An Approach for Character Segmentation of Handwritten Bangla and Devanagari Script

Anmol J Bhattad
Electronics & Communication Engineering
National Institute of Technology Karnataka, Surathkal
Mangalore, India
bhattad.anmol@gmail.com

Bidyut B. Chaudhuri
Computer Vision & Pattern Recognition Unit
Indian Statistical Institute
Kolkata, India
bbcisical@gmail.com

Abstract- **In this paper a scheme for segmentation of unconstrained handwritten Devanagari and Bangla words into characters and its sub-parts is proposed. Firstly, the region above headline is identified by counting the number of white to black transitions in each row, which followed by its separation. Then the characters are segmented using fuzzy logic. For each column, the inputs to the fuzzy system are the location of first white pixel, thickness of the first black stroke, count of white pixels, and the run length count of white pixels.**

Index Terms— **handwritten, character segmentation, fuzzy logic, Bangla handwriting, Devanagari handwriting**

I. INTRODUCTION

Segmentation of handwritten word into to character is an essential step in Optical Character Recognition (OCR) of textual documents. The components resulting from segmentation are taken for feature extraction. The characters are recognized based on these features [28]. The accuracy of character recognition is dependent upon character segmentation.

One of the early studies on word segmentation is based on detecting local minima along the contour and regions of low vertical profiles [1]. In [3], locating a minima in the lower or upper contours of a word image is used to detect ligatures. Using contour analysis, multiple segmentation points are determined between characters by [4]. The most plausible touching point is decided by word recognition and character recognition. The segmentation method used by [5] partly mimics the cognitive-behavioral process used by human subjects to recover motor-temporal information. Also, approaches based on over-segmentation by heuristics [6], rule-based segmentation [7], vertical contour-based segmentation [8] and modified direction feature-based segmentation [9] are proposed in recent past. Neural network is used to validate segmented characters in [6-9].

In [2] segmentation of handwritten digit by contour chains subdivided into valleys, mountains, holes, and open regions is proposed. The paper [10] discusses an approach which recognizes whether a pairs of characters are separable or not. The drop-fall algorithm that mimics the fall of raindrop is used for digit segmentation in [11].

In [12] the best segmentation boundaries for Chinese characters are found by applying dynamic programming and knowledge-based merging methods. In [13] Lengths of sub-strokes and the topological relations between these sub-strokes are used to find the connected point followed by thinning based segmentation path. In [14] block sequences are extracted from the images of mail address using connected components, projection profile and stroke cross number analysis. Later, a dynamic programming search is performed based on recognition confidence. In [15] a coarse segmentation is performed based on the height and width of the component. Following which a fine segmentation is performed by Fuzzy threshold and feedbacks. In [16] also coarse segmentation followed by fuzzy decision rules learned from examples are used to evaluate the segmentation paths. A subspace method for character recognition in segmentation process is used in [17]. In [18] potential ligatures between Chinese characters are located using split and merge strategy. The merging is performed on the over-split components. A recognizer aids the segmentation process. The thinning of background regions is used in [19] to segment connected two-digit strings. Feature points are matched to construct the possible segmentation paths, which are then ranked using fuzzy rules. They are further tested using a nearest neighbor classifier.

Among work in Indian scripts [20] divides the word into different zones and determines Matra, followed by recursive contour following in one of the zones across the height of the word to find the segmentation points. In [21] character segmentation is done using water reservoir principle. A fuzzy set based technique [22] was proposed for detecting the black pixels constituting the Matra and identifying pixel on Matra along which character segmentation is done. In characters are identified and segmented using a polynomial kernel based Support Vector Machine (SVM). Further heuristics are used to recheck the segments obtained. [23] uses a structural approach to segment simple Devanagari words. After headline detection, a vertical projection method is used in [25]. Also, [26] uses profiling method based on the vertical and horizontal density of black.

In this paper we propose a segmentation technique for Bangla and Devanagari characters. The alphabets in Bangla script consists of 11 vowel and 39 consonant characters.

Among the 49 basic characters of Devanagari script, there are 11 vowels and 38 consonants. These characters may be called basic characters. A modified character (allograph) is a vowel following a consonant [27]. In both scripts, depending on the vowel a modified shape is placed at the bottom, right, left (or both) of the consonant. In these scripts, the characters are connected together to form a word by a horizontal line at the upper part called headline. In Devanagari and Bangla languages, headline is called Shirorekha and Matra, respectively.

Most of the work done on character segmentation is based on determination of headline [20-22]. But the absence of distinct headline (Fig. 1) makes the task more difficult. Here, a method has been proposed here for character segmentation, without determination of headline.

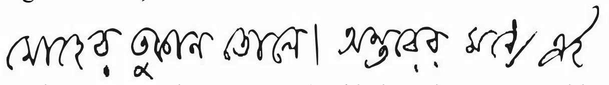


Fig. 1. A sample from one of Rabindranath Tagore`s writing.

II. METHODOLOGY

Bangla and Devanagari characters are connected in the upper most portion to form a word. This ligature is generally a part of headline. If the word has characters like ো, , etc the ligature is shifted vertically downwards (Fig. 2(a)). To shift the ligature back to the upper most portion of the word before segmentation, it is necessary to separate out the part of character below the headline. The headline might not be there in handwriting of many persons. The word in Fig. 2(a), doesn't have a headline, but the region inside the dashed box can be roughly approximated to be the headline. In such cases it is very difficult to separate the region above headline. Hence, the following algorithm is proposed.

A. *Separation of the region above headline*

This process is carried out in two steps. Step 1 identifies the words having region above headline. This causes few false identification of words which don't have a region above headline, which is validated in Step 2. The region above the headline is separated from main character string in Step 2.

Step 1: *Identification of words with region above headline*

The image of the word is traversed row-wise from left to right and the number of transitions from white pixel to black pixel in each row is calculated. Let T_i denote the number of transitions in the i^{th} row of the image, which is plotted in Fig. 2.(b) for the word shown in Fig. 2.(a). The total number of

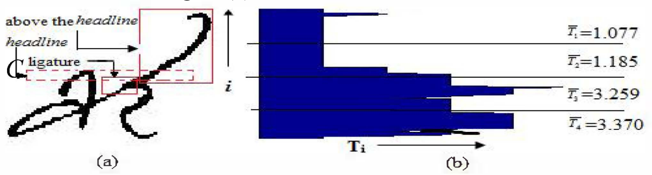


Fig. 2. (a) Word (b) No of transitions(T_i) is plotted against row no(i) for the word in (a).

rows H, is divided into 4 bands of almost equal length. Let the average value of number of transition(T_i) in each band be denoted by $\overline{T_l}$, where $l \in \{1,2,3,4\}$ The value of $\overline{T_l}$ for each band is denoted in Fig. 2(b).

On observing Bangla and Devanagari script we can see that, the region above headline, if exists, will occupy only the first band and the other bands are occupied by the main characters. Also, the part of word below headline will have more transitions compared to the region above headline and will depend on the number of characters in the word.

If $\overline{T_1} < k*(\max(\overline{T_2}, \overline{T_3}, \overline{T_4}))$ i.e, if the average transition in first band is lesser than k times the maximum average transition of the other three bands, then the word is identified to have a region above headline. The value of k is chosen to be 0.6 for our experiment, which is sufficiently to identify all the words having the region above the headline correctly. But keeping this threshold so high will also cause false identification of words which don't have region above headline, which will be validated in Step 2.

Step 2: *Segmentation and validation of the region above headline*

Firstly, we apply morphological thinning to these words. Then, beginning with the first row, the image is traversed from left to right until a black pixel is encountered. From this pixel we traverse in all possible directions of 3x3 neighbourhood having a black pixel (shown with red arrows in Fig. 3(a)), till we encounter a junction. The junction is the point where the part above headline region meets the main character and denotes a pixel which has more than three black pixels in its 3x3 neighbourhood. The word is then segmented at this point to separate the region above headline (Fig. 3(b)), provided that the height of the region is less than a threshold height and the junction is present in the upper half of the word. The threshold height is chosen to be half the total height of the word. This validates if the word is wrongly identified in Step 1.

Repeat Step 1 and 2 until the word is rejected in either of the steps i.e, there no part of the word which is above the headline still unsegmented from the main character string. We also store the point of segmentation along with the region above the headline which will join to its respective character, once the character segmentation is complete.

B. *Fuzzy Logic*

In contrast to crisp sets, where an element has either a full or zero membership i.e, it takes value from {0,1}, in fuzzy sets

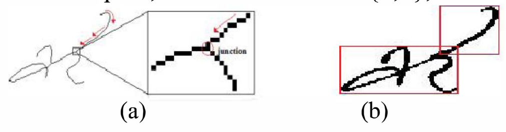


Fig. 3. (a) junction in a morphologically thinned image. (b)separation of the region above *headline*.

elements have a degree of membership i.e, it takes values from [-1,1]. The function that ties a real number from the set [-1,1] to each element x of the universe is called membership function μ(x). Based on the property of the element a one to one function such as triangular, trapezoidal, piecewise linear, Gaussian, or singleton are chosen to be membership function.

C. Segmenting the Character

Let the horizontal (rows) and vertical (column) lengths of the word to be *h* and *w* respectively. Each column is traversed downwards to find the row number (i) of first black pixel, say p_1 and the first white pixel following it, say p_2 , i.e, i values where the first stroke of each column starts and ends (Fig. 4).

The first feature to the fuzzy system is based on the fact that the ligatures are present in the upper portion of the word. The location of the first black pixel in a column, is the first input **X1** (Fig. 4) for our fuzzy system i.e, **X1**= p_1 . Lesser the value of **X1**, more is the belongingness of that column to a valid point of segmentation. Thus, the membership function μ_1 in Fig.5(a), is designed for **X1**.

The thickness of the first stroke of each column is the second feature **X2** (Fig. 4) to the fuzzy system, where **X2** $= p_2 - p_1$. More the thickness of the stroke, less is the belongingness of that stroke to a valid point of segmentation, because the column which is a part of ligature has comparatively lesser thickness than the rest of the word. Thus, the membership function μ_2 shown in Fig. 5(b), is designed for **X2**. Based on the average thickness of the stroke, we can decide the threshold (Fig. 5(b)) d_1 and d_2 . For our database we take d_1 =5 and d_2 =15.

It is observed that when two characters are connected to form a word, the space below the point of joining is either empty or have very less black pixel. We use this feature of Bangla and Devanagari script to design next two inputs. For

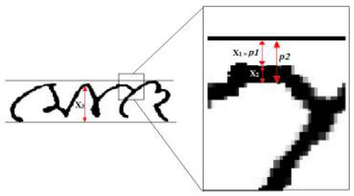
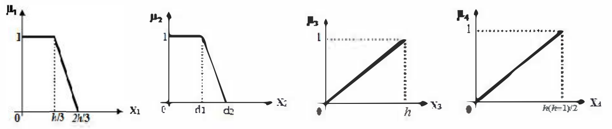


Fig. 4. An illustration of feature X_1, X_2, X_3 and X_4



(a) (b) (c) (d)

Fig. 5. Membership function μ_1 , μ_2 , μ_3 and μ_4 .

each column, starting from p_2 upto the last pixel, we count the number of white pixels. We call this as white pixel count **X3** (Fig. 4). This is the third feature to the fuzzy system. More the value **X3**, more is the belongingness of that column to a valid point of segmentation. Thus the membership function μ_3 shown in Fig. 5(c), is designed for **X3**. The max value of **X3** can be the height of the word *h*. The run length value of a white pixel is its distance from the previous black pixel vertically above it (in the same column). For a white pixel (i , j), the run length value is k , iff there exist white pixels ((i-k+1, j), (i-k+2, j),...,(i-1,j)). So, the run length count of continuous run of n pixels is the sum of the run length value of those n pixels, i.e., Σ k= n*(n+1) / 2. For each column, starting from p_2 , we find the sum of run length value of the white pixels in that column. We call this as vertical run length count (**X4**) of white pixels from p_2 . This is the fourth feature to our fuzzy system. More the value of vertical run length count, more is the belongingness of that column, to a valid point of segmentation. Thus the membership function μ_4 in Fig. 5(d), is designed for **X4**. Max value of **X4** can be *h**(*h*+1)/2.

To determine the column having the ligature, we find the value of features, **X1**, **X2**, **X3** and **X4** for each column. The possibility of ligature in the j^{th} column is given by, $s_j = \mu_1 (X_4) * \mu_2 (X_2) * \mu_3 (X_3) * \mu_4 (X_4)$.

We pass s_j through Savitzky-Golay filter which is a generalized moving average with filter coefficients determined by an unweighted linear least-squares regression and a polynomial model of specified degree (we used degree 1). Let the output of the filter be o_j (Fig. 6(a)). The possibility of occurrence of ligature in each column as given by the fuzzy system is represented in greyscale level in Fig. 6 (b), (c), (d). Traverse the columns from left to right and note the points where the o_j is forming a maxima / peak (Fig. 7(b)). A peak is the point where o_j of the column is greater than the adjacent columns. Segment the word vertically along such columns (Fig. 7(a)).

Finally, we verify that the segmenting along the peaks actually forms a character or not. Consider the peaks in consecutive pairs. Split the word from p_1 to p_2 in the columns were these two peaks occur. If it forms a component of height < *h*/2, then discard that peak which has lesser possibility (o_j) among them. This is based on the observation that each character has height > *h*/2. For example, , consider peak 1, 2 and 3 of Fig. 7(b). If we segment the word along all the peaks, we can clearly see that the second component is wrong (Fig. 7(c)) and thus ignore

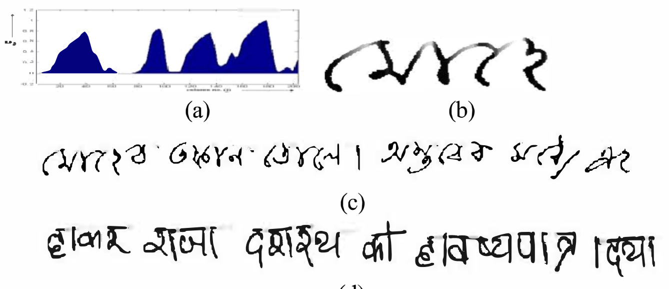


Fig. 6. (a) o_j plotted against the column no. for the word shown in Fig. 4. (b),(c),(d) The possibility of occurrence of ligature in each column is represented in greyscale level.

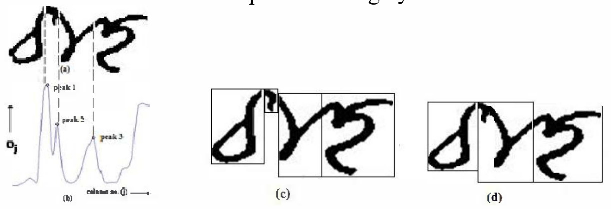


Fig. 7. (a)The possible segmentation points based on the peaks. (b) O_j plotted against column no. for the word in (a). Segmentation of the word (c) before verification. (d) after verification.

peak2. Therefore, only peak1 and peak3 are valid and used for segmentation (Fig. 7(d)). Now that the word is segmented into characters, the region above headline, which was separated earlier, is now joined to the word. Thus, we have separated each word into its constituent characters.

III. RESULTS AND DISCUSSION

The database used for the evaluation of the proposed algorithm is categorized into 3 types.

1. Bangla words from manuscripts of Rabindranath Tagore. This database doesn't have distinct headline(size-503words).
2. Bangla word images written by various people with a headline (size - 516 words).
3. Devanagari word images written by various people with a headline (size - 1004 words).

The segmented characters are divided into 4 types, i.e, correctly segmented (cc), over segmented (co), under segmented (cu) and unsegmented touching characters (ct).

Accuracy= cc/(cc + co + cu + ct)*100%
The results are tabulated in Table I.

Our Algorithm gives fairly good results on Category 1 handwriting. There is no past work done on character segmentation without detecting headline, to compare our

TABLE I. Result of character segmentation

Database	No of cc	No of co	No of cu	No of ct	Accuracy
Category 1	1900	102	109	124	85.01%
Category 2	1874	68	34	0	94.84%
Category 3	3558	114	116	22	93.39%

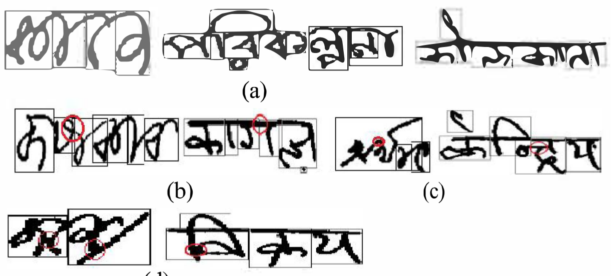


Fig. 8. (a) Correctly segmented (b) Over Segmented (c) Under Segmented (d) Touching characters (shown in red).

results. Also, the results obtained for category 2 and 3 are quite satisfactory. Fig. 8(a) shows correctly segmented words from each of the database.

The majority of the over segments (Fig. 8(b)) were observed in the Bangla characters are গ , প ঋ , শ , ল and Devanagari characters are आ , ऑ, औ, ग, श, ल, भ. In handwritten documents these characters can have a potential peak, which cause the over segmentation. These errors may be overcome by considering them in recognition stage.

The constituent of under segmentation (Fig. 8(c)) were the words which failed in the stage where the region above the headline is separated. In category 1 database it occurred in cases where the point of segmentation was in the lower half of the word (due to large variation in font size in a single word) or when the characters were too close and skewed.

In category 1 and 3 database few errors were due to the neighboring characters touching at multiple places. Such characters are called as touching characters (Fig. 8(d)). Category 1 database has very high percentage of touching characters (5.55%), because of the complexity of Tagore's writing. These characters have a larger width when compared to the individual character and can be filtered out and their segmentation can be considered in the future.

IV. CONCLUSION

The proposed algorithm gives good results on both Bangla and Devanagari scripts. It doesn't require a distinct headline for character segmentation. The use of Fuzzy technique for character segmentation helps us utilize multiple features of these scripts, making the algorithm robust. In future, more features can be added to fuzzy system to make the segmentation process better. If the present segmentation technique is backed by a recognition system, then better results can be achieved.

V. REFERENCES

[1] R.M. Bozinovic, and S.N. Srihari, "Off-line Cursive Script Word Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, 11(1), pp. 68-83, 1989.

[2] N.W. Strathy, C.Y. Suen, and A. Krzyyzak, "Segmentation of Handwritten Digits using Contour Features," Proc. ICDAR '93, pp. 577-580, 1993.

[3] M. Holt, M. Beglou, and S. Datta, “Slant-Independent Letter Segmentation for Off-line Cursive Script Recognition,” from Pixels to Features III, S. Impedovo and J.C. Simon (eds.), Elsevier, pp. 41, 1992.

[4] H. Yamada, and Y. Nakano, “Cursive Handwritten Word Recognition Using Multiple Segmentation Determined by Contour Analysis,” Proc. IEICE Transactions on Information and Systems, E79-D, pp. 464-470, 1996.

[5] P. Plamondon, and C. M. Privitera, "The segmentation of cursive handwritten: an approach based on off-line Recovery of the motor-temporal information," Proc. IEEE Transactions on Image Processing, 80 -91, 1999.

[6] M. Blumenstein, and B. Verma, “A New Segmentation Algorithm for Handwritten Word Recognition,” Proc. of the International Joint Conference on Neural Networks (IJCNN ’99), Washington D.C., pp. 878-882, 1999.

[7] B. Verma, “A Contour Code Feature Based Segmentation for Handwriting Recognition,” Proc. 7th ICDAR, pp. 1203 - 1207, 2003.

[8] F. Kurniawan, A. Rehman, and D. Mohamad, " From Contours to Characters Segmentation of Cursive Handwritten Words with Neural Assistance," Proc. ICICI-BME, pp. 1-4, 1999.

[9] C.K. Cheng, and M. Blumenstein, “The Neural Based Segmentation of Cursive Words Using Enhanced Heuristics,” Proc. 8th International conference on Document analysis and recognition, vol. 2, pp. 650-654, 2005.

[10] X. Wang, V. Govindaraju, and S. N. Srihari, “Holistic Recognition of Handwritten Character Pairs,” Pattern Recognition, vol. 33, pp. 1967-1973, 2000.

[11] X. Wang, K. Zheng, and J. Guo, “Inertial and Big Drop Fall Algorithm,” International Journal of Information Technology, vol. 12, no.4, pp. 39-48, 2006.

[12] L.Y. Tseng, and R.C. Chen, "A new method for segmenting handwritten Chinese characters," Proc. ICDAR, vol.2, pp. 568 - 571, 1997.

[13] S. Zhao, and P. Shi, "Segmentation of connected handwritten Chinese characters based on stroke analysis and background thinning," PRICAI'00, pp. 608-616, 2000.

[14] Z. Han, C.P. Liu, and X.C. Yin, "A two-stage handwritten character segmentation approach in mail address recognition," ICDAR'05, Vol. 1, pp. 111-115, 2005.

[15] F. Yang, J. Hui, Y.H. Liu, and X.D. Tian "Fuzzy Threshold Method For Off-Line Chinese Handwritten Character Segmentation," Proc. of the 8th International Conference on Machine Learning and Cybernetics, Vol. 2, pp. 679-682, 12-15 July 2009.

[16] S. Zhao, Z. Chi, P. Shi and Q. Wan, "Handwritten Chinese Character Segmentation Using a Two-Stage Approach," ICDAR'11, pp.179-183, 2011.

[17] Y. Ariki, and Y. Mot, “Segmentation and Recognition of Handwritten Characters using Subspace Method,” Proc. 3rd ICDAR, Vol. 1, pp. 120-123, 1995.

[18] J. Gao, X. Ding and Y. Wu, "A Segmentation Algorithm for Handwritten Chinese Character Strings," ICDAR '99, pp. 633-636, 1999.

[19] Z. Lu, Z. Chi, W. C. Siu and P. Shi, "A background thinning-based approach for separating and recognizing connected handwritten digit strings," Pattern Recognition, vol 32, Issue 6, pp. 921–933, June 1999.

[20] A. Bishnu and B.B. Chaudhuri, "Segmentation of Bangla handwritten text into characters by recursive contour following," Proc. ICDAR '99, pp. 402-405,1999.

[21] U. Pal and S. Dutt, "Segmentation of Bangla unconstrained handwritten text," Proc. ICDAR'03, Vol. 3, pp. 1128-1132, 2003.

[22] S. Basu, R. Sarkar, N. Das, M. Kundu, M. Nasipuri and D.K. Basu, "A Fuzzy Technique for Segmentation of Handwritten Bangla Word Images," Proc. ICCTA '07, pp. 427-433, 2007.

[23] G. Agrawal, Kshitij, A. Mukharjee, and N. Kumar, "Handwritten Devanagari Script Segmentation using Support Vector Machines," International conference on Neural Network (IJCNN), 2004.

[24] M. Hanmandlu, P. Agrawal, and B. Lall, " Segmentation of Handwritten Hindi Text: A Structural Approach," International Journal of Computer Processing of Languages, Vol 22, Issue 01, March 2009.

[25] N.K. Garg, L. Kaur, and M.K. Jindal, "A New Method for Line Segmentation of Handwritten Hindi Text,” Seventh International Conference on Information Technology, pp. 392-397, 2010.

[26] B. Singh, N. Gupta, R. Tyagi, A. Mittal, and D. Ghosh, "Parallel Implementation of Devanagari Text Line and Word Segmentation Approach on GPU," International Journal of Computer Applications, Volume 24– No.9, pp. 7-14, June 2011.

[27] B. B. Chaudhuri, and U. Pal, "Skew Angle Detection of Digitized Indian Script Documents," IEEE Transactions On Pattern Analysis And Machine Intelligence, Vol. 19, No. 2, February 1997.

[28] B. B. Chaudhuri, and U. Pal., "An OCR system to read two Indian language scripts: Bangla and Devnagari (Hindi)," Proc. ICDAR'97, vol. 2, pp. 1011-1015, Aug 1997.