Image and Vision Computing 20 (2002) 945–954

Identification of different script lines from multi-script documents

U. Pal, B.B. Chaudhuri*

Computer Vision and Pattern Recognition Unit, Indian Statistical Institute, 203 B.T. Road, Calcutta 700 035, India

Accepted 11 July 2002

Abstract

For wider readership, some documents may be printed in several scripts and languages. For optical character recognition (OCR) of such a document page, a software module is necessary to identify the scripts before feeding them to their individual OCR systems. This paper deals with an automatic technique for the identification of printed Roman, Chinese, Arabic, Devnagari and Bangla text lines from a single document. For this purpose script characteristics, shape-based features, statistical features and some features obtained from the concept of water overflow from the reservoir have been employed. The scheme shows an accuracy of about 97.33%.

Keywords: Optical character recognition; Script lines; Head-line

1. Introduction

For wider readership, a single document page may contain several language scripts. To develop a multi-script optical character recognition (OCR) system it is necessary to separate different script forms before feeding them to the individual script recognizers. This is so because development of hybrid OCR for multiple scripts is more difficult than separate OCRs of individual scripts.

Among the pieces of earlier work of different script separation, one of the first attempts to this problem is due to Spitz [1,2]. In 1990, he described a method for classification of individual text from a = document containing English and Japanese text [1]. Later on he [2] proposed a technique for distinguishing Han and Latin based scripts on the basis of spatial relationships of features related to the character structures. Identification within the Han-based script class (Chinese, Japanese, Korean) is performed by analysis of the distribution of optical density in text images. Latin-based script are identified using a technique based on character shape codes, and finding the frequency of language specific patterns. Among others, Wood et al. [3] described an approach using filtered pixel projection profiles. Ding et al. [4] proposed a method for separating two classes of scripts: European (comprising Roman and Cyrillic scripts) and Oriental (comprising Chinese, Japanese and Korean scripts). An identification scheme using cluster-based template of the scripts is proposed by Hochberg et al. [5]. Here, templates for each script are created by clustering textual symbols from a training set. Script identification is then done by comparing textual symbols of an unknown document with these templates. Using rotation invariant multi-channel Gabor filter texture features Tan [6] described a method for the identification of Chinese, English, Greek, Russian, Malayalam and Persian text. Zhang and Ding [7] presented a cluster based bilingual script (English and Chinese) segmentation approach. We [8] have developed an automatic scheme for the text line identification from document containing Indian script triplets. Script characteristics, shape based features and statistical features are used for the purpose.

This paper describes an automatic identification scheme of different script lines from a printed document containing five most popular scripts in the world, namely Roman, Chinese, Arabic, Devnagari and Bangla, which rank the first five positions in terms of popularity. Of them, Devnagari and Bangla are the first and second-most important scripts in India. Briefly, our method is as follows. At first, the head-line information is used to separate Devnagari and Bangla script lines in one group and English, Chinese and Arabic script lines into another group. Next, from the first group Bangla script lines are distinguished from Devnagari using some structural features. Similarly, using vertical run

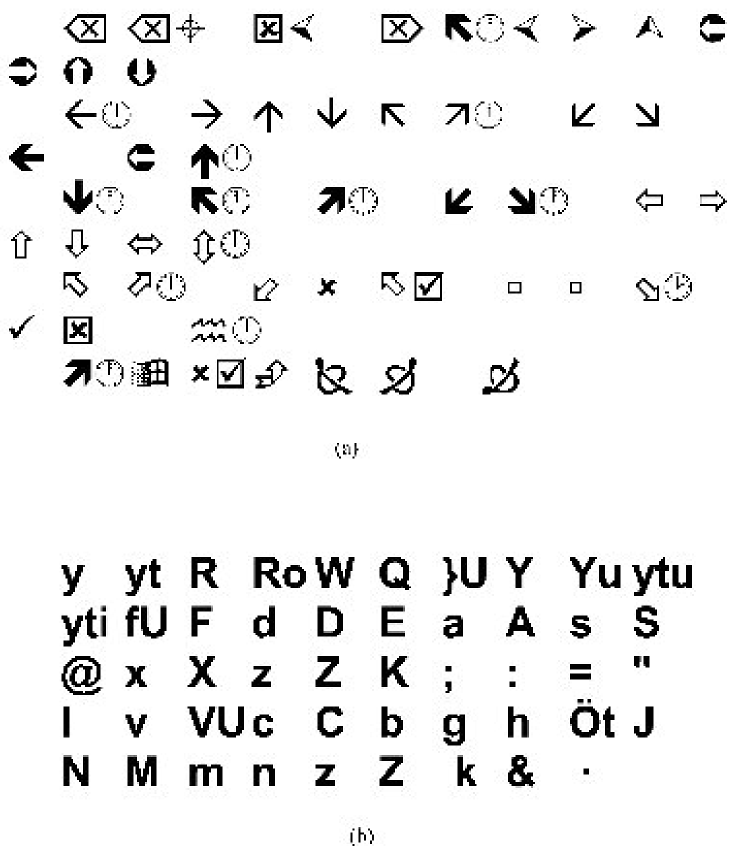

Fig. 1. Basic characters of (a) Bangla and (b) Devnagari alphabet. The first 11 characters are vowels and rest are consonants in both alphabets.

number and run length smoothing, the Chinese text lines have been identified from the second group and then using feature obtain from the concept of water overflow analogy from reservoir as well as statistical features, English text lines have been separated from Arabic.

The rest of the paper is organized as follows: In Section 2 the distinctive properties of the scripts are described. The preprocessing technique is discussed in Section 3. Section 4 deals with the features used as well as script identification scheme in more details. Finally, results and discussions are provided in Section 5.

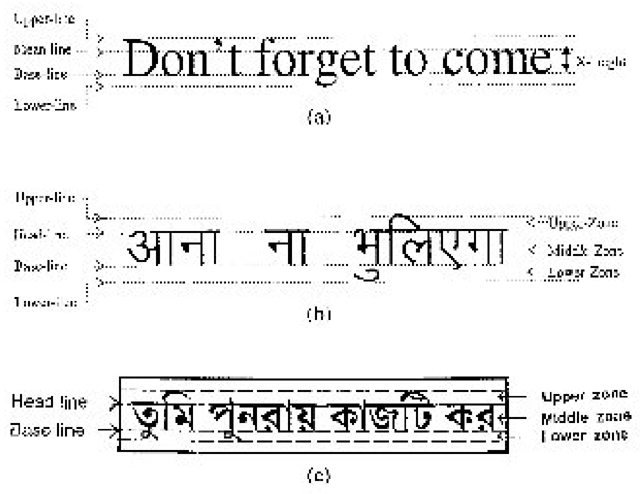

Fig. 2. Different zones of (a) English, (b) Devnagari and (c) Bangla script lines.

2. Distinctive properties of the scripts

There are 11 vowels and 39 consonant characters in modern Bangla alphabet. They are called basic characters (shapes). In Devnagari there are 49 basic characters. Out of these 49 basic characters 11 are vowels and 38 are consonants. Examples of basic characters of Bangla and Devnagari alphabet are shown in Fig. 1. In Bangla and Devnagari script it is noted that many characters have a horizontal line at the upper part. In Bangla, this line is called *matra* while in Devnagari it is called *shirorekha*. However, in this paper, we shall call it as *head-line*. When two or more characters sit side by side to form a word in these scripts, the head-line portions touch one another and generate a long head-line (for example see Fig. 2(b) and (c)). This characteristics is used as one of the features for script identification because horizontal scanning histogram shows a sharp peak at this head-line region.

In Arabic alphabet, there are 28 basic characters. These characters may change their shapes according to their positions (beginning, medium, end or isolated) in the word. Each character can take up to four different shapes. The use of special stress marks called diacritics is another

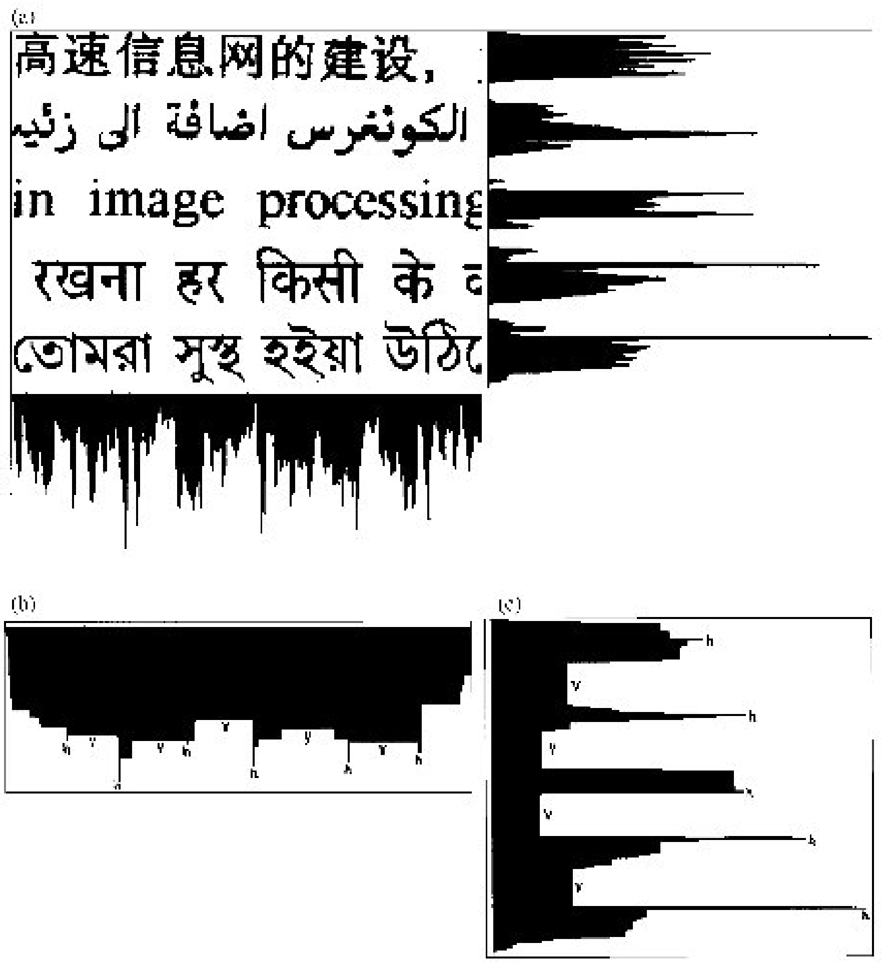

Fig. 3. (a) Horizontal and vertical projection profiles of a text document. (b) Smoothed version of the vertical projection profile. (c) Smoothed version of the horizontal projection profile. (Here 'h' and 'v' denote the top-most hill point and bottom-most valley point of each smoothed hill and valley region).

characteristics of Arabic. Most of the Arabic characters have one, two or three diacritics. These diacritics can be situated at the top or bottom of the characters. In addition to the 28 basic characters Arabic language has some additional characters. An important feature of this script, from the script identification point of view, is the non uniformity of characters size. The character size varies according to its position in the word. Sometimes, in Arabic script a long head-line may occur, but the occurrence position of the head-line is different from that of Bangla or Devnagari.

There are approximately 400 basic components and some 20 essential strokes in Chinese script. These numbers are much less than the total number of Chinese characters, which is over 50,000. There are several basic formation rules for Chinese characters. A Chinese character is formed by a number of components of simple structures. In general, Chinese character shapes are very complex. This complex behavior helps us to separate Chinese script from others.

There are 26 letters in English alphabet. Each letter has two different forms: small and capital. Since English is well known we are not giving details of it.

The direction of writings is from left to right for all scripts excepting the Arabic, which is from right to left. Except English, the scripts considered here are case independent. We identify a text line as English upper-case if upper-line and lower-line coincides with mean-line and base-line, respectively. By mean-line (base-line) of a text line we mean the imaginary horizontal line which contains most of the uppermost (lowermost) points of the characters of the line. We call the uppermost and lowermost boundary lines of a text line as upper-line and lower-line (Fig. 2). From the structural shape of the script characters we notice that partitioning of Chinese and Arabic text lines into three zones is very difficult while English text line can easily be partitioned into three zones. The *upper-zone* denotes the portion above the mean-line, the *middle zone* covers the portion between mean-line and base-line, the *lower-zone* is the portion below base-line. Bangla and Devnagari text lines also can be partitioned into three zones. Different zones in English, Devnagari and Bangla text lines are shown in Fig. 2. The shapes of zone-wise features in Bangla and Devnagari are not similar. These distinguishable shapes are used as the features for the identification of Bangla and Devnagari script lines.

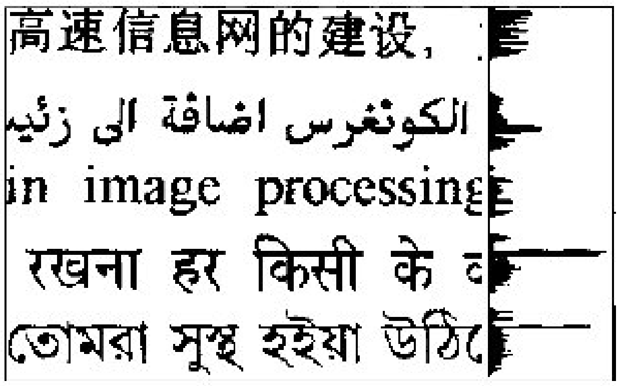

Fig. 4. Row-wise longest horizontal run. (From top to bottom: Chinese, Arabic, English, Devnagari and Bangla text lines).

3. Preprocessing

Text digitization for the experiment of the system has been done by a flatbed scanner (manufactured by HP, Model no ScanJet 4C). The digitized images are in gray tone and we used a histogram based thresholding approach to convert them into two-tone images. The digitized image may contain spurious noise pixels and irregularities on the boundary of the characters and we used a simple and efficient method due to Mahmoud [9] for removing them.

For accurate text classification, the system should properly detect individual text blocks (columns) and should accurately segment the lines from each text block. For text block detection a run length smoothing approach due to Wang et al. [10] is used.

Next the mode of the text block (portrait or landscape mode) of the document is determined. A text block is in portrait (landscape) mode if the text lines in that block are horizontal (vertical). For mode detection we use the property that the white space between characters is much smaller than that between the lines. We compute the horizontal and vertical projection profiles of a text block. In a projection profile, a text line will appear as a black peak and a white stream between two text lines will appear as a valley. A spatial feature, called Peak–Valley–Distance (PVD), is extracted to estimate the orientation of the text block. To find the peak and valley points, the projection profiles are smoothed by a run-length based approach. To smooth the horizontal profile, white runs having length less than a threshold T (computation of this threshold value is discussed in next paragraph) is changed into black. Vertical profile is also smoothed in a similar way. The smoothed versions of the vertical and horizontal profiles of Fig. 3(a) are shown in Fig. 3(b) and (c). In each smoothed peak and valley region, the top-most point (h) and bottom-most point (v) are noted (see Fig. 3(b) and (c) where these points are shown). Let h_1 and h_2 are heights of two consecutive peaks and v_1 be the height of the valley between these two peaks (height of a peak means the length of the top-most point of the peak from the base, and height of the valley means the length of the bottom-most point of the valley from the base. Here, by base we mean the line on which the projection profiles are drawn). We compute the PVD as follows:

$$\mathrm{PVD} = (h_1 + h_2)/2 - v_1$$

We compute average PVD values for both horizontal and vertical profiles. Let these values be W_h and W_v, respectively. We decide that the mode of the text block is portrait if $W_h > W_v$. Otherwise, the orientation is landscape. In the rest of the paper, we assume the text mode is portrait.

The value of threshold T can be estimated as follows. For most printed text documents, the character size is not smaller than 9 points. Then, the minimum spacing between two lines is 9 points. If the document is digitized at P dpi then the minimum distance between two text lines will be $P \times 9/72$ pixels $= P/8$ pixels (since 72 points $= 1$ in.). For a document digitized at 300 dpi we have $T = 37$.

After text block mode detection, the system should accurately detect individual text lines from a text block. Lines of a text block are segmented by finding the valleys of the projection profile computed by row-wise sum of pixel values. The position where profile height is the least denotes one boundary line. A text line can be found between two consecutive boundary lines. For the experiment we consider documents having nearly zero skew angle.

4. Separation of different script lines

If we take the longest horizontal run of black pixels on the rows of a text line then such run length for Bangla or Devnagari line is usually larger than that of other scripts because of connecting tendency of characters through the head-line. For illustration, see Fig. 4. Where fourth line is Devnagari and last line is Bangla. This run length information has been used to isolate the Bangla and Devnagari text lines from others.

To test the likelihood of long head-line we note from Fig. 1 that out of 50 basic characters in Bangla there are 32 characters with head-line while in Devnagari out of 49 basic characters 42 characters have head-line. We computed the character occurrence statistics in Bangla language [11] which indicates that out of twelve most frequent characters, only one has no head-line. We can make a better quantitative analysis of the presence of head-line in a word as follows. According to our statistics [11], the average length of Bangla words is about 6 characters. We noted vowel modifiers (in Bangla and Devnagari some vowels may take a modified shape, which depending on the vowel is placed to the left, right, top or bottom of the consonant. These modified shapes are called modifiers) are small in width and contribute very little to the head-line of the word. The occurrence percentage of vowel modifiers is about 30%. Also, we noted that compound characters are very infrequent, occurring in 5% cases only. (Sometimes two or more characters combine and generate a complex

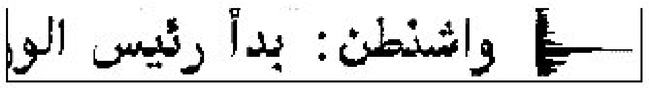

Fig. 5. Example of an Arabic text line where a long horizontal run is obtained. Note that here long horizontal run appears at the lower-half of the text line.

shape in both Bangla and Devnagari. The resultant shape may be called as *compound character*, see Refs. [12,13]).

Since vowel modifiers contribute very little to the head-line, we assume that on an average four basic characters only contribute to the head-line in a word. We also assume that each character is equally likely in a word. In Bangla 41 characters can appear in the first position of a word. Out of these 30 characters have head-line. So probability of getting a character with head-line in the first position of a word is $p_1 = 30/41$. Hence, probability of getting a character without head-line in the first position is $1 - p_1 = 11/41$. Since modifiers are small in width and basic vowel characters mostly occur in the first position of the word, the characters which can contribute to the head-line in other positions of a word are mostly consonants. Since 28 out of 39 Bangla consonants have head-line, the probability of getting a consonant with head-line for other positions in a word is $p_2 = 28/39$. Then probability of getting a character without head-line in other positions is $1 - p_2 = 11/39$.

Thus, probability of all four characters without head-line in a word is $(1 - p_1)(1 - p_2)^3 = 0.006$ (assuming that all characters are equally likely and independently occurring in a word). Hence, probability that a word will have at least one character with head-line is $1 - 0.006 = 0.994$. Analyzing in the same way we note the corresponding probability for Devnagari is 0.997. The practical situation is better than these estimates since characters are not equally likely in a word and most frequently used characters have head-line. Hence, the use of head-line is justified for text classification.

At first, we use this long head-line feature for classification. If the length of the longest horizontal run in a text line is greater than a threshold T_1 then decide it as a Bangla or Devnagari line. Otherwise, it is assigned to English, Arabic or Chinese group. From some experiments the value of T_1 is chosen as the twice of the text-line middle zone height. Sometimes, we can get a longer horizontal run in cursive Arabic text line when two or more characters get connected. For example see Fig. 5. This line may be wrongly identified as Bangla or Devnagari, but we note that the longer run for Arabic script appears in the lower half of the text line, whereas it appears in the upper half of the text line (near head-line) in Bangla and Devnagari. Thus, we can exclude an Arabic text line from the group of Bangla and Devnagari.

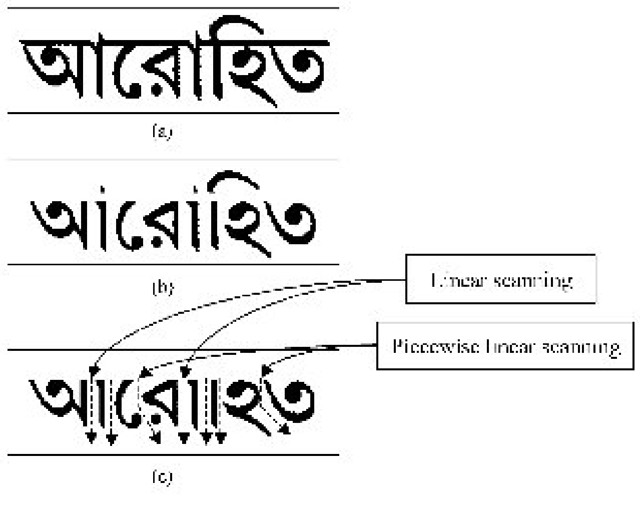

Fig. 6. Character segmentation approach (a) A Bangla word, (b) situation after deletion of head-line area from the word and (c) linear or piecewise linear scanning for character segmentation.

4.1. Separation of Bangla and Devnagari text lines

Separation of Bangla and Devnagari text lines is tricky because of their structural similarity and we use some zone-wise character level features for their separation. To do so, segmentation of text line into words and word into characters is necessary. The word and character segmentation approach is as follows:

Word and character segmentation: After a script line is segmented, it is scanned vertically. If in one vertical scan, two or less black pixels are encountered then the scan is denoted by 0, else the scan is denoted by the number of black pixels. In this way, a vertical projection profile is constructed. Now, if there exist at least k_1 consecutive zero values in the profile then the midpoint of that run of 0's is considered as the boundary of a word. The value of k_1 is a function of text line height.

The method of character segmentation is as follows. The characters in a Bangla or Devnagari word are mostly connected through the head-line, and our idea is to rub the head-line from the middle zone of the word to break the connection. An estimate of average thickness t of character strokes is found from the run-length statistics of black pixels. From the onset of head-line, t scan-lines are horizontally rubbed off from the middle zone to make the characters topologically disconnected, as shown in Fig. 6(b).

Now starting from the headline a scan is initiated in vertical direction to segment the characters. If during a scan, one can reach the base-line without touching any black pixel then this scan marks a boundary between two characters. Sometimes *kerned characters* (kerned characters are the characters that overlap with neighboring characters) may be present in the script for which a piece-wise linear scanning method [13] is invoked (see Fig. 6(c) for illustration).

In actual practice the head-line is not rubbed off to segment the characters. The lowermost row of the head-line is noted. A linear or piece-wise linear scan (as shown in Fig. 6) starting from this noted row in the middle zone will segment the characters.

Feature selection and detection: We considered a few

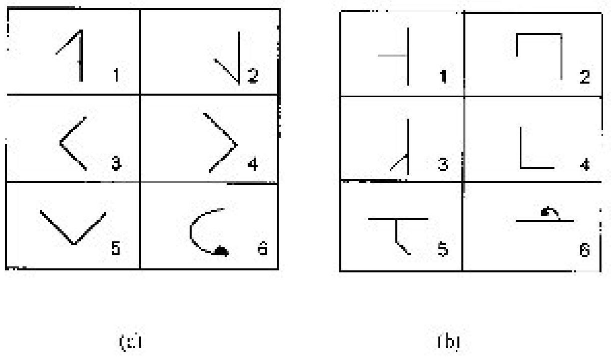

Fig. 7. Stroke features used for (a) Bangla and (b) Devnagari text lines identification.

stroke features for identification of Bangla and Devnagari text lines. In Fig. 7 the *principal stroke features* chosen for the separation are shown. Apart from these principal feature some other features are also used for classification. The features are chosen with the following considerations (a) Presence in characters of one script and absence in characters of other script, (b) Robustness, accuracy and simplicity of detection, (c) Speed of computation and (d) Insensibility of fonts.

The methods of detecting the features of Fig. 7 are described below. Here, the stroke lengths are standardized with respect to the character middle zone height.

To detect the presence of feature number 1 of Fig. 7(a), at first we detect the presence of a vertical stroke of length l_v. The length l_v is at least 75% of the height of the middle zone of the character. The right quarter part of the character middle zone is scanned vertically. If a scan contains continuous black pixels whose number is more than or equal to l_v then the vertical stroke is assumed to be present in that character. If the vertical stroke is present, we detect a left slant stroke of length l_s. The length of l_s is at least 40% of the height of the middle zone of the character. A scan in the upper left part of the middle zone of the character at an upwards inclination of 45° is made. If a scan contains continuous black pixels whose number is greater than or equal to l_s then feature number 1 is present in the character. The feature number 2 of Fig. 7(a) can be detected in a similar way.

To detect the feature number 3 of Fig. 7(a), the length of the arms of the stroke is assumed to be 40% of the height of the middle zone of the character and the angle between them is 45°. A two way scan is performed where starting point of the scan is from the left middle part of the middle zone of the character. If in a scan each arm contains continuous black pixels whose number is greater than or equal to the stroke length, then stroke number 3 of Fig. 7(a) is deemed present in that character. This feature is tested if a vertical line is present in a character.

The detection method of the feature number 4 is similar to that of feature number 3, only the scanning mode is different. Detection of other features of Fig. 7 can be done in a similar way.

Shape features from the upper-zone and lower-zone of characters are also considered here for different script line classification. For example, a modified character (*ekar*) of Devnagari script [12] always appears in the upper-zone with high occurrence frequency. The shape of this modified character is different from the upper-zone modified characters of Bangla script [13].

To identify Bangla and Devnagari lines, we test the principal stroke features in the characters of the words of that line. Two bins are maintained for the purpose. If a feature of Bangla (Devnagari) script is present in the character then the bin for Bangla (Devnagari) is incremented by one. Now, if the value of Bangla bin is more than that of Devnagari then we decide the line as Bangla. Otherwise, it is declared as Devnagari. No decision is taken for a script line if there is tie in the bin values.

For a character all the features are not necessarily tested. At first, the robust stroke features are checked and the subsequent feature testing depends on the presence of these strokes. For example, in a character we test the feature number 3 of Fig. 7 only if a vertical line exists in that character.

4.2. Separation of Chinese text lines

For the separation of Chinese text line from English and Arabic text we use the following features.

Vertical black run information: One of the most distinctive and inherent characteristics of the Chinese script is the existence of many characters with four or more vertical black runs. In English and Arabic alphabet, the number of such characters is very few. We compute character-wise maximum run number of English, Arabic and Chinese text lines. The results are shown in Fig. 8. Here, the run statistics is computed from 50,000, 40,000 and 35,000 characters of English, Chinese and Arabic text, respectively. From the figure it can be observed that the graph for Arabic and English have similar behavior after the run number 3. Also it can be observed that the graph obtained from Chinese text has a different nature. We note that in a Chinese text line there is about 57% characters having four or more black runs. In English there is only one character ‘g’ which has four vertical runs. We observe that the percentage of characters with one and two vertical black

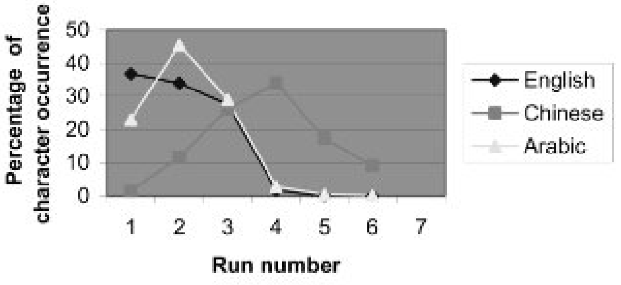

Fig. 8. Run numbers with their occurrence percentage of characters are shown for English, Chinese and Arabic text.

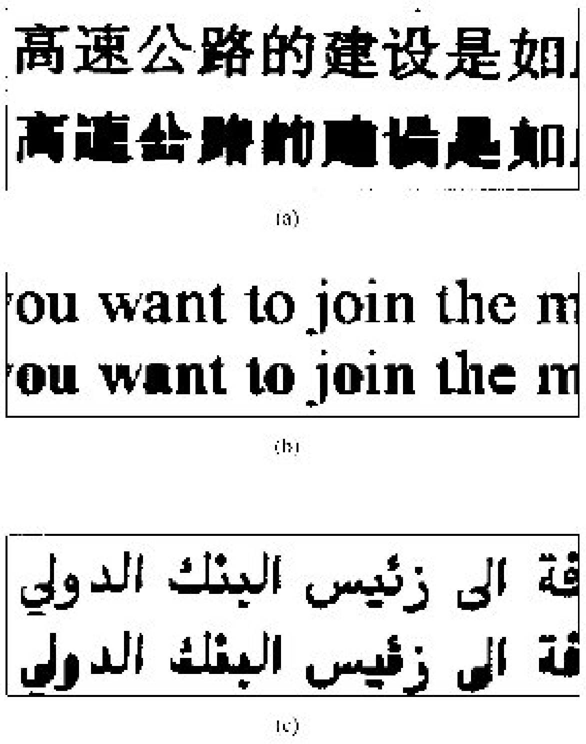

Fig. 9. Text line and modified RLSA result for (a) Chinese, (b) Roman and (c) Arabic script. (Top line original, bottom line modified RLSA).

runs in Chinese text line is very low (about 14%) whereas in English and Arabic text there are many characters (more that 65%) with one or two runs. In this regard Chinese script shows different behavior from the English and Arabic scripts. We use this simple but important criterion for the separation of Chinese text from English and Arabic ones. From a large data set we experimentally found that if 57% of the characters in a text line have four or more black runs then we can decide the line as Chinese.

Run length smoothing feature: The run number based feature can identify about 95.6% of the Chinese text lines from document mixed with English and Arabic. But from the experiment we noted that when (i) characters in a word touch one another, or (ii) a short Chinese text line containing some characters with run number less than 4, script line identification by run based feature may produce wrong results. The reasons of this wrong result are noted as follows: If the characters in a word touch one another due to digitization and if one of the characters in that word has run number 4 or more the word will be treated as a single component with run number 4 or more. Thus, an English or Arabic text lines may be classified as a Chinese text line. On the other hand, a short Chinese text line with most of the characters having run number less than 4 may be wrongly classified as English or Arabic text line. To tackle these types of misclassification, we use a run length smoothing algorithm (RLSA) [10] for script line identification.

Since run number based feature detection is very fast and can identify about 95.6% text lines correctly, we apply that feature at the beginning.

In actual implementation a slight modification of RLSA is used. The modified RLSA (MRLSA) is applied to a binary sequence in which white pixels are represented by 0's and black pixels by 1's. The algorithm transforms a binary sequence X into an output sequence Y according to the following rules:

1. 0's between two consecutive 1's in X are changed to 1's in Y if the number of 0's is less than or equal to a predefined limit L.
2. 1's in X are unchanged in Y.
3. 0's in the boundaries of X are unchanged in Y

For example, in row-wise MRLSA, with $L = 5$ the sequence X is mapped into Y as follows:

X: 0000110000000010010001001000000011100
Y: 0000110000000011111111111100000011100

The MRLSA is applied row-by-row as well as column-by-column to a text line, yielding two distinct bit-maps which are then combined by a logical AND operation. The original text lines and the result of applying MRLSA are shown in Fig. 9. For our experiment L is considered as 1/3 of the text line height.

From Fig. 9 it can be noted that most of the character portion get filled up when MRLSA is applied to a Chinese text line, which is not true for English and Arabic text lines. We compute the percentage of character coverage area of the MRLSA output text line and its original version. If for a text line, the coverage area is greater than 45% then we assume this feature is satisfied for Chinese script line and we identify that line as Chinese. Otherwise, it is English or Arabic.

4.3. Separation of English and Arabic text lines

For the separation between English and Arabic text, we use two features. One feature is based on the distributions of lowermost points of the components in the text line. The other is a feature obtained from the concept of water overflow from a reservoir. The features are as follows:

Distribution of lowermost points of the components: In English text, we note that the lowermost points of characters in a text line lie only on two horizontal lines. For the characters which do not have descenders, these points lie on the base-line. For other characters which have descenders (i.e. characters g, j, p, q, y), these points lie on a lower-line (Fig. 2(a)). But the lowermost points of most characters of an Arabic text line are more randomly distributed with respect to base-line and lower-line. Usually the lowermost points of an Arabic text line components do not lie on such lines. This property has been used for the separation of English and Arabic text line. Thus, for a text line we

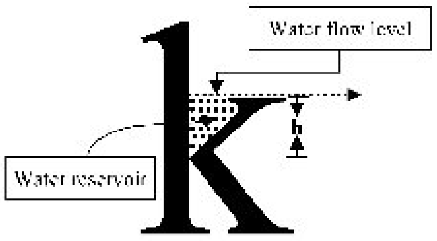

Fig. 10. Example of water reservoir. Here, reservoir area in the character 'k' is shown by the dots and reservoir height is shown by h.

compute two sets A and B of points corresponding to base-line and lower-line. If the lowermost point of a component does not lie on any one of these two lines, then we include this point in A or B according to nearest neighborhood rule. If the normal distance from this point to base-line is smaller than that to the lower-line, the point is assigned to A and so on. Since lowermost points of English characters lie either on base-line or lower-line, the standard deviation of the points of set A (σ_A) from base-line and standard deviation of the points of set B (σ_B) from lower-line row will be reasonably small. We compute the normalized standard deviations σ_A and σ_B to make this feature size independent. It is found from the experiment that if $\sigma_A + \sigma_B \geq 2.5$ then we assume this feature has satisfied the characteristics of Arabic text line and the text is considered as Arabic. This threshold value is obtained from experiments with a large volume of data.

Due to small components like dots on the English characters like 'i' and 'j', or due to salt and pepper noise, sometimes we may get high standard deviation value. This is because the lowermost points of these dots do not lie on base-line or lower-line. Thus, a English text line may be wrongly identified as Arabic. To get rid of this type of situation we select only those components whose bounding box widths are greater than half of the average bounding box width of all components in the line. Hence, small and irrelevant components like dots as well as random noise and punctuation marks are mostly filtered out.

Feature based on water overflow analogy: We noted that the feature based on distribution of lowermost points of the components can identify about 90.5% Arabic text line from English text. From the experiment we observed that when the characters in a word touch one another, script line identification by the above feature may produce wrong results. We use a new feature to tackle this problem.

The new feature may be explained by the analogy of water overflow from a reservoir. If we pour water from above the character, water will be stored in the concavity of the character, which may be imagined as a reservoir. The situation when water flows out of the reservoir, the water level height of the reservoir is noted. For illustration see Fig. 10. Water reservoir for English character 'K' is shown in this figure. Reservoir area in the character in shown by dots and the water overflow level is shown by arrow. Here 'h' shows the water level height in the reservoir when water overflows from the reservoir.

The characters in English where water can be stored are the upper-case characters H, J, K, L, M, N, U, V, W, X, Y and lower-case characters j, k, u, v, w, x, y (Fig. 11(a)). Water cannot be stored in the English characters like a, c, e etc. since their upper parts are not open. From the figure it can be noted that for most of the English characters water overflow levels do not coincide with the top of the bounding box of the character. From an analysis, based on the occurrence statistics of English characters computed by Suen [14] from one million English words, we find that for English text about 9% of the cases where water overflows from the reservoir, the water overflow levels do not coincide with the top row (upper-line) of the bounding box of the characters. On the other hand, in Arabic text there are many characters where water overflow level does not coincide with the top row (upper-line) of the bounding box of the characters (Fig. 11(b)).

For a text line we compute the total number of reservoirs as well as the number of reservoirs for which the overflow level coincides with either mean-line or upper-line (mean-line and upper-line are shown in Fig. 2). If the number of reservoirs for which the overflow level coinciding with either mean-line or upper-line is less than or equal to 9% of total reservoir number then we say that feature is true for English text and classify that line as English. Else, it is assumed to be Arabic.

Script identification algorithm: Based on different features discussed above, we present here the text line identification algorithm as follows:

Step 1
Take a text line from the document and calculate the longest horizontal run from the line. If the length of this run is greater than the threshold T_1 (the threshold is mentioned

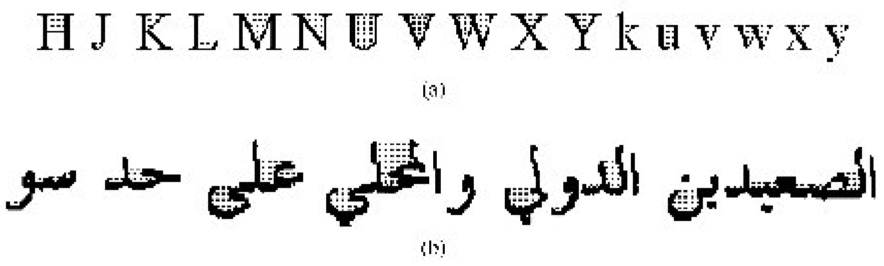
H J K L M N U V W X Y k u v w x y
(a)
الصعيدين الدولي والحلي على حد سو
(b)

Fig. 11. Examples of water reservoirs obtained in some (a) English and (b) Arabic characters. Here reservoirs are marked by dots.

Table 1
Identification results of the system

Script	Recognized as (%)				
	English	Chinese	Arabic	Devnagari	Bangla
English	97.32	0.61	1.16	0.46	0.44
Chinese	0.59	98.65	0.35	0.22	0.20
Arabic	1.38	0.28	97.53	0.61	0.20
Devnagari	0.61	0.46	0.63	96.05	2.23
Bangla	0.46	0.22	0.72	1.47	97.12

earlier) go to Step 1(a). Else, go to Step 2.
Step 1(a)
If the longest run occurs at the upper half of the text line, decide it as either Bangla or Devnagari (identification of Bangla or Devnagari is done according to the procedure given in Section 4.1) and go to Step 5. Else, consider it as Arabic and go to Step 5.
Step 2
If the number of characters in a line is greater than C, go to Step 2(a). Else, go to Step 2(b). (From the experiment $C = 10$ is chosen).
Step 2(a)
Compute the vertical run feature. If 57% of the characters in a text line have four or more black runs, then classify that line as Chinese and go to Step 5. Else, go to Step 2(b).
Step 2(b)
Compute run length smoothing features. If this feature satisfies the condition of Chinese (as mentioned in Section 4.2) script then classify that line as Chinese, and go to Step 5. Else, it is an English or Arabic line. Go to Step 3 for their identification.
Step 3
Compute the feature based on the distribution of lowermost points of the components. If this feature satisfies the characteristics of Arabic script we classify the line as Arabic and go to Step 5. Else, go to Step 4.
Step 4
Compute the feature based on water overflow analogy from reservoir. If this feature is true for English, we identify the line as English. Else, it is Arabic.
Step 5
If there are more text lines in the document image for identification, go to Step 1. Else, stop.

5. Results and discussion

For the experiment 25,500 text lines were considered from different documents like question papers, books, journals, magazines, computer printouts, translation books, multi-lingual operational and service manuals, etc. Identification results of different script lines are shown in Table 1. From the Table it can be noted that identification rates are 97.32, 98.65, 97.53, 96.05 and 97.12% for English, Chinese, Arabic, Devnagari and Bangla script. The overall accuracy of the system on these test data set is about 97.33%. It can be noted from the table that the Chinese script line identification has maximum accuracy which is about 98.65%. It can also be noted that maximum misclassification error occurs in the identification between Devnagari and Bangla text lines. This is due to their structural shape similarity.

From the experiment, we observed that most of the identification errors are obtained for short lines containing ten or less characters only. To reduce the error due to short lines, we use a heuristic. If a line contains only one word and if the position of that word is the extreme left (or right for Arabic, since the writing mode in Arabic is from right to left) then it is highly likely that it is the continuation of the previous line. We classify such short line to the class of its previous line.

Because of the use of simple features, which are easy to compute, our separation scheme is very fast. The scheme does not depend on the size of characters in the text line. Also, this approach is font, style and case insensitive. Experimented documents were of different font sizes ranging from 10 to 24 points with different styles. Italic style does not affect the separation of English, Arabic and Chinese text lines. The italic style may affect the Bangla and Devnagari text separation because in our scheme we use vertical line-like features. In this case we use the algorithm due to Chaudhuri and Garain [15] to detect italic lines. Now, for italic text instead of vertical line-like strokes, we search for slanted line-like strokes.

For our experiment we considered documents of different fonts. Vivek, Anand, Lino and Gitanjali fonts are considered for Bangla script documents, the fonts called Nataraj, Jogesh, Kishore, Bankim fonts are considered for Devnagari script documents, for English documents Times, Helvetica, Courier, Palatino fonts are considered, and SongTi, KaiTi, FangSong fonts are considered for the Chinese script documents. For Arabic text Al-Jawaher and liner true type fonts are mainly considered.

When characters of a word touch one another due to digitization or thresholding, script identification using the feature based on run length or distribution of lower-most points may provide wrong result. For this purpose we used the features obtained from the concept of water overflow from reservoir and modified run length smoothing. These features will not be affected if the characters in a word touch one another. Thus, lines where characters in the word touch one another also be segmented properly by the proposed script identification scheme.

This approach can be used for the separation of other similar scripts. For example, the *Punjabi*, *Marathi* and *Nepalese* scripts are similar to Devnagari script, Assamese Script is similar to Bengali script while Urdu script is similar to Arabic etc. Thus, this approach can be used for their identification.

6. Conclusion

In this paper we have proposed a method to identify different script lines from a document containing the five most popular scripts in the world. Script characteristics, character-shape based features which are easy to detect, statistical features and features obtained from the concept of water overflow from the reservoir have been used to separate them. This work is useful for multi-lingual OCR development.

References

[1] A.L. Spitz, in: R. Furuta (Ed.), Multilingual Document Recognition, Electronic Publishing, Document Manipulation, and Typography, Cambridge University Press, Cambridge, 1990, pp. 193–206.

[2] A.L. Spitz, Determination of the script and language content of document images, IEEE Trans. Pattern Anal. Mach. Intell. 19 (1997) 235–245.

[3] S. Wood, X. Yao, K. Krishnamurthi, L. Dang, Language identification for printed text independent of segmentation, Proc. Int. Conf. Image Process. (1995) 428–431.

[4] J. Ding, L. Lam, C.Y. Suen, Classification of oriental and European scripts by using characteristic features, Proc. Fourth Int. Conf. Doc. Anal. Recogn. (1997) 1023–1027.

[5] J. Hochberg, P. Kelly, T. Thomas, L. Kerns, Automatic script identification from document images using cluster-based templates, IEEE Trans. Pattern Anal. Mach. Intell. 19 (1997) 176–181.

[6] T.N. Tan, Rotation invariant texture features and their use in automatic script identification, IEEE Trans. Pattern Anal. Mach. Intell. 20 (1998) 751–756.

[7] T. Zhang, X. Ding, Cluster-based bilingual script-segmentation and language identification, Character Recogn. Intell. Inf. Process., Tsinghua University, China 6 (1999) 137–148.

[8] U. Pal, B.B. Chaudhuri, Script line separation from Indian multi-script documents, Proc. Fifth Int. Conf. Doc. Anal. Recogn. (1999) 406–409.

[9] S.A. Mahmoud, Arabic character recognition using Fourier descriptors and character contour encoding, Pattern Recogn. 27 (1994) 815–824.

[10] K.Y. Wang, R.G. Casey, F.M. Wahl, Document analysis system, IBM J. Res. Dev. 26 (1982) 647–656.

[11] B.B. Chaudhuri, U. Pal, Relational studies between phoneme and grapheme statistics in current Bangla, J. Acoust. Soc. India 23 (1995) 67–77.

[12] U. Pal, B.B. Chaudhuri, Printed Devnagari Script OCR System, Vivek, vol. 10, 1997, pp. 12–24.

[13] B.B. Chaudhuri, U. Pal, A complete printed Bangla OCR system, Pattern Recogn. 31 (1998) 531–549.

[14] C.Y. Suen, n-gram statistics for natural language understanding and text processing, IEEE Trans. Pattern Anal. Mach. Intell. 1 (1979) 163–172.

[15] B.B. Chaudhuri, U. Garain, Automatic detection of italics, bold and all capital words in document images, Proc. 14th ICPR (1998) 610–612.