2009 10th International Conference on Document Analysis and Recognition

Handwritten Text Line Identification In Indian Scripts

Bidyut B. Chaudhuri
CVPR Unit, Indian Statistical Institute
203, B. T. Road, Kolkata-700 108, INDIA
E-Mail: bbcisical@gmail.com

Sumedha Bera
CVPR Unit, Indian Statistical Institute
203, B. T. Road, Kolkata-700 108, INDIA
E-Mail: meet2sumedha@gmail.com

Abstract

Preprocessing in handwritten text OCR involves line, word and character segmentation. This paper deals with text line identification of handwritten Indian scripts, especially of Bangla, as well as English, Hindi, Malayalam, etc. Here, a new dual method based on interdependency between text-line and inter-line gap is proposed. The method draws curves simultaneously through the text and inter-line gap points found from strip-wise histogram peaks and inter-peak valleys. The curves start from left and move right while one type of points guides the curve of other type so that the curves do not intersect. Then these curves are allowed to iteratively evolve so that the text-line curves cross more character strokes while inter-line curves cross less character strokes and yet keep the curves as straight as possible. After several iterations, the curves stabilize and define the final text-lines and inter-line gaps. The approach works well on text of different scripts with various geometric layouts, including poetry.

1. Introduction

OCR of handwritten text is of great importance for electronic conversion of historical and genealogical information including letters, diaries, wills and other manuscripts. Other applications involve check and invoice processing, tabular form recognition, postal mail sorting etc. The problem is challenging because of human handwriting variability, uneven skew and orientation as well as noise and distortion such as smudges, smears, faded print, bleed-through and insect-perforation in the document. However, remarkable progress has been made on this problem in recent times [1].

The preprocessing task for any OCR engine is to identify the text region, isolate individual lines and words as well as segment the characters. Of the above tasks, sufficient attention has been focused on line identification [2]. The earliest and still popular line identification approach is based on projection profile, where the minima between two prominent peaks indicate the line separators. To get sharp peak and trough, Shapiro et al used Hough transform to get global skew and then used projection along that skew angle [3]. To handle local variation of line orientation, text region is vertically partitioned into strips and profile approach is employed on each strip. The results obtained on all strips are combined to form the full line separator. See, for example [4].

To get better effect on projection histogram, a smearing-based emphasis [5] is imparted on the document image. Black run length smearing along horizontal direction increases the histogram strength, so that small (say a one-word) text line also shows prominent peak.

The above methods are less robust to interline overlap and touching, especially when line separation is small. Several other approaches are proposed to tackle this problem. Some of them are based on fuzzy run length [6], linear programming [7], level set [8], Kalman filter [9], adaptive local connectivity map [10], min-cut/max-flow graph cut [11], local neighborhood of word [12] etc. In addition, vertical projection profile, enhanced by vertical smearing, accompanied with a scale space method has been used for word separation [13]. These approaches have shown some degree of success in detecting handwritten text lines, but further research is welcome.

In this paper we propose a new dual method of line identification that is based on interdependency between text line and inter-line gap. A text line will have inter-line white gaps both above and below it. Conversely, a gap will be defined by two text lines above and below. We can mark the text line by an imaginary curve called *intra-line curve* that cuts through the character strokes of that text line as many times as possible and yet remain maximally straight. Similarly, the gap can be marked by an imaginary curve called *inter-line curve* that separates two text lines above and below, with minimal intersection of character strokes. We also consider a predefined function to make intra-line curve pass through middle of the characters. Clearly, a intra-line curve should not cross a inter-line curve and vice versa. Our goal is to draw both types of curves, by using, among others, this non-crossing property. Here the idea is to start with a set of guiding points for generating the initial curves of both categories. These curves are allowed to change their local shape with the restriction that (a) they will be as (straight) linear as possible, (b) they can bend to a certain extent decided by the local horizontal projection spread, (c) they will cut through character strokes maximally/minimally depending on intra-line/inter-line identification and (d) one type of line will not touch or intersect the other type. This is like an elastic string that tries to remain straight but can bend to a certain extent at some points to satisfy property (c) as much as possible. After a few iterations with these constraints, the curves will evolve as semi-optimal piecewise linear curves for both text lines and inter-line gaps. Minute refinement of the curves is done as a post-processing step.

The dual detection has the advantage that one curve can partially guide the evolution of the other and prohibit it to go astray. This is useful in case of widely varying word gap, e.g. in a handwritten poetry or table. For complex documents, where there are annotations in the margin and at other places, the text line characters are better threaded if intra-text curves are also drawn along with inter-line curves.

The work has been primarily done for Bangla, a major Indian script having second highest popularity in the sub-continent. It has later been extended for several others Indian scripts and English. Test on so many types of scripts show robustness of the method.

In the rest of the paper, the basic characteristics of Indian scripts are described in section 2. The main approach of line identification is elaborated in section 3, while results and discussions are presented in Section 4.

2. Indian script characteristics

Most Indian scripts have 500 or more characters or symbols used in running text, though the number of basic vowels and consonants is not more than 50. The number is multiplied by three types of vowel modifiers that may be glued below the consonants, thus generating threefold consonant-vowel combinations. Further increase in number is possible where a consonant with another consonant creates a complex orthographic shape called *compound characters*. For some scripts like Bangla, Gujarathi or Devanagari, up to 200 compound characters are formed. Moreover, for Bangla script, there are two variants of some compound shapes called *transparent shape* and *opaque shape*. These compounds too can take vowel modifiers to generate threefold more shapes. Thus, orthographic shapes may run of the order of thousand. Only Tamil and Punjabi scripts are relatively simpler, where the numbers of character/symbol are about 150 and 70, respectively.

Most Indian script lines can be partitioned into three sub-zones. The upper and lower zones may consist of parts of the basic characters as well as vowel modifiers. These parts of two consecutive text lines normally do not overlap or touch in case of printed script, but for handwriting, people have the tendency to write them bigger, leading to overlap and touching.

Overall, these characteristics make handwritten Indian text recognition more challenging. Also, people do not normally have good training of handwriting in primary schools. The handwriting quality is further worsened by limited education and lack of regular writing practice among the community.

3. Proposed approach

There are several stages in the proposed approach, which are systematically described below.

3.1. Initial curve drawing points determination

In this stage, a subset of points for drawing initial inter and intra-line curves are found. Since strip-wise histogram is a good indicator of local text line and gap, we have utilized it in a flexible way to get these points. Here the vertical strip width is computed in a data-driven way. To do so, we start with a connected component labeling of the document text and delete very small components as noise. Next, the median of bounding-box heights of the remaining components, say H_m, is computed. This median gives an estimate of character height in the handwriting. The median of 'heights' is preferred because the 'widths' of components are more varying than height. A measure of $6H_m$ is taken as strip width for vertical partitioning, that captures from 2 to 6 characters in a strip. Next, the image is partitioned into vertical strips of this width, but with 50% overlap. The overlap helps in maintaining more continuity, and a small-sized text line (say of one word at the end of a paragraph) is not partitioned in two strip regions.

The task of next sub-stage is to increase the black pixel count within text region for generating prominent horizontal histograms on all strips. Unlike smearing, this is done as follows. The distances between two successive black-to-white transitions in horizontal runs are counted over the image. Let the median of such distances be T_m. This median gives an expected value of distance between two strokes. Next, the center of a rectangular window of length $2T_m$ and height $0.2T_m$ is placed at each black pixel and the number of distinct components within this window is counted. If the number is more than one, then the window is filled in black (Fig. 1a). On the other hand, if the number of component is only one, no action is taken (Fig. 1b). Such points belong to extension of strokes that can interfere into the inter-line gap (Fig. 2b). At the next stage, those parts on which no action has been taken are deleted (Fig 2c). Occasionally, such deletion creates some small components. They are also deleted (Fig 2d). These steps make deeper valley and higher peak in the histogram.

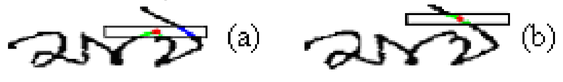

Figure 1. Block-filling window choice

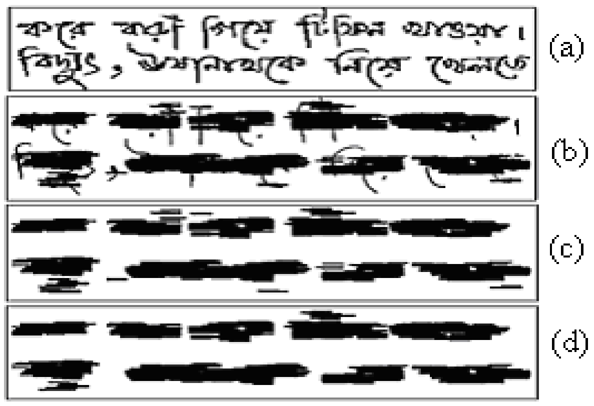

Figure 2. Black-run based enhancement for prominent histogramming

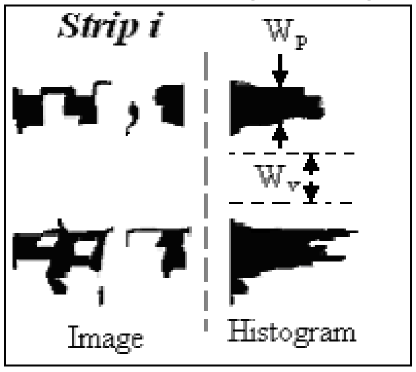

Figure 3. Range of Peak (W_p) and Valley (W_v)

After the histogram is drawn, instead of single maximum (or minimum) points, we choose a (vertical) range around them. The range is data driven, decided by the local standard deviation σ of histogram between two valleys. The range is taken as 2σ and its midpoint is considered as initial point for drawing the intra-line curve. Let us call it *sigma-mid* point. For inter-line gap, range computation by standard deviation is not reliable, since black pixel count is small or zero in the valley region. We take the range equal to 66% of the width of the valley. If a histogram follows Normal distribution, then 2σ around its mean value covers 66% area. With this analogy we choose 66% for valley range. The concept of ranges is illustrated in Fig 3.

3.2. Drawing of initial curves

Now, initial piecewise linear curve is generated using the following procedure:

The method starts drawing curves from left to right. To do so, the information about all possible text lines and gaps are ported in all strips. This is needed for some writings e.g. in poetry, some verses may be of small length, which may start and end midway on the horizontal extent of the document (Fig. 4).

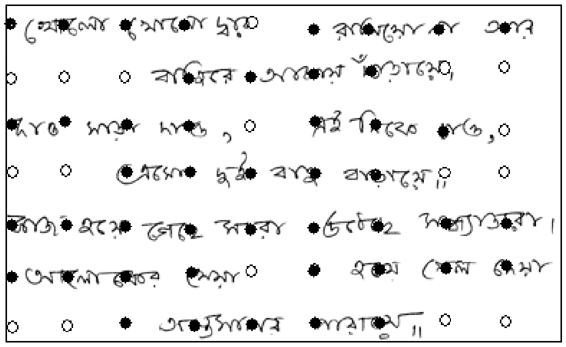

Figure 4. Sigma-mid or Initial control points

In such case no single strip contains the information of all possible lines. But between a pair of neighboring strips we get indication about beginning or ending of a text line. For example, if a line ends at the left strip, then it will increase inter-line gap in corresponding right strip. Conversely, a new line in the right strip may start at what was the inter-line gap of the left strip. By noting the histogram peak range (i.e., the 2σ range) in one strip falling in the inter-line range of the next strip or vice versa, and extending such information left and right over all pairwise neighboring strips, we can generate points for all possible text lines in the document. The procedure is stepwise described below:

1) Calculate average standard deviation σ_{av} from the peak ranges of all strips. Now, start with the first strip. If the gap between two neighboring sigma-mid points (shown in Fig. 4 by ●), say P_1 and P_2 is larger than 4σ, then we look at the next strip if there are sigma-mid points corresponding to P_1 and P_2. A sigma-mid point P_1' in the next strip corresponds to P_1 if their 2σ ranges of them have 50% overlap. If no such P_1' is found, then a dummy sigma-mid point (shown by ○ in Fig. 4) is placed at the same y-value as that of P_1. It is likely that the text line has ended in the previous strip, but we have to draw the curve corresponding to it till the right end. The same condition holds for P_2 also.

2) On the other hand, between P_1' and P_2' there may be one or more sigma-mid points in the next strip. Since in the previous strip the gap between P_1 and P_2 was larger than 4σ, these sigma-mid points may genuinely represent one or more text lines. So, we place dummy sigma-mid points at the same y-values in the previous strip corresponding to new sigma-mid points of this strip.

3) Therefore, the mechanism of sigma-mid point correspondence is done for all pair of neighboring strips while extension of the dummy sigma-mid points to right or left end are done by inspecting each individual strip. In this way, all points for drawing initial intra-line curve are generated (Fig. 4).

4) After completing the points for any two neighboring text lines, we calculate the points for inter-line gap. For each pair of sigma-mid points, two situations may arise: a) At least one sigma-mid point is real. Then, the mid point of the valley between corresponding peaks depicts the inter-line gap points. b) Both sigma-mid points are dummy. Then the y-value of its nearest inter-line point found by case (a) is used.

3.3. Evolution of the final curves

The final intra and inter-line curves are iteratively drawn using the following steps.

1) At first, the initial curves are drawn using the above inter-line and intra-line (sigma-mid) points, now called control points (CP) of the curve (Fig. 5.a).

2) Let for an initial intra-line curve there be n CPs. For i-th CP, where i ranges from 2 to n-1, we allow it to move up/down keeping (i-1)-th and (i+1)-th CP fixed, so that the curve crosses as many character strokes as possible without touching any inter-line curve. More weight is given if the curve is near the skeleton of component as in Fig. 2(d). The same type of operation is done on the subsequent inter-line curve so that it passes through minimum number of character strokes without touching any intra-line curve. When all curves are treated in this way, one epoch is complete. After several epochs, the curves stabilize in terms of stroke crossings (Fig. 5.b). On an average, 3 epochs are required to stabilize the curves.

3) Some inter-line curves may cross strokes extended in the inter-line region. Since the curves are piecewise linear, they cannot bypass such strokes by sharp bending. So, a post processing is done to avoid such extended strokes. In such a situation, we try to follow the border pixels of the crossing stroke clock-wise or anti clock-wise. We choose the direction in which we do not enter or cross intra-text line. If that is possible then at that region the inter-line curve is modified accordingly. Otherwise, the curve is not changed (Fig. 5.c second line right side and last line left side).

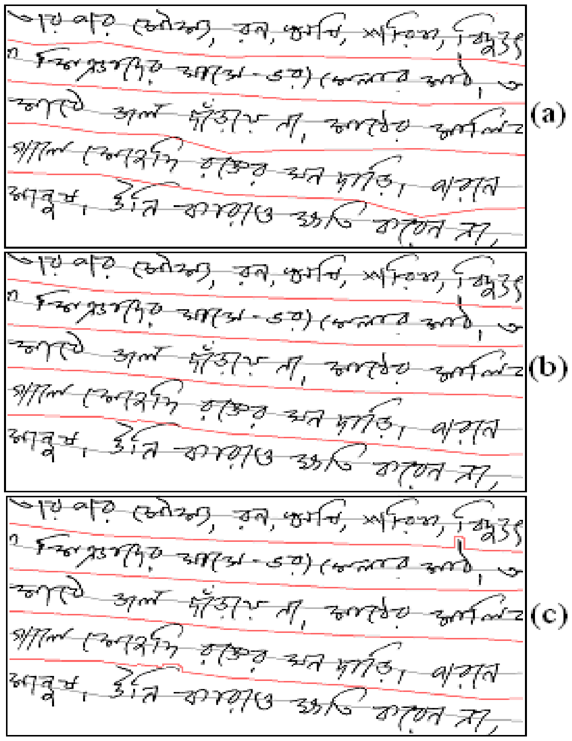

Figure 5. Evolution of curves for text lines and their refinement

4. Experimental Results and Discussions

We have collected 200 handwritten document pages from different people. These persons belong to both sexes and different age groups. The text documents contain six different scripts such as Bangla, Hindi, English, Oriya, Gujarathi and Malayalam. The text contains both prose and poetry and samples of all basic characters and modifiers for the respective script alphabet.

Typical examples of line detection results are shown in Fig. 6. Of them, Fig. 6(a) is for Malayalam script, Fig. 6(b) is for Gujarathi script, Fig. 6(c) is for Oriya script, Fig. 6(d) is for Bangla script and 6(e) is for English script. The overall accuracy on different scripts with all variations is shown in Table 1. Here mixing error means two lines are mixed in the identification process. Under-detection refers to the case where two lines have been detected as only one. Mixing has occurred more in poetry or table-like writings, while under-detection has occurred where there are annotations in between text lines or when the gap between lines is too small. We are working to improve the efficiency of our algorithm for such situations.

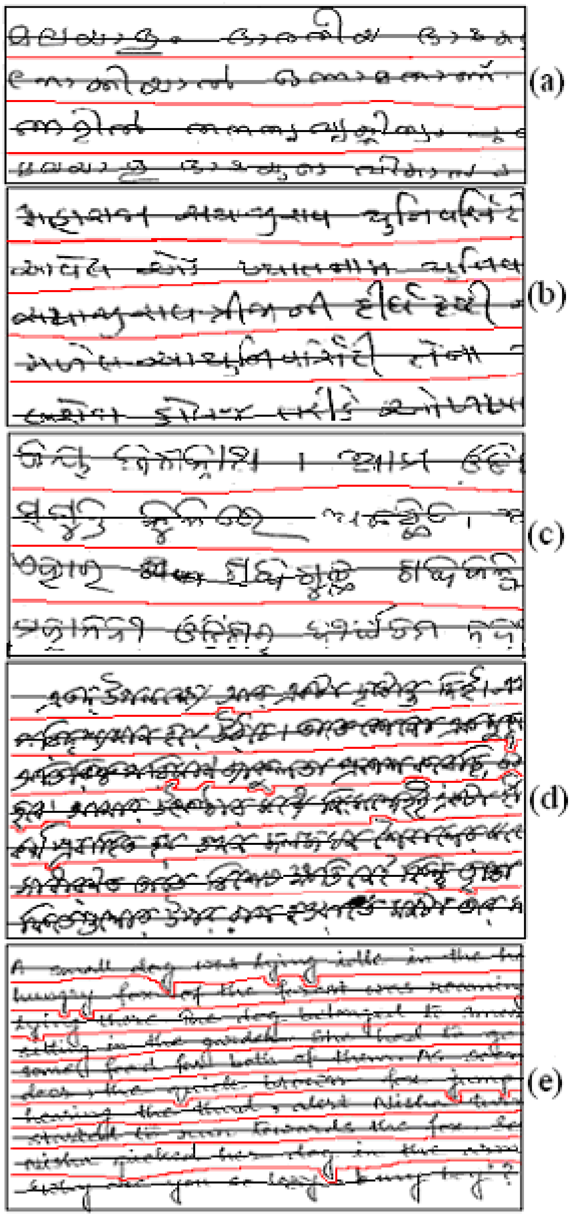

Figure 6. Line identification in various scripts

Table 1. Accuracy and error in %

Script	Accuracy	Mixed Error	Underdetection
Bangla	92.42	5.30	2.27
Eng	94.31	4.72	0.97
Malayalam	92.85	5.73	1.42
Oriya	93.28	3.89	2.83
Gujarathi	93.34	6.65	0.0
Hindi	93.17	5.21	1.62

The approach has been tested to work well on writing with multiple size, variable skew and spacing as well as varied line length and line position, like the verses in poems. Hence it is quite versatile and comparable with other methods.

References

[1] R. Plamondon, and S. Srihari, "On-line and off-line handwriting recognition: a comprehensive survey", IEEE Trans on PAMI, 22(1) , 2000, pp. 63-84.

[2] L.Likforman Sulem, A. Zahour, B. Taconet, "Text line segmentation of historical documents: a survey", IJDAR, Vol. 9, No. 2-4, 2007, pp. 123-138.

[3] V. Shapiro, G. Gluhchev, V. Sgurev, "Handwritten document image segmentation and analysis", Pattern Recognition, Letters archive, Vol. 14, Issue 1, 1993, pp. 71-78.

[4] A. Zahour, B. Taconet, P. Mercy, and S. Ramdane, "Arabic Hand-written Text-line Extraction", Proc. ICDAR, Seatle, USA, 2001, pp. 281-285.

[5] K. Wong, R. Casey and F. Wahl, "Document Analysis System", IBM J. Res. Dev. , 26(6), 1982, pp. 647-656.

[6] Shi Z., Govindaraju V., " Line separation for complex document images using Fuzzy Runlength", Proc. Int. Workshop on DIAL, 2004, pp. 306-312.

[7] B. Yanikoglu, and P. A. Sandon, "Segmentation of off-line cursive handwriting using Linear Programming", Pattern Recognition, Vol. 31, No. 12, 1998, pp. 1825-1833.

[8] Y. Li, Y. Zheng, D. Doremann and S. Jaeger, "A new algorithm for detecting text line in handwritten documents", Proc. IWFHR, 2006, pp. 35-40.

[9] A. Lemaitre and J. Camillerapp, "Text line extraction in handwritten document with Kalman Filter applied on low resolution image.", Proc. Int. Conf. on DIAL, 2006, pp. 38-45.

[10] Z. Shi, S. Setlur and V. Govindaraju, " Text extraction from gray scale historical document images using Adaptive Local Connectivity Map", ICDAR, Vol. 2, 2005, pp. 794-798.

[11] D. J. Kennard, W. A. Barret, " Separating Lines of text in Free-Form handwritten historical documents", Proc. Int. Conf. on DIAL, 2006, pp 12-23.

[12] S. Basu, C,Chaudhuri, M. Kundu, M. Nasipuri and D. K. Basu, " Text line extraction from multi-skewed handwritten documents", Pattern Recognition,2007, pp. 1825- 1839.

[13] R. Manmatha and J.L. Rothfeder, " A scale space approach for automatically segmenting words from historical handwritten documents.", IEEE Trans on PAMI, 27(8), 2005, pp. 1212-1225