identification of header line is not able to find the header line pixels as the algorithm finds detects the pixels that lie before the right most end point in the top half part.

Figure 9. (a) Binarized and thinned characters image, (b) Character image after applying header line algorithm

(ii) In this case all pixels of the header line cannot be found. This situation results from Step 6 of Algorithm 1 that does not remove pixels if the difference between the two adjacent pixels column is more than a threshold value.

Figure 10. (a) Binarized and thinned characters image, (b) Character image after applying header line algorithm

Algorithm 2:

(i) Some characters are identified as end-bar characters despite without a bar. According to Algorithm 2 a character will be identified as an end bar character if it satisfies the condition of Step 9. Character given in Fig. 11 satisfies the condition of Step 9, so it is detected as an end bar characters.

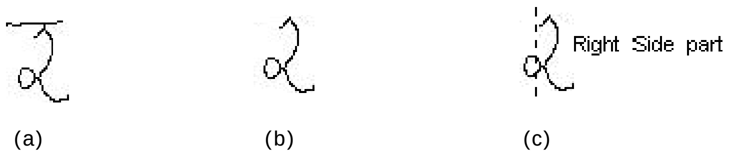


Figure 11. (a) Binarized and thinned characters image, (b) Character image after applying header line algorithm, (c) Character identified as an end-bar character

(ii) The character (shown in Fig. 12) is identified as without bar character despite belonging to the character that has end-bar. This occurs because the condition given in Step 11 of Algorithm 2 is satisfied.

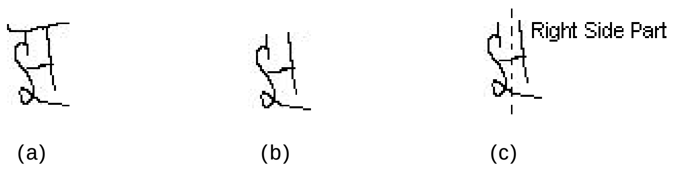


Figure 12. (a) Binarized and thinned characters image, (b) Character image after applying header line algorithm, (c) Character identified as a without bar character

5. Conclusions

This paper presents a system for the categorization of he complete handwritten Hindi character set into sub-groups based on some similarity measure. The coarse classification is based on the space present in between the components of the same character, position and presence of the vertical bar, presence of one or more closed loops. Header line is removed from the characters at the time of pre-processing in spite of having a slant(Top). A novel algorithm is developed for the identification and removal of header line from Hindi handwritten characters. An algorithm is developed for the identification of vertical bar. Experimental results demonstrate the effectiveness of the proposed algorithms.

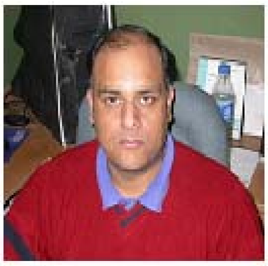
References

[1] Veena Bansal and R.M.K. Sinha, "A complete OCR for printed Hindi text in Devanagari script, Document Analysis and Recognition", Proceeding IEEE Sixth International Conference, Seattle WA, 2001, pp. 800-804.
[2] Huanfeng Ma and David Doermann, "Adaptive Hindi OCR Generalized Hausdoff Image Comparison", ACM Trans. On Asian Information Processing, 2(3), 2003, pp. 193-218.
[3] U. Garain, B.B. Chaudhuri, "Segmentation of touching characters in printed Devanagari and Bangla Scripts using fuzzy multifactorial analysis", Proc. 6th ICDAR, Seattle, WA, USA, 10-13 Sept. 2001, pp. 805-809.
[4] R. C. Gonzalez, R. E. Woods, "Digital Image Processing", Addison Wesley publishers, 1993.
[5] R. Stefanelli and A. Rosenfeld, Some Parallel thinning algorithms for digital pictures, Journal of the Association for Computing Machinery, 18(2), 1971, pp. 255-264.
[6] M. Hanmandlu, Pooja Agrawal, "A Structural Approach for Segmentation of Handwritten Hindi Text" Proceeding of the International Conference on Cognition and Recognition, Mandya, Karnataka, India, 22-23 December 2005, pp.589-597.
[7] Anjum Ali, Mohmood Ahmad, Nasir Rafiq, Javed Akber, Usman Ahmad and Shahwar Akmal, "Language Independent Optical Character Recognition for Hand Written Text", IEEE, INMIC 2004

Authors

POOJA AGRAWAL received her B.E. degree in Electronics and Communication from University of Rajasthan, India, in 2004, and M.S degree in Electrical Engineering from Indian Institute of Technology, Delhi, India, in 2008. Her research interests include Image Processing, Bioinformatics and Handwritten Text Segmentation and Recognition. She is reviewer of many national and international reputed conferences and journals including Pattern recognition and IEEE Transactions on SMC.

MADASU HANMANDLU received his B.E. degree in Electrical Engineering from Osmania University, Hyderabad, India, in 1973, M.Tech. degree in power systems from R.E.C. Warangal, Jawaharlal Nehru Technological University (JNTU), India, in 1976, and the Ph.D. degree in control systems from Indian Institute of Technology, Delhi, India, in 1981. From 1980 to 1982, he was a Senior Scientific Officer in Applied Systems Research Program (ASRP) of the Department of Electrical Engineering, IIT Delhi. He joined the EE department as a lecturer in 1982 and became Assistant Professor in 1990, an Associate Professor in 1995 and finally a Professor in 1997. He was with Machine Vision Group, City University, London, from April –November, 1988, and Robotics Research Group, Oxford University, Oxford from March-June, 1993, as part of the Indo-UK research collaboration. He was a Visiting Professor with the Faculty of Engineering (FOE), Multimedia University, Malaysia from March 2001 to March 2003. He worked in the areas of Power Systems, Control, Robotics and Computer Vision, before shifting to fuzzy theory. His current research interests mainly include Fuzzy Modeling for Dynamic Systems and applications of Fuzzy logic to Image Processing, Document Processing, Medical Imaging, Multimodal Biometrics, Surveillance and Intelligent Control. He has authored a book on Computer Graphics in 2005 under PBP publications and also has well over 190 publications in both conferences and journals to his credit. He has guided 15 Ph.Ds and 90 M.Tech students. He has handled several sponsored projects. He is presently an Associate Editor of both Pattern Recognition Journal and IEEE Transactions on Fuzzy Systems and a reviewer to other journals such as Pattern Recognition Letters, IEEE Transactions on Image Processing and Systems, Man and Cybernetics. He is a Senior member of IEEE and is listed in *Reference Asia; Asia's who's who of Men and Women of achievement; 5000 Personalities of the World (1998), American Biographical Institute.*



BREJESH LALL was born in Delhi, India, on September 3, 1970. He received the B.E. and M.E. degrees, both from Delhi College of Engineering, Delhi University, in 1991 and 1992 respectively. He received the PhD degree in the area of Signal Processing from Indian Institute of Technology Delhi in 1999. He worked in the Signal Processing group at Hughes Software Systems, Gurgaon, India from 1997 to 2005. He has been with Indian Institute of Technology Delhi since 2005 as an Assistant Professor in the Department of Electrical Engineering. His research interests include multirate signal processing, biometrics and image and video compression