Sādhanā Vol. 34, Part 5, October 2009, pp. 755–765. © Indian Academy of Sciences

A contour distance-based approach for multi-oriented and multi-sized character recognition

U PAL[1] and N TRIPATHY[2]

[1]Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, Kolkata 700 108
[2]IBM India Private Ltd., Saltlake, Kolkata 700 091
*e-mail: umapada@isical.ac.in; niltripa@in.ibm.com

MS received 14 September 2008; revised 7 July 2009

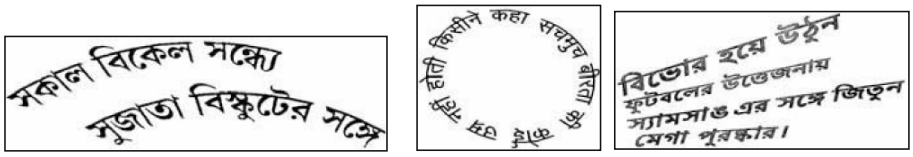
Abstract. In this paper, we propose a novel scheme towards the recognition of multi-oriented and multi-sized isolated characters of printed script. For recognition, at first, distances of the outer contour points from the centroid of the individual characters are calculated and these contour distances are then arranged in a particular order to get size and rotation invariant feature. Next, based on the arranged contour distances, the features are derived from different class of characters. Finally, we use these derived features of the characters to statistically compare the features of the input character for recognition. We have tested our scheme on printed Bangla and Devnagari multi-oriented characters and we obtained encouraging results.

Keywords. OCR; document analysis; multi-oriented character recognition; Indian script; Bangla and Devnagari scripts.

1. Introduction

There are many printed documents written in stylistic (artistic) way. Text lines of a stylistic document may not be parallel to each other. These text lines may have different orientations and/or the characters in these documents may be written in curved, rotated, or in other stylistic ways. Some examples of stylistic documents are shown in figure 1. Because of multi-oriented or curved shape of characters in an artistic document, it is very difficult to recognize such arbitrarily oriented characters. Many pieces of work are available on normal printed character recognition of Indian scripts (Chaudhuri & Pal 1988, Dhandra *et al* 2006, Lehal & Singh 2000, Negi *et al* 2001, Pal & Chaudhuri 2004) but there is no work towards recognition of Indian stylistic documents. In this paper, we propose a scheme towards the recognition of printed multi-oriented and multi-sized Bangla and Devnagari characters and the recognition of multi-oriented and multi-sized characters is done using rotation invariant features.

*For correspondence



(a) **(b)** **(c)**

Figure 1. Examples of stylistic document images. **(a)** Bangla Magazine image, **(b)** Devnagari synthetic image, **(c)** Bangla newspaper image.

Some pieces of work on English stylistic text recognition are available in the literature (Hase *et al* 2003, Hase *et al* 2001, Loo & Tan 2002, Lu *et al* 2004, Sato *et al* 2000, Tang *et al* 1991, Uchida *et al* 2007, Uchida *et al* 2006, Xie & Kobayashi 1991). Xie & Kobayashi (1991) proposed a rotation invariant recognition system using the patterns of different angular variation of the component and 97% recognition accuracy is obtained from the 10 digits of English alphabet. Some of the multi-oriented character handling approaches consider character re-alignment (Hase *et al* 2001) for recognition. Based on the types of the text (horizontal, vertical, curved, inclined, etc.), the characters in a text line are re-aligned horizontally and then OCR techniques are used. The main drawback of these methods is the distortion due to realignment of curved text. Adam *et al* (2000) used Fourier Merllin Transform for multi-oriented symbol and character recognition in Engineering drawings. This method is time consuming which is the main drawback of this technique. Parametric eigen-space based method is used by Hase *et al* (2003) for rotated and/or inclined English character recognition. Monwar *et al* (2007) proposed a rotation invariant approach where each character is described by a small set of 2D characteristic views of different angles (0 to 360 degrees) for feature extraction. Pal *et al* (2006) proposed a MQDF-based method for multi-oriented English character recognition.

Although there are a few pieces of published work on English stylistic text recognition (Hase *et al* 2003, Hase *et al* 2001, Loo & Tan 2002, Lu *et al* 2004, Sato *et al* 2000, Tang *et al* 1991, Uchida *et al* 2007, Uchida *et al* 2006, Xie & Kobayashi 1991) but no work is done on the recognition of Indian stylistic text documents. There exists a work (proposed by Pal and Roy (2004)) on Indian stylistic documents but it deals with extraction of individual text lines from Indian stylistic documents. In this paper, we propose a scheme for the recognition of Indian multi-oriented and multi-sized characters. Here, we consider isolated characters of Bangla and Devnagari scripts, the two most popular Indian scripts. The recognition technique is rotation invariant and it is based on the features obtained from the contour distances calculated from the center of gravity (CG) of a character. To get contour distance based features, at first, CG of a component is calculated and the distances of all outer contour points of the component from its CG are computed. These contour distances are then arranged in an order to make the contour distance based feature size and rotation invariant. We create a template database of contour distance based features of all character class and computing statistical measure between the features derived from arranged contour distance of the input character and that of the individual characters of template database, the input character is recognized.

Overall organization of the rest of the paper is as follows. Section 2 describes some important properties of Bangla and Devnagari scripts. Recognition technique of multi-oriented and multi-sized characters is detailed in section 3. Results and discussions are provided in section 4. Finally, conclusion is given in section 5.

Figure 2. Basic characters of **(a)** Bangla and **(b)** Devnagari alphabet are shown. First eleven characters are vowels and rest is consonants in both the alphabet sets.

2. Properties of Bangla and Devnagari scripts

The alphabet of the modern Bangla script consists of 11 vowels and 39 consonants. These characters are called as *basic characters*. Out of the 49 basic characters of Devnagari script, 11 are vowels and 38 are consonants. The basic characters of Bangla and Devnagari scripts are shown in figure 2. Writing style in both the scripts is from left to right. The concept of upper/lower case is absent in these scripts. From figure 2 it can be seen that most of the characters in Bangla and Devnagari scripts have a horizontal line (*Shirorekha*) at the upper part. From a statistical analysis we notice that the probability that a Bangla word will have horizontal line is 0·994 and the corresponding probability for Devnagari is 0·997 (Chaudhuri & Pal 1998).

In both Bangla and Devnagari scripts a vowel following a consonant takes a modified shape and placed at the left, right, both left and right, or bottom of the consonant. These modified shapes are called *modified characters*. Examples of modified characters are shown in figure 3. A consonant or a vowel following a consonant sometimes takes a compound orthographic shape, which we call as *compound character*. Compound characters can be combinations of two consonants as well as a consonant and a vowel. Compounding of three or four characters also exists in these two scripts. In both the script forms, there are about 280 compound characters (Chaudhuri & Pal 1998).

A Bangla or Devnagari text line can be partitioned into three zones. The *upper-zone* denotes the portion above the head-line, the *middle zone* covers the portion between head-line and

Bangla vowels	আ	ই	ঈ	উ	ঊ	ঋ	এ	ঐ	ও	ঔ
Modified shape	া	ি	ী	ু	ূ	ৃ	ে	ৈ	ো	ৌ
When attached to consonant ক	কা	কি	কী	কু	কূ	কৃ	কে	কৈ	কো	কৌ
Devnagari vowels	आ	इ	ई	उ	ऊ	ऋ	ए	ऐ	ओ	औ
Modified shape	ा	ि	ी	ु	ू	ृ	े	ै	ो	ौ
When attached to consonant क	का	कि	की	कु	कू	कृ	के	कै	को	कौ

Figure 3. Examples of Bangla and Devnagari modified characters.

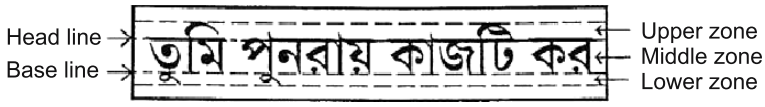


Figure 4. Different zones of a Bangla text line.

base-line, the *lower-zone* is the portion below base-line. Different zones in a Bangla text line are shown in figure 4. Languages like Hindi, Nepali, Sanskrit and Marathi are written in Devnagari while Bangla, Assamese and Manipuri languages are written in Bangla script. Moreover, Hindi and Bangla are the national languages of India and Bangladesh, respectively. Also, Hindi is the third most and Bangla is the fifth most popular language in the world (Pal & Roy 2004).

3. Character recognition

Size and rotation invariant contour distance features are considered in this proposed scheme for the recognition of multi-sized/multi-oriented characters. If a character is rotated in different angles then the number of contour points and its CG (Center of Gravity) do not change with the rotation of the character and hence the distance between contour point and CG will not change. Our plan is to compute the features based on these invariant characteristics of the characters. Here, considering outer contour points of the characters the contour distance features are calculated. By contour distance of an outer contour point of a character we mean Euclidean distance of the contour point from a reference point of the character. A point invariant to the rotation of the character is chosen as reference point. Since CG is rotation invariant, we consider CG as the reference point. The CG (x_c, y_c) of a character is calculated as follows:

$$x_c = \frac{1}{N}\sum_{i=1}^{P} x_i \quad \text{and} \quad y_c = \frac{1}{N}\sum_{i=1}^{P} y_i,$$

where $(x_i, y_i), i = 1, 2, \ldots, P$, are the P object points of the character. Contour distance formula for a contour point (X, Y) is $\sqrt{(X - x_c)^2 + (Y - y_c)^2}$.

Contour distances of the outer contour points of a character are computed as follows. Starting from the topmost left contour point of the character we sequentially compute the contour distances of all the outer contour points of the character in clockwise direction. For a component with B outer contour points we get B distances and based on these contour distances we define contour distance function $C(i)$ as follows:

$C(i) = \sqrt{(x'_i - x_c)^2 + (y'_i - y_c)^2}$, where $(x'_i, y'_i), i = 1, 2, \ldots, B$ are the consecutive points obtained by clock-wise traversing the outer contour of a component starting from topmost left contour point of the component, and (x_c, y_c) is the CG of the component. Contour distance function of a Bangla and a Devnagari character are shown with their three different size and orientations in figures 5a and b, respectively. In this figure X-axis represents contour points and Y-axis represents contour distances. (Here value of X is equal to 1 we mean the topmost left contour point of a component, and $Y(1)$ represents the contour distance of this topmost left contour point from the CG of the component. By value of X is equal to 2, we mean the second point obtained by clockwise tracing of the contour starting from the topmost left contour point, and $Y(2)$ represents the distance of this second contour point from the

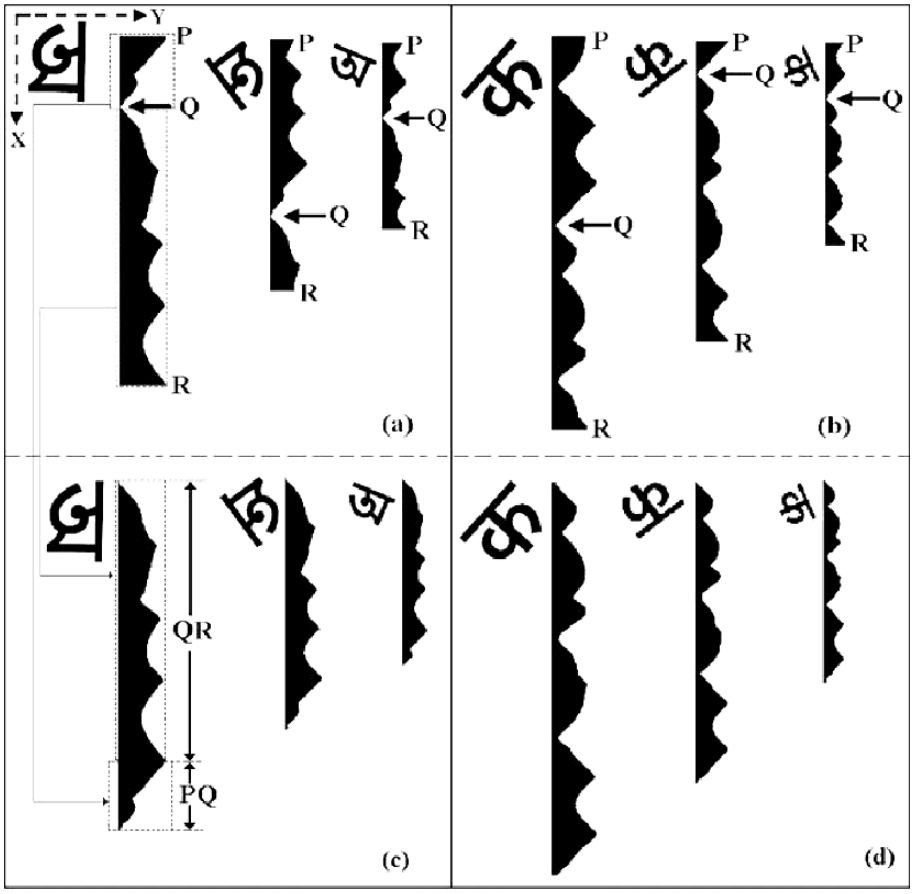


Figure 5. Contour distances of Bangla character and a Devnagari character are shown with their three different size/orientations in **(a)** and **(b)** respectively. Rearranged contour distance functions are shown in **(c)** and **(d)**, respectively.

CG. Similarly, $X = k$ we mean the k^{th} point obtained by clockwise tracing of the contour starting from the topmost left contour point, and $Y(k)$ represents the distance of this contour point from the CG). For efficient computation, we consider the contour distance function $C(i)$ as one dimensional array of size B if there are B contour points (here the index i represents the i^{th} points obtained by clockwise tracing of the contour starting from the topmost left contour point and $C(i)$ represent the contour distance of the i^{th} contour point from the CG. From figures 5a and b it can be noted that contour distance function shows different shape behaviours for different orientations of a character. Since we always start from the topmost left corner point for contour distance computation, we get different shape behaviour of contour distance function for different orientations of a character. To get similar shape of the contour distance function for different orientations of a character we have rearranged the contour distance values in the following way. For an arbitrarily oriented input character, analyzing its contour distance array $C(i)$, we note the contour point that has the smallest value. See figures 5a and b, where 'Q' denotes the point of the smallest contour distance. The contour distance function is then divided into two parts at the point Q. Let these two

parts be PQ and QR (see figures 5a and b). Now, we rearrange the contour distance function so that smallest contour distance point Q comes at the beginning. This is done by placing PQ portion after QR portion of the contour distance array. For example see the first figure of figure 5c where QR and PQ portions are shown after rearrangement. Rearrangement of such contour distance array is fast and it has complexity of $O(n)$, where n is the number of contour points of the character. Rearranged contour distances of the characters of figures 5a and b are shown in figures 5c and d, respectively. Because of this rearrangement it can be noted from figures 5c and d that a character always shows similar shape behaviour of contour distance even if the character is multi-sized and/or multi-oriented. We used the rearranged contour distance features for recognition purpose.

In some characters we may get two or more contour points having contour distance nearer to the smallest contour distance. Contour points having contour distance less than or equal to $\delta + R_L$ (R_L is stroke width of the character and δ is the smallest contour distance) are noted for a character and considered them as contour points nearer to the smallest contour distance of the character. If for a character we obtain two or more contour points having contour distance nearer to the smallest contour distance then we rearrange the contour distance considering each of such points. So for a character, if we have N such points then we will get N rearranged versions of contour distance function from that character. Note that the value of N does not depend on the size of a character and from our experiment we noted that average number of N in a Bangla (Devnagari) character is 4·51 (5·47).

Stroke width (R_L) is the statistical mode of the black run lengths of a character. For a character, R_L is calculated as follows. The character is, at first, scanned row-wise (horizontally) and then column-wise (vertically). If n different runs of lengths $r_1, r_2, \ldots, r_n$ with frequencies $f_1, f_2, \ldots, f_n$, respectively are obtained after these two scanning from the character, then value of R_L for that character will be r_i if $f_i = \max(f_j)$, $j = 1, 2, \ldots, n$.

For the use of contour distance feature in recognition, we segment the rearranged contour distance function into 16 parts of equal length using bisection concept. For segmentation, we first divide contour distance function into two equal parts segmenting at the middle of the contour points. See figure 6, here the middle point is marked by 'F'. Next, each of two segmented parts is further divided into two parts at their middle. Every time we note the middle point where segmentation is done. If we continue this procedure four times we will get 16 segments and 15 middle points where segmentations are done. Such 16 segments are shown in figure 6 for three different size/oriented images of a Bangla character. From the figure it can be seen that respective segments of the images show similar shape behaviour because of the rearrangement of the contour distances, and we statistically measure this shape behaviour for character recognition. Contour distances of the 15 middle points are noted and we use these 15 contour distances for the recognition purpose. See figure 6 where contour distances at these 15 points are shown by white horizontal lines. For a character, these 15 contour distances are used as the feature. In this figure FF^1 is the contour distance of the middle-most segmented point. We compute such 15 contour distances from samples of different class of characters and form a template set (D', say). For faster processing, D' is divided into a few subsets based on the number of valleys of the rearranged contour distance function of the characters. These subsets are generated in such a way that difference of number of valleys between any two characters of a subset will be less or equal to one. To find the number of valleys of the contour distance function we use water reservoir concept. The water reservoir principle is as follows: If water is poured from a side of a component, the cavity regions of the component where water will be stored are considered as reservoirs of the component. For details about water reservoir see Pal & Roy (2004).

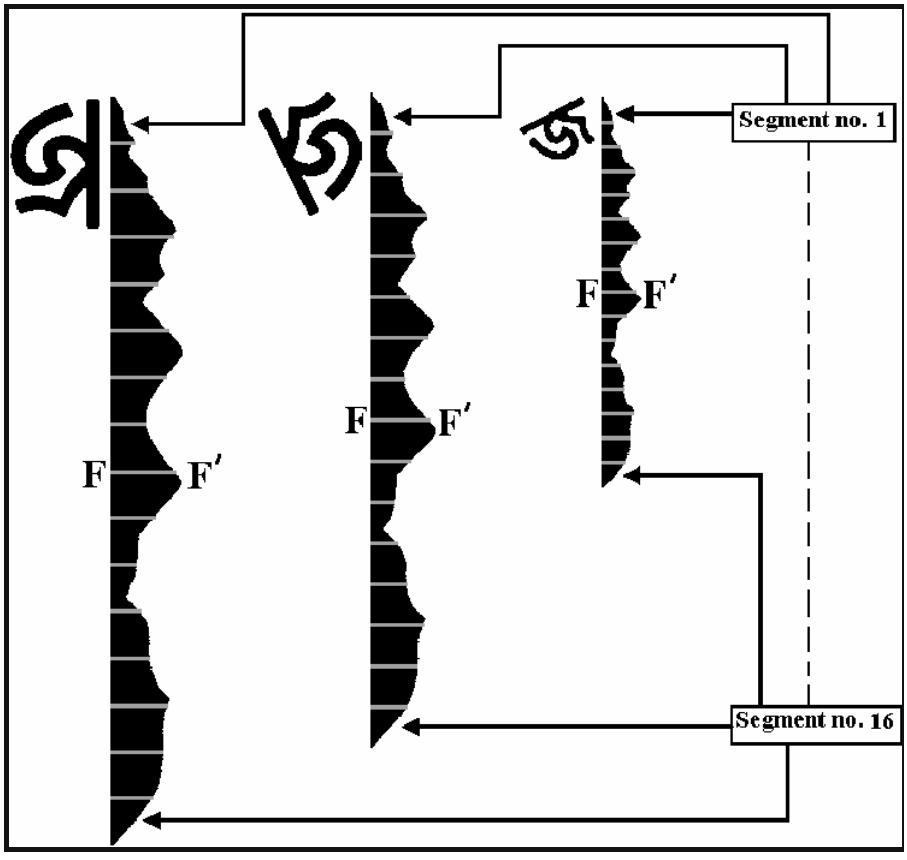


Figure 6. Sixteen segments of the contour distances for three different size/oriented images of a Bangla character are shown here. Fifteen contour distance values obtained from the Bangla characters are marked by white lines in the figures. Lengths of white horizontal lines between two consecutive segments represent contour distance values.

The portions of the contour distance function where water will be stored are considered as valley of the contour function. We compute number of reservoirs having height greater than stroke width (R_L) of the component. If V such reservoirs are obtained in the contour distance function of a character then the number of valleys of the character is V. Water reservoir based valley detection is shown in figure 7. In this figure there are five reservoirs and the heights of all the reservoirs are greater than stroke width hence there are five valleys in this figure.

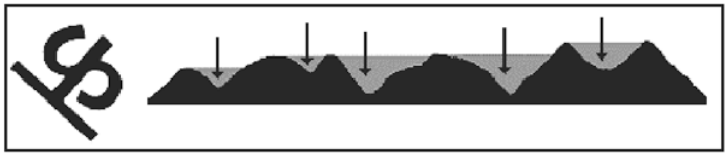


Figure 7. Valley detection by water reservoir principle. There are five top reservoirs (shown by arrow) and hence there are five valleys.

For faster computation, our recognition scheme is divided into two parts: (a) Initially depending on the number of valleys of the rearranged contour distance function of an input character, a subset (say S) from D' in which the input character may belong is selected, next (b) computing statistical measure between the features derived from arranged contour distance of the input character and that of the individual characters of the selected subset S, the input character is recognized.

For statistical measure, we compute the respective differences of the 15 contour distance values of the input character with that of the individual characters of the selected subset S and estimate of variance of the 15 difference values. Let these 15 difference values are the 15 elements of a vector, say, Z_i'. When two characters have similar rearranged contour distance behaviour then all the elements of Z_i' will be similar and hence we will get nearly zero variance from the elements of Z_i'.

For classification of an input character, we select the subset S of D' depending on the number of valleys (valley is discussed earlier) in the contour distance function. Let us assume that the subset S contains contour distance feature for M number of characters. Let $(I_1, I_2, I_3, I_4, I_5, I_6, I_7, I_8, I_9, I_{10}, I_{11}, I_{12}, I_{13}, I_{14}, I_{15})$ be the values of the 15 contour distance of the input character I. Also, let the values of the 15 contour distance of the i^{th} element of the subset S be:

$$(C1_i : C2_i : C3_i : C4_i : C5_i : C6_{i:} : C7_i : C8_i : C9_i : C10_i : C11_i : C12_i : C13_i : C14_i : C15_i), i = 1, 2, \ldots, M.$$

We compute $Z_i', i = 1, 2, \ldots, M$ defined as follows:

$$\begin{aligned} Z_i' = (&C1_i - I_1, C2_i - I_2, C3_i - I_3, C4_i - I_4, C5_i - I_5, C6_i - I_6, \\ &C7_i - I_7, C8_i - I_8, C9_i - I_9, C10_i - I_{10}, C11_i - I_{11}, C12_i - I_{12}, \\ &C13_i - I_{13}, C14_i - I_{14}, C15_i - I_{15}), i = 1, 2, 3, \ldots, M. \end{aligned}$$

The elements of $Z_i', i = 1, 2, \ldots, M$ are the difference of the 15 contour distance values of the input character and that of the i^{th} element of S. All the elements of Z_k' will be similar when the input character and the k^{th} element of S have similar rearranged contour distance function. As a result, variance of the elements of Z_k' will be nearly zero. The character corresponding to the i^{th} element of the subset from which we get minimum variance from the elements of $Z_i', i = 1, 2, \ldots, M$ is the recognized character of the input character.

4. Results and discussion

4.1 *Data*

To compute the accuracy of the proposed recognition scheme, we consider only basic characters of Bangla and Devnagari scripts. For experiment, we consider two popular fonts of both Bangla and Devnagari scripts. For Bangla we consider Satyajit and Shamit fonts, and for Devnagari we consider Jogesh and Nataraj fonts. We also consider 12, 16, 20, 26, 30, 36 and 40 point-size characters for the experiment. We considered 6000 samples (2900 from Bangla and 3100 from Devnagari) for our experiment.

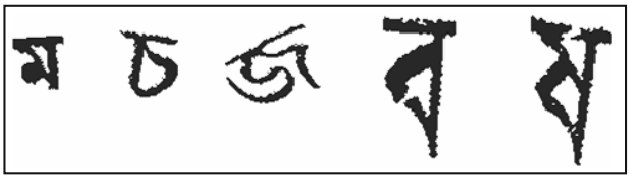


Figure 8. Examples of some noisy characters considered for our experiments and we obtained correct results on these images.

4.2 *Results on character recognition*

From the experiment we noticed that the overall accuracy of the proposed recognition approach was 97·8% in Bangla (98·1% in Devnagari). We also notice that 99·1% (98·9%) accuracy is obtained if we consider first two top choices of the output in Bangla (Devnagari). Note that for template generation of our recognition scheme we use another 400 data. Because of poor quality image if there is a broken part on the contour of a character, our method may not work on the character. To handle degraded or broken images we have smoothed the images using the algorithm due to Roy *et al* (2004). If there is small broken part in the characters this smoothing technique can join the broken part and our proposed method works well. From the experiment we noticed that most of the error occurred in 12 point-size characters. Because of digitization effect and smoothing algorithm some parts of a character of this font sometimes touch to its nearer part and the error occurs in the recognition process. We also noticed that highest accuracy is obtained from bigger point-size characters. For examples, in 12 point-size Bangla characters we obtained 95·8% accuracy whereas in 36 point-size characters we received 98·6% accuracy. Also from the experiment we noticed that some of the errors occur because of the similar structural shape of some of the characters in Bangla/Devnagari script. For an example, two Bangla characters 'খ' and 'ঋ' sometimes generate error because of their similar shape. As mentioned earlier, if there is small break point in a character in some of the degraded documents then our broken part joining algorithm joins this broken part and we get correct results. For example, see figure 8 where degraded images are shown and we got correct results on these images. But if the size of the break point is big (more than the stroke width (R_L) of the character) then broken part joining algorithm cannot join this part and as a results we get erroneous results. The rejection rate of the proposed algorithm is 2·9% in Bangla (2·5% in Devnagari). An input character is rejected if in a subset there are two characters whose variance with input character differs by less than a threshold 2·5. This threshold value is obtained from the experiment. Detail of the recognition results is given in table 1.

The recognition scheme proposed here is independent of size and orientation. To make this system font invariant we have included samples of different fonts in the set D'. We considered 26 point-size character of each class for template feature generation. Since for experiment we consider 12 point to 40 point-size characters, our idea is to use the middle size $[(40+12)/2 = 26]$ font for feature generation.

Table 1. Recognition result based on different choices.

	Accuracy	
Number of choices from top	Bangla	Devnagari
Only 1 choice	97·8%	98·1%
Only first 2 choices	99·1%	98·9.%
Only first 3 choices	99·6%	99·5%

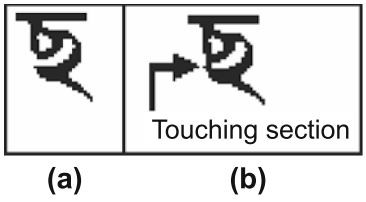


Figure 9. A Bangla character. **(a)** Original structure of the character and **(b)** touching section of the character is shown by arrow.

Since the proposed approach is based on the outer contour of the characters, the characters like '**ষ**' and '**য**' are recognized as same class. We use loop feature for their final classification. We also used a few post-processing techniques for final classification of some other characters.

4.3 *Comparison of results*

There is no work related to the proposed work on any Indian language. So we cannot compare our results on Indian script. However, to get an idea about the comparative results we compare our results with that of Xie & Kobayashi (1991), and Adam *et al* (2000). But these pieces of work are done on English, and characters in English are simpler than the complex shaped Indian script characters. The method due to Xie and Kobayashi was tested on ten numerals and obtained 97% accuracy from the numerals. Adam *et al* (2000) received 97·5% accuracy on English characters. The method due to Adam *et al* use Fourior Merllin transform and hence is time consuming, which is the main drawback of their method. On the other hand, our proposed method is fast as it based only on the contour points of the characters.

4.4 *Drawback*

The main drawback of the proposed method is that it cannot recognize characters properly if some part of a character touches to its nearer part within it. Example of such character is shown in the figure 9. Similarly, if there is a large broken part in the contour of a character which cannot be joined by our smoothing technique then the recognition fails. In both the cases our system will mis-recognize or it will reject the character according to the variance obtained from the feature sets.

5. Conclusion

In this paper, we propose a scheme for isolated character recognition of Indian stylistic documents. The proposed character recognition method does not depend on the size and orientation of the character. The recognition of individual characters is done based on the features obtained from the contour distances calculated from the centroid of the characters. For contour distance-based features, distances of all outer contour points of the component from CG are computed. These contour distances are then arranged in a particular order to get size and rotation invariant feature. Finally, computing statistical features on these ordered contour distances the input character is recognized. We tested our method on the two most popular Indian scripts, Devnagari and Bangla and obtained encourging results. The proposed method can be applied on any of the other Indian scripts. Broken character recognition is the main drawback of our scheme and in future we plan to take care of it.

References

Adam S, Ogier J M, Carlon C, Mullot R, Labiche J, Gardes J 2000 Symbol and Character recognition: application to engineering drawing. *Int. Journal of Document Analysis and Recognition* 3: 89–101

Chaudhuri B B, Pal U 1998 A complete printed Bangla OCR system. *Pattern Recognition* 31: 531–549

Dhandra B V, Nagabhushan P, Hangarge M, Hegadi R, Malemath V S 2006 Script Identification Based on Morphological Reconstruction in Document Images. In *Proc. International Conf. on Pattern Recognition* 950–953

Hase H, Shinokawa T, Yoneda M, Suen C Y 2003 Recognition of Rotated Characters by Eigen-space. In *Proc.* 7th *International Conference on Document Analysis and Recognition* 731–735

Hase H, Yoneda M, Shinokawa T, Suen C Y 2001 Alignment of Free layout colour texts for character recognition. In *Proc.* 6th *Int. Conference on Document Analysis and Recognition* 932–936

Lehal G S, Singh C 2000 A Gurumukhi Script Recognition System. In *Proc. International Conf. on Pattern Recognition* 2557–2560

Loo P K, Tan C L 2002 Word and sentence extraction using irregular pyramid. In *Proc.* 5th *International Workshop on Document Analysis Systems* 307–318

Lu Y, Wang Z, Tan C L 2004 Word Grouping in Document Images Based on Voronoi Tessellation. In *Proc.* 6th *International Workshop on Document Analysis Systems* 147–157

Monwar M, Haque W, Paul P P 2007 A New Approach For Rotation Invariant Optical Character Recognition Using Eigendigit. In *Proc. Canadian Conference on Electrical and Computer Engineering* 1317–1320

Negi A, Chakravarthy B, Krishna B 2001 An OCR System for Telugu. In *Proc.* 6th *Int. Conference on Document Analysis and Recognition* 1110–1114

Pal U, Chaudhuri B B 2004 Indian Script Character Recognition: A Survey. *Pattern Recognition* 37: 1887–1899

Pal U, Roy P P 2004 Multi-oriented and curved text lines extraction from Indian documents. *IEEE Trans. on Systems, Man and Cybernetics- Part B* 34: 1676–1684

Pal U, Kimura F, Roy K and Pal T 2006 Recognition of English Multi-oriented Characters. In *Proc. International Conf. on Pattern Recognition* 873–876

Roy K, Pal U, Chaudhuri B B 2004 A System for Joining and Recognition of Broken Bangla Numerals for Indian Postal Automation. In *Proc. of* 4th *Indian Conference on Computer Vision, Graphics and Image Processing* 641–646

Sato S, Miyake S, Aso H 2000 Evaluation of Two Neocognitron-type Models for recognition of rotated patterns. In *Proc. ICONIP* 295–299

Tang Y Y, Cheng H D, Suen C Y 1991 Translation-ring-projection (TRP) algorithm and its VLSI Implementations. *Character and Handwriting Recognition* Ed. Wang P S P World scientific, Singapore, 25–56

Uchida S, Iwamura M, Omachi S, Kise K 2006 OCR Fonts Revisited for Camera-Based Character Recognition. In *Proc. International Conf. on Pattern Recognition* 1134–1137

Uchida S, Sakai M, Iwamura M, Omachi S, Kise K 2007 Extraction of Embedded Class Information from Universal Character Pattern. In *Proc.* 9th *International Conference on Document Analysis and Recognition* 437–441

Xie Q, Kobayashi A 1991 A construction of pattern recognition system invariant of translation, scale-change and rotation transformation of pattern. *Trans. of the Society of Instrument and Control Engineers* 27: 1167–1174