Word-wise Hand-written Script Separation for Indian Postal automation

K. Roy, U. Pal

Word-wise Hand-written Script Separation for Indian Postal automation

K. Roy
Dept. of Comp. Sc. & Engg.
West Bengal University of Technology,
Sector –1, Saltlake City, Kolkata-64, India

U. Pal
Computer Vision and Pattern Recognition Unit,
Indian Statistical Institute, Kolkata-108, India
Email: umapada@isical.ac.in

Abstract

In a multi-lingual multi-script country like India, a postal document may contain words of two or more scripts. For recognition of this document it is necessary to separate different scripts from the document. In this paper, an automatic scheme for word-wise identification of hand-written Roman and Oriya scripts is proposed for Indian postal automation. In the proposed scheme, at first, document skew is corrected. Next, using a piece-wise projection method the document is segmented into lines and then lines into words. Finally, using different features like, water reservoir concept based features, fractal dimension based features, topological features, scripts characteristics based features etc., a Neural Network (NN) classifier is used for word-wise script identification. For experiment we consider 2500 words and overall accuracy of 97.69% is obtained from the proposed identification scheme.

Keywords: Script separation, Indian script, Multilingual OCR, Handwritten recognition.

1. Introduction

In India there are more than 19 official languages, and 12 scripts are used for these languages. In this multi-lingual multi-script country, a single text line of a document page may contain words of two or more scripts. Optical Character Recognition (OCR) of such a document page can be made in one of the following two options: (1) Development of a generalized OCR system which can recognize all characters of the alphabets of the possible scripts that may present in the document pages (2) Development of a script separation scheme to identify different scripts present in the document pages and then run individual OCR to be developed for each script alphabets. Development of a generalized OCR system for Indian languages is more difficult than a single script OCR development. This is because of large number of characters in each Indian script alphabet. On the other hand, second option is simpler for a country like India because of many scripts. In this paper, we propose an automatic scheme for word-wise identification of hand-written Roman and Oriya scripts from a document image.

There are many pieces of work on script identification from a single document. Spitz [1] developed a method to separate Han-based or Latin-based script separation. He used optical density distribution of characters and frequently occurring word shape characteristics for the purpose. Using cluster based templates, an automatic script identification technique has been described by Hochberg *et al.* [4]. Wood *et al.* [2] described an approach using filtered pixel projection profiles for script separation. Ding *et al.* [3] proposed a method for separating two classes of scripts: European (comprising Roman and Cyrillic scripts) and Oriental (comprising Chinese, Japanese and Korean scripts). Recently, using fractal-based texture features, Tan [5] described an automatic method for identification of Chinese, English, Greek, Russian, Malayalam and Persian text. Using cluster based technique Zhang and All the above pieces of work deal with mainly non-Indian documents. Among Indian script, there are some pieces of work. Pal and Chaudhuri [6] proposed a line-wise script identification scheme from tri-language (triplet) documents. Later, Pal et al. [7] proposed a generalized scheme for line-wise script identification from a single document containing all the twelve Indian scripts. Pal et al. [8] also proposed some work on word-wise identification from Indian script documents. Dhanya and Ramakrishnan [9] proposed a Gabor filter based technique for word-wise segmentation from bi-lingual documents containing English and Tamil scripts.

All the above pieces of work are done for script separation from printed documents. There are many hand-written documents in India where a single document page may contain two or more scripts. For example, two or more scripts may be used to write the address part of an Indian postal document. For the automatic sorting of Indian postal documents it is necessary to identify different scripts by which the postal document is written. Although there are many work on the identification of printed script but there are only two pieces of work for hand-written script identification and these work are done on Roman and Bangla/Devnagari script [13-14]. In this paper a scheme for word-wise identification of hand-written Roman and Oriya scripts is proposed. Because of writing style of different individuals, work on hand-written document is more difficult than printed documents.

In the proposed scheme, at first, the document skew [15] is corrected. Using a piece-wise horizontal projection the document is segmented into lines and by vertical histogram the lines into words. Next by applying water reservoir concept [10] we compute the busy-zone of a word. Finally, fractal based features, water reservoir based features, topological features, scripts based features etc, the Oriya and English script words are identified by a Neural Network (NN) classifier.

2. Properties of Oriya script

The alphabet of the modern Oriya script consists of 11 vowels and 41 consonants. These characters are called basic characters. The basic characters of Oriya script are shown in Figure 1. Writing style in the script is from left to right. The concept of upper/lower case is absent in Oriya script.

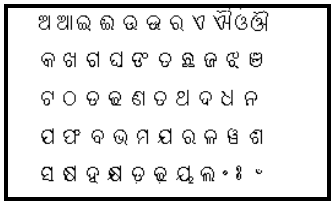


Figure 1. Basic characters of Oriya alphabet. (First 11 are vowels and rests are consonants).

Figure 2. Different zones of an Oriya text line.

In Oriya script a vowel following a consonant takes a modified shape, which, depending on the vowel, is placed at the left, right (or both) or bottom of the consonant. These are called modified characters. A consonant or vowel following a consonant sometimes takes a compound orthographic shape, which we call as compound character. Compound characters can be combinations of consonant and consonant, as well as consonant and vowel.

In Oriya script, a text line may be partitioned into three zones. The *upper-zone* denotes the portion above the mean-line, the *middle zone* (busy-zone) covers the portion of basic (and compound) characters below mean-line and the *lower-zone* is the portion below base-line. An imaginary line, where most of the uppermost (lowermost) points of characters of a text line lie, is referred as mean-line (base-line). Example of zoning is shown in Figure 2. Here mean-line along with base-line partitions the text line into three zones.

From Figure 1 it can be noted that out of 52 basic characters 37 characters have a convex shape at the upper part (near-mean line portion). When two or more characters sit side by side to form a word, these convex parts touch and generate touching characters in most of the cases. For example see Figure 3. From a statistical analysis on 4000 touching components we note that 72% of the touching components touch near mean-line portion because of the convex shape of most of the Oriya characters, 11% of the touching components touch in lower zone and 17% touch mainly in lower half of middle zone. Based on this statistical analysis we have designed the proposed scheme.

Figure 3. Example of Oriya touching character.

3. Preprocessing

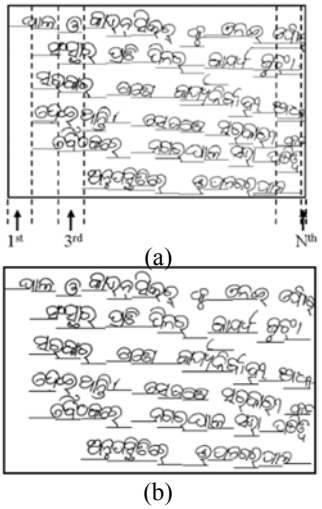
Document digitization for the present work has been done from individual handwritings as well as from the address part of some Indian postal documents. The images are in gray tone and digitized at 300 dpi. We have used a two-stage approach to convert the images into two-tone (0 and 1). In the first stage a pre-binarization [11] is done using a local window based algorithm in order to get an idea of different regions of interest. On the pre-binarized image, Run Length Smoothing Algorithm (RLSA) is applied. After this, using component labeling, we select each component and map them in the original grey image to get respective zones of the original image and the final binarized image is obtained using histogram based global binarization algorithm [11] on these regions of the original image. We noted that this thresholding technique is useful both in postal documents as well as normal text documents. The digitized document images may be skewed and we used the algorithm due to Shi and Govindaraju [15] to de-skew the documents.

4. Line and Word Segmentation

The global horizontal projection method computes sum of all black pixels on every row and constructs corresponding histogram. Based on the peak/valley points of the histogram individual lines are generally segmented. Although this global horizontal projection method is applicable for line segmentation of printed documents but it cannot be used in unconstrained hand-written documents because the characters of two consecutive text-lines may be touched or overlapped. For example, see the document shown in Figure 4(a). Here, two consecutive text lines are mostly overlapped. To take care of unconstrained hand-written documents here we use a piece-wise projection method. In this method we divide the text into vertical stripes of width W (here we assume that a document page is in portrait mode). Width of the last stripe may differ from W. If the text width is Z and the number of stripe is N then the width of the last stripe is [Z-W*(N-1)]. Computation of W is discussed latter. Next we compute Piece-wise Separating Lines (PSL) from each of these stripes. We compute row-wise sum of all black pixels of a stripe. The row where this sum is zero is a PSL. We may get few consecutive rows where sum of all black pixels is zero. Then the first row of such consecutive rows is the PSL. The PSLs of different stripes of a text are shown in Figure 4(a) by horizontal lines. All these PSLs may not be useful for line segmentation. We choose some potential PSLs as follows. We compute the normal distances between two consecutive PSLs in a stripe. So if there are *n* PSLs in a stripe we get *n-1* distances. This is done for all stripes. We compute the statistical mode (M_{PSL}) of such distances. If the distance between any two consecutive PSLs of a stripe is less than M_{PSL} we remove the upper PSL of these two PSLs. PSLs obtained after this removal are the potential PSLs. The potential

PSLs obtained from the PSLs of Figure 4(a) are shown in Figure 4(b). By proper joining of these potential PSLs we get individual text lines. We use similar technique due to Datta & Pal [16]. For detail about line segmentation see this paper.

An example of line segmentation technique is shown in Figure 5. It may be noted that sometimes because of overlapping or touching of one component of upper line with a component of its lower line, we may not get PSLs in some regions. Also, because of some modified characters of Oriya we find some extra PSLs in a stripe. We have taken care of them during PSLs joining.



(a)

(b)

Figure 4: (a) N-stripes and PSL lines in each stripe are shown for a Oriya hand-written text. (b) Potential PSLs of Figure 4 (a) are shown.

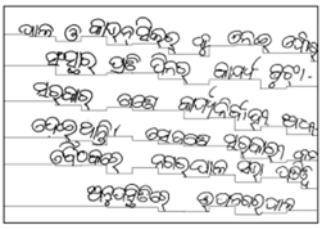
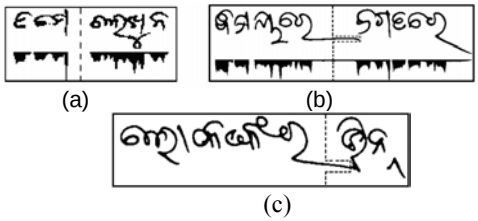


Figure 5. Line segmented result of the image shown in Figure 4. Text line segmentation is shown by solid line.

To get size independent measure, computation of W is done as follows. We compute the statistical mode (R_L) of the widths of the bottom reservoirs obtained from the text. This mode is generally equal to character width. Since average character in an Oriya word is four, the value of W is assumed as $4*R_L$ to make the stripe width as word width.

For word segmentation from a line, we compute vertical histogram of the line. In general the distance between two consecutive words of a line is bigger than the distance between two consecutive characters in a word. Taking the vertical histogram of the line and using above distance criteria we segment words from lines. For example see Figure 6(a). Because of the characteristics of some Oriya characters in some of the handwritings we may not get enough white space between two consecutive words. For example see Figure 6(b). Here we cannot get big gap between two words because of the long horizontal line-like structure of the rightmost character of the left word. For the segmentation of such words we use a modified technique described as follows. We scan each column of a text line staring from the top row of the line. Also, for each column we note the position of topmost black pixel. Similarly, each column is scanned from bottom and the position of it bottommost pixel is noted. If for a column the distance between the top and bottommost points is less than $2*S_L$ then we mark that column as zero (white). Else it is marked as one (black). S_L is the stroke width of the word. So after completion of column scanning we get a row of ones and zeros. If the length of a run of zero is greater than W (value of W is computed earlier) then we assume that this run corresponds as a separator of two words and the midpoint of this run is noted. If the column corresponding to midpoint of the image is white then we consider that column as word boundary. If there is any black pixel of a component in the column corresponding to midpoint, starting from the topmost black pixel the boarder of component is traced clockwise and the path obtained by this tracing is considered as the separator of the two words. For example see Figure 6(b). Here, the segmentation path is marked by dots. Sometimes because of handwriting styles two consecutive words may touch. For example, see Figure 6(c). For such cases, from the topmost black pixels of the segmented line we clockwise trace the boarder of the component to find an obstacle point for segmentation. During tracing, the length of vertical black run at each tracing point is computed. The boundary point where this run length is greater than $1.5*S_L$ is considered as obstacle point. We disconnect the touching by replacing black pixels of the vertical run where obstacle point lies. The path obtained by tracing along with the disconnected portion is considered as the separator of the two words.



(a) (b)

(c)

Figure 6: Example of word segmentation. Segmentation is shown by dotted line. (a) Two words having enough space between them. (b) Two words without enough space between them. (c) Two touching words.

5. Water Reservoir Principle and Busy-zone Detection

Water reservoir principle: The principle of water reservoir 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 [10]. **:**

Water reservoir principle is used here to compute the busy-zone of an image and for script identification.

By top (bottom) reservoir of a component we mean the reservoirs obtained when water is poured from top (bottom) of the component. A bottom reservoir of a component is visualized as a top reservoir when water will be poured from top after rotating the component by 180°. Examples of top and bottom reservoir are given in figure 7.

When two or more characters sit side by side to form a word, the convex parts of Oriya characters touch and generate touching characters in most of the cases. We use water reservoir technique to compute this convex part and these water reservoirs are very helpful for script identification. Some people write words in isolated fashion where characters in a word do not touch each other. As a result, water reservoirs are not generated and hence water reservoir feature based scheme may not work. To take care of this situation we join these isolated characters of a word. Joining procedure of the characters of a word is as follows. Here we first take one component of the word image and grow it around its boundaries until it touches its nearest neighbour component. This touching point is noted for joining. Let this point be X_1. Now from this point we draw a circle of radius equal to number of times the component was grown to search the neighbour component. The point where the circle touches the first component is noted. Let this point be X_2. Two points X_1 and X_2 are joined with width equal to the stroke width to get touching component. For illustration see Figure 8. Above process is repeated until all the components of a word get connected. Example of isolated characters of an Oriya word and its joined version is shown in Figure 9.

Figure 7. (a-b) Top and bottom reservoirs (in gray) are shown. (a) English word (b) Oriya word.

Figure 8. Joining of isolated characters in a word image is shown (a) A word containing two isolated characters. (b) The grown version of the left component and touching point (X_1) in the right component is shown. (c) From X_1 the point (X_2) on the first component is shown for joining. (d) Joined version of the word.

Computation of Busy-zone: Busy-zone of a word is the region of the word where maximum parts of its characters lie. We use reservoir property to detect the busy-zone. Here at first all top and bottom reservoirs are detected from the characters of a word. We calculate the average height of all the top and bottom reservoirs. For all top and bottom reservoirs whose height is less than 1.25 times of the average reservoirs height, we fill them by black pixel. Also all the loops are filled up with black pixel before computing the busy-zone. Filled-up version of the image of Figure 10(a) is shown in Figure 10(b). After filling of the reservoirs and loops with black pixel we compute the busy-zone height of this filled-up image as follows. At first we compute horizontal projection profile on this filled-up image. We draw a vertical line through the midpoint of the width of the horizontal projection profile. Let this line be XY as shown in Figure 10(c). The portion of the XY that belongs to projection profile are marked by p and q. The distance between p and q is the busy-zone height of a word image. The region within the horizontal lines passing through p and q gives us the busy-zone. Busy-zone of the word shown Figure 10(a) is shown in Figure 10(d). Area of top and bottom reservoirs in this busy-zone area plays an important role in our script identification approach.

6. Feature Extraction and Recognition

6.1. Feature Extraction

Feature selection is a very important step for any recognition scheme. They should be robust and easy to compute. In the Neural Network based classification of our proposed scheme, we have used mainly the following features. (i) Fractal based feature, (ii) Water reservoir based feature, (iii) Presence of small component, (iv) Topological features, etc.

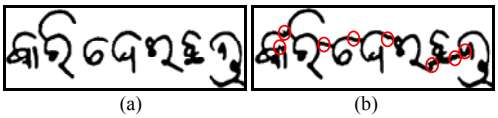


Figure 9. Joining of isolated characters in an Oriya word image is shown (a) Example of a word with some of its isolated characters. (b) Joined version of the word. The portions where joining is done are marked by small circles.

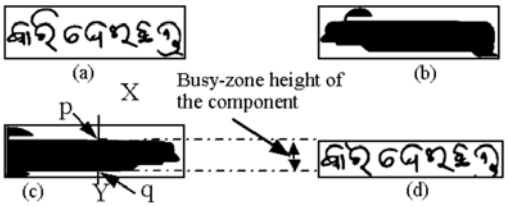


Figure 10(a) Original Image, (b) Filled-up version of the image (a), (c) Computation of busy-zone, (d) Busy-zone area of the image (a).

Fractal based feature: A fractal [12] is defined as a set for which the Hausdorff-Besikovich dimension is strictly larger than the topological dimension. The fractal dimension is a useful method to quantify the complexity of feature details present in an image. The fractal dimension is an important characteristic of the fractals because it contains information about their geometric structures. When employing fractal analysis researchers typically estimate the dimension from an image. Many researchers used fractal dimension in pattern recognition

work. Here our idea is to use of fractals feature in script identification.

The fractal theory developed by Mandelbrot and Van Ness was derived from the work of mathematicians Hausdorff and Besikovich. The Hausdorff-Besikovich dimension, D_H, is defined as:

$$D_H = \lim_{\varepsilon \to 0^+} \frac{\ln N_\varepsilon}{\ln 1/\varepsilon}$$

where N_ϵ is the number of elements of ϵ diameter required to cover the object.

When working with discrete data, one is interested in a deterministic fractal and the associated fractal dimension (D_f) which can be defined as the ratio of the number of self-similar pieces (N) object—objects whose dimensionality is integer valued. However, the surfaces of many objects cannot be described with an integer value. These objects are said to have a ''fractional'' dimension. Consider the following example to illustrate the magnification factor (1/r) into which an image may be broken. D_f is defined as:

$$D_f = \frac{\ln N}{\ln 1/r}$$

D_f may be a non-integer value, in contrast to objects lying strictly in Euclidean space, which have an integer value. However, D_f can only be directly calculated for a deterministic fractal. There are a variety of applicable algorithms for estimating D_f, and we have used Box-counting algorithm for the same. Here we have used the fractal dimensions of the full image, its full contour, upper contour and lower contour of the image for our script identification purpose. The original image, its full contour, upper and lower contour are shown in Figure 11. The fractal dimension for the above images (full image, full contour, upper and lower contour) for the English word (shown in Figure 11(a)) are 1.29576, 1.26316, 1.18885, 1.09732 and for the Oriya word (shown in Figure 11(b)) are 1.48137, 1.34867, 1.17161, 1.26105 respectively. From the Figure it can be seen that for the case of Oriya words the upper contour is smoother than lower contour and there is zigzags structure in the lower contour. But English has different characteristics. In English lower contour is smoother than upper contour. This is because while writing in English the characters generally touch each other at their lower part and making lower part smoother than that of the upper one. But in case of Oriya the characters in a word mostly touch in the upper part and thus creating a smooth structure in the upper part than that of the lower part. As we intend to use Neural Network for the classification, we have used the property of fractal dimension to convert these characteristics of the word images into feature (as for an image the fractal dimension is always between 1 and 2 and subtracting one we can easily get the normalized feature factor of the same). We noticed that fractal feature plays an important role in our script separation scheme. We have seen that in the absence of the Fractal feature accuracy of our script separation scheme drops about 10%.

Water reservoir based feature: In English script it can be found that there are many big top reservoirs in the characters of a word. On the other hand Oriya script word has many big bottom reservoirs. For example see Figure 12 (a-b). The ratio of the area of the top reservoir to that of bottom reservoir of a word image is used as feature. Here we note that the value of this feature is greater than one for English and less than one for Oriya. Another distinct feature of Oriya and English is that the base-lines of most of the bottom reservoirs lie in the lower half of the busy-zone for English. For Oriya it is in the upper part as can be seen from Figure 12(a). By base-line of a reservoir we mean the line passing through the average of all the base points (the deepest point of the reservoir) of the reservoir. We have used 6 features based on busy-zone.

Figure 11. Examples of some images for Fractal feature computation (a) English (b) Oriya (i) Original image (ii) Full contour (iii) Upper contour (iv) Lower contour.

Small component based feature: Here we take all the components whose height and width is less than twice of the stroke width (mode of the horizontal run-length of the image) and compares their position with respect to the busy-zone. If such components lie completely above or bellow the busy-zone then those component number and position is used as a feature. This feature is selected based on the characteristics of scripts. For example, in English we find some characters with disjoint upper part. Such positional information of small components are used for script identification.

Topological features: The topological features such as aspect ratio (ratio of length and width), area of loop, maximum length of the horizontal and vertical black run, the statistical modes of horizontal and vertical black run etc have also been used in our script separation scheme. We have used 9 features based on topological features of the image.

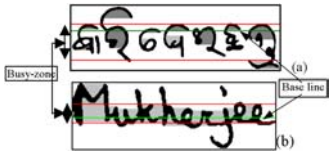


Figure 12. Water reservoir based feature. (a) Oriya word, (b) English word.

6.2. Recognition

Considering all the above-mentioned features a set of 25 features is generated. All the features stated above are normalised before feeding them to neural network. We have used a Multilayer Perceptron Neural Network based scheme for identification of hand-written Oriya

and English script. Here we have used a 2-layer NN with number of neurons in input and output layers as 25 and 2 respectively since the number features is 25 and the number of possible classes in hand-written script selected for the present case is 2. The number of hidden units as 10, Back propagation learning rate and acceleration factor is set to suitable values, based on trial runs. A network of 25-10-2 is thus finally designed.

7. Results and Discussion

In the experiment of our script identification scheme we have used a database of 2500 (1200 Oriya and 1300 English) hand-written words collected from postal data as well as from some individuals. Out of these data a set of 1500 (750 Oriya and 750 English) words are used for training and the rest of the dataset is used for the purpose of testing for current experiment. From the experiment we obtained 99.6% accuracy in the training data and 97.69% in the test data. The identification results are shown in Table 1. The confusion matrix of the identification results obtained from the test data is shown in Table 2.

Table 1: Word-wise script identification results.

Data set	Correct Identification	Error
Training	99.6	0.4
Test	97.69	2.31

Table 2: Confusion matrix of the identification results.

Script / Recognised as	Oriya	English
Oriya	96.92	3.08
English	0.84	99.16

The main sources of mis-recognition are small words and poor quality of postal documents. Due to presence of some small components in the upper part of the busy-zone of Oriya script, sometimes Oriya script is mis-recognized as English script by the proposed system. From the experiment we note that the proposed system generates more errors when characters are mostly in isolated form in a word. Some mis-recognised words are shown in Figure 13. The image shown in Figure 13(a) is an Oriya word but it is recognized as English. This is because of the smaller length of the word (word length is two here as there are only two characters in this word). Also, the first character of this word looks similar to English character 'v'. The image shown in Figure 13(c) is an English word but it is recognized as Oriya. This is so because most of the characters in this English word touch at the upper part of the word (in handwritten English text, characters generally touch at the lower part).

The propose scheme is independent of text size. Also, there is no need of any normalization of the image in the proposed technique. This is the main advantage of the proposed scheme. Our propose scheme will also work for similar Indian scripts and we are also planning to use this scheme as a general script separation module for the development of multi-script OCR for the multi-lingual country like India.

(a) (b) (c)

Figure 13. Examples some of mis-classified words. (a-b) Oriya words identified as English (c) English word identified as Oriya.

References

[1] A. L. Spitz, "Determination of the script and language content of document images", *IEEE Trans. on PAMI*, vol. 19, 1997, pp. 235-245.

[2] S. Wood, X. Yao, K, Krishnamurthi and L. Dang, "Language identification for printed text independent of segmentation", *In Proc. Int'l Conf. on Image Processing*, 1995, pp. 428-431.

[3] J. Ding, L. Lam and C. Y. Suen, "Classification of oriental and European scripts by using characteristic features", *In Proc. 4th ICDAR*, 1997, pp. 1023-1027.

[4] J. Hochberg, P Kelly, T Thomas and L. Kerns, "Automatic script identification from document images using cluster-based templates", *IEEE Trans. on PAMI*, vol. 19, 1997, pp. 176-181.

[5] T. N. Tan, "Rotation invariant texture features and their use in automatic script identification", *IEEE Trans. on PAMI*, vol. 20, 1998, pp. 751-756.

[6] U. Pal and B. B. Chaudhuri, "Script line separation from Indian multi-script documents", *IETE Journal of Research*, vol. 49, 2003, pp. 3-11.

[7] U. Pal, S. Sinha and B. B. Chaudhuri, "Multi-Script Line identification from Indian Documents", *In Proc. 7th ICDAR*, 2003, pp. 880-884.

[8] S. Chanda and U. Pal "English, Devnagari and Urdu text identification", *Proc. Int. Conf. on Cognition and Recognition*, 2005, pp. 538-545.

[9] D. Hhanya, A. G. Ramakrishna and P. B. Pati, "Script identification in printed bilingual documents", *Sadhana*, vol. 27, part-1, 2002, pp. 73-82.

[10] U. Pal and P. P. Roy, "Multi-oriented and curved text lines extraction from Indian documents", *IEEE Trans. On SMC - Part B,* vol. 34, 2004, pp.1676-1684.

[11] K. Roy, S. Vajda, U. Pal, and B. B. Chaudhuri, "A System towards Indian Postal Automation", *In Proc. of IWFHR-9*, 2004, pp. 266-271.

[12] B. B. Mandelbrot, *The Fractal Geometry of Nature*, Freeman, NY, 1983.

[13] L. Zhou, Y. Lu and C. L. Tan, "Bangla/English script identification based on analysis of connected component profiles", *In Proc. of 7th IAPR Workshop on DAS*, 2006, pp. 243-254.

[14] K. Roy, A. Banerjee, and U. Pal, "A System for Word-wise Hand-written Script Identification for Indian Postal Automation", *IEEE India Conference*, 2004, pp. 266-271.

[15] Z. Shi and V. Govindaraju, "Skew Detection for Complex Document Images/using Fuzzy Runlength", *In Proc. of 7th ICDAR*, 2003, pp. 715-719.

[16] U. Pal and S. Datta, "Segmentation of Bangla Unconstrained Handwritten Text", *In Proc. 7th ICDAR*, 2003, pp. 1128-1132.