An improved document skew angle estimation technique

U. Pal, B.B. Chaudhuri *

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

Received 2 November 1995; revised 5 March 1996

Abstract

When a document is fed to a scanner either mechanically or by a human operator for digitization, it suffers from some degrees of skew or tilt. Skew angle detection is an important component of any Optical Character Recognition (OCR) and document analysis system. In this letter we consider skew estimation of *Roman* script. The method considers the lowermost and uppermost pixels of some selected characters of the text which may be subject to Hough transform for skew angle detection. A fast approach is also proposed which works almost as accurately as Hough transform. Experimental results are presented and compared with results on several other skew detection methods.

Keywords: Optical character recognition; Skew angle; Connected component; Hough transform; Document processing

1. Introduction

The study of Optical Character Recognition (OCR) by machine has attracted more attention recently for its applications in many areas including bank cheque processing, postal ZIP codes and address recognition, office automation, document and text recognition and reading aid for the blind.

Whenever a document is fed to the OCR either mechanically or by a human operator a few degrees of skew (tilt) is unavoidable. Skew angle is the angle that the text lines of the recorded digital document make with the horizontal direction. As part of the development of a successful OCR (Pal and Chaudhuri, 1995) and document structure system we have devised a method, based on the lower pixels and upper pixels of some selected characters, for the estimation of the skew angle of the recorded digital document.

There exist three classes of skew estimation techniques that are based on (a) Projection profile, (b) Component nearest neighbor clustering, and (c) Hough transform.

The horizontal (vertical) projection profile (Hou, 1983) is a histogram of the number of black pixels along horizontal (vertical) scan lines. For a script with horizontal text lines the horizontal projection profile will have peaks at text line positions and troughs at positions in-between successive text lines. To determine the skew of a document, the projection profile is computed at a number of angles and for each angle a measure of difference of peak and trough height is made. The maximum difference corresponds to the best alignment with the text line direction which, in turn, determines the skew angle. Baird (1987) proposed a modification for quick convergence of this iterative approach. Akiyama and Hagita (1990) described an approach where the document is partitioned into vertical strips. The horizontal projection profiles are calculated for each strip and from the correlation

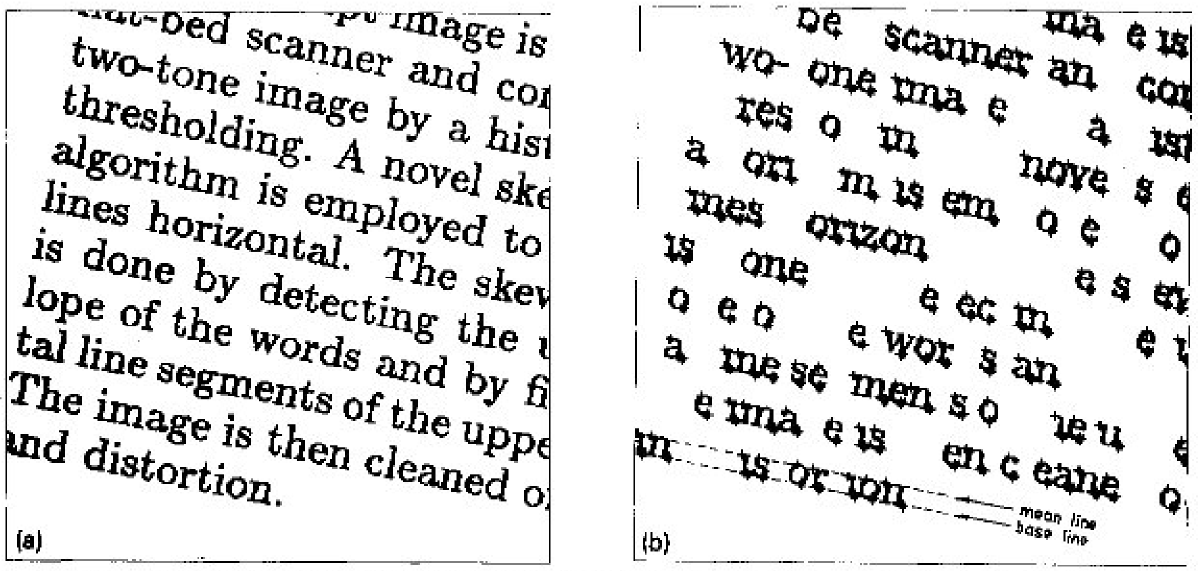

Fig. 1. (a) An example of English skewed text. (b) Components selected from (a) whose bounding box height is less than H. (Note that numerals, upper case characters, punctuation marks and characters which do not lie between the mean and base line as well as the dots of i and j are removed by Step 2 of our approach. Selected base line and mean line pixels are marked by * and +, respectively.)

of the profiles of the neighboring strips the skew angle is determined. This method is fast but less accurate. Pavlidis and Zhau (1992) proposed a method based on vertical projection profile of horizontal strips which works well if the skew angle is small. Another method using cross correlation between the lines at a fixed distance s shifted by a variable amount d in a vertical direction has been proposed by Yan (1993). The correlation for all pairs of lines in the image are accumulated. The shift for which the accumulated cross-correlation takes the maximum is then used for skew angle determination.

Hashizume et al. (1986) proposed nearest neighbor clustering to skew detection. He found all the connected components in the document and for each component computed the direction of its nearest neighbor. A histogram of the direction angle are computed, the peak of which indicates the document skew angle. O'Gorman (1993) generalized the approach in so called 'docstrum' analysis.

Hough transform has been used by Srihari and Govindaraju (1989) for skew detection. The basic method consists of mapping points in Cartesian space (x,y) to sinusoidal curves in (ρ,θ) space via the transformation

$$\rho = x\cos\theta + y\sin\theta.$$

Each time a sinusoidal curve intersects another at a particular value of ρ and θ, the likelihood increases that a line corresponding to that (ρ,θ) coordinate value is present in the original image. An accumulator array is used to count the number of intersection at various ρ and θ values. The skew is then determined by the θ values corresponding to the highest number of counts in the accumulator array.

Modifications of the Hough transform based approach have been proposed with mainly two aims. One aim is to discard the irrelevant data pixels so that the transform peaks are accurate. The other aim is to reduce the data so that less computation is necessary to execute Hough transform. For example, Hinds et al. (1990) made some modification based on the run length of black pixels to reduce the amount of data to be processed by Hough transform. Li et al. (1994) used the information that characters of a text line are aligned at the base line. Thus, they considered the bottom pixels of each character for doing Hough transform. However, characters like g, j, p, q and y have lowermost pixels below the base line. Also lowermost pixels of the dots of the characters like i and j, punctuations marks like comma, hyphen etc. also contribute to their data. These irrelevant pixels can create some error in the line approximation by Hough transform. One of the aims of our paper is to clean the data automatically so that only relevant pixels corresponding to the mean line and base line (see Fig. 1(b)) can be used for skew estimation. Another aim of this paper is to avoid Hough transform and devise a quick method of skew estimation using these relevant pixels.

2. Proposed skew estimation algorithm

Assume that the document is in portrait mode. Since *Roman* characters of a word are mostly separated, the characters in a document can be detected by the method of component labeling. Component labeling is a process by which pixels of a maximally connected component are marked by a single label.

Our method starts with component labeling. At the time of component labeling, the extreme (both in x and y) pixel co-ordinates are calculated for each labeled object and the bounding box is defined. For faster processing we have used these co-ordinates in future steps. The average height (H) of the bounding boxes is found and only the components with bounding box height less than H are considered. The upper case characters, numerals and characters like b, d, f, g, h, j, k, l, p, q, t, y do not participate in our algorithm as their heights are more than the average height. Also, we have debarred those components whose box height is very small so that the dots of the characters like i and j, punctuations marks like full stop, comma, hyphen, etc. are removed. For example, see Fig. 1(b). Now, from each selected component its lowermost pixel and uppermost pixel are chosen. If multiple lowermost pixels are there then the rightmost among them is selected. Similarly, for multiple uppermost pixels of a component the leftmost one is chosen. Selected lowermost and uppermost pixels are shown by '*' and '+', respectively, in Fig.1(b). Thus, we get the pixels that lie on base lines and mean lines of the text image. Let the set of mean line pixels (marked by '+' in Fig. 1(b)) be L_1 and the set of base line pixels (marked by '*' in Fig. 1(b)) be L_2.

Now we can use the Hough transform on all pixels of L_1 and L_2 to get a more accurate estimate of the skew angle. This approach is better than that of Li et al. (1994) because those pixels that can contribute to noise are removed while those pixels that can contribute to accurate estimation are nearly doubled (uppermost and lowermost pixels are both considered).

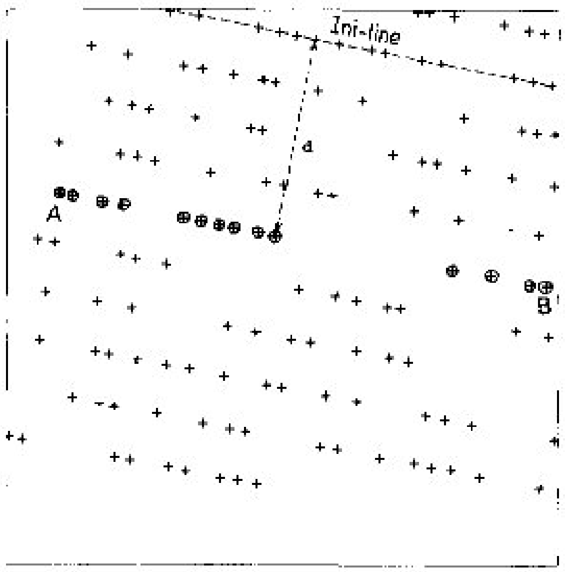

Fig. 2. Clustering of pixels of L_1 belonging to a single text line. (Encircled pixels are at a distance $d = \varepsilon$ from *ini-line*, where ε is a very small real quantity. A and B are two furthest pixels of the same cluster.)

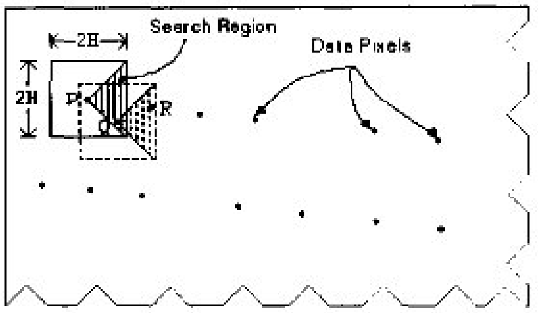

Fig. 3. The approach of choosing *ini-line*. (Pixel Q corresponds to a suffix character.)

To reduce the processing time further, we propose a quick approach as follows. We apply the approach separately on pixels of L_1 and L_2. Consider the pixels in L_1. Our idea is to cluster pixels corresponding to individual mean lines. For example, we would like to have the encircled pixels of Fig. 2 in one cluster and find their furthest pixels A and B. The slope of line $\overline{AB}$ gives an estimate of the skew for these encircled pixels. In this way pixels of each mean line are clustered and an estimate of the skew angle is obtained from them. An average of these skew angles is considered to be the estimate over L_1. Similarly, an estimate from L_2

can be found.

To cluster the pixels of each mean line we start with a line called *ini-line*. The pixels on L_1 are scanned rowwise and a $2H$ neighborhood of the first pixel encountered is considered (see Fig 3.). The neighborhood of $2H$ is considered because the spacing between two characters (hence the spacing between two pixels in L_1) is usually less than H. Now for a pixel P in L_1 we find its nearest neighbor pixel Q from L_1 that stays within a square of size $2H$ about P and the slope of the line $\overline{PQ}$ lies within $[-1, 1]$ (at present disregard the dashed square around Q). The slope restriction indicates that the expected skew angle is not greater than $\pm 45°$. The line joining P and Q could be used as *ini-line* but if a character has a subscript then it would lead to an error. In order to ensure that Q is not a pixel from a suffixed character we search around Q also, within a neighborhood of $2H$ (shown by dashed square) and find a neighbor pixel R in a similar way. If the angle between $\overline{PQ}$ and $\overline{QR}$ is nearly $180°$, then the line joining P and Q may be accepted as *ini-line*. In Fig. 3 we cannot accept $\overline{PQ}$ as *ini-line* since the angle between $\overline{PQ}$ and $\overline{QR}$ is not nearly $180°$. Now we replace Q and R by P and Q, respectively, and search for a nearest neighbor pixel of the current Q to test if the condition of *ini-line* is satisfied. This process may be repeated if necessary, for finding *ini-line*.

Finding pixels of different mean lines in different clusters is similar to filling parametric space bins as in Hough transform. A bin corresponds to pixels having nearly constant normal distance to *ini-line*. Hence, for an unprocessed pixel X in L_1 its normal distance to *ini-line* is computed and if this distance is within $\pm\varepsilon$ (where ε is a very small quantity) of the distance of any existing bin pixel then X is placed in that bin for clustering. Otherwise, a new bin is created and X is placed in the new bin. The whole procedure can be executed in single scan over L_1. In the end, K bins and hence K clusters are usually produced if there are K text lines in the document.

Now consider pixels in L_2 and apply the same procedure. The final estimation of skew angle is the average of these two angles obtained from L_1 and L_2. In the next section it is shown that this fast approach is nearly as accurate as doing Hough transform on pixels of L_1 and L_2.

In brief, our fast skew detection algorithm has the following steps.

Algorithm SKEW

Step 1. Find connected components in the binary document image and compute average bounding box height H.

Step 2. Choose those connected components whose height is less than H and remove very small connected components so that the dots of the characters i and j, punctuations marks like full stop, comma, hyphen, etc. are deleted. Let the chosen set be S.

Step 3. For each component in S find an uppermost pixel and lowermost pixel to obtain L_1 and L_2.

Step 4. From the pixels of L_1 find *ini-line*.

Step 5. For each pixel X in L_1 find the normal distance of X to the *ini-line*.

Step 6. Cluster the pixels of L_1 corresponding to individual text lines. If the distance of one pixel of a cluster from *ini-line* is d, then the distance of any other pixel in the cluster is less than $d \pm \varepsilon$, where ε is a very small quantity.

Step 7. From each cluster of pixels find the slope of the line joining two furthest pixels. The slope, in turn, gives a local estimate of the skew angle.

Step 8. Find the average of such angles.

Step 9. Repeat Steps 4 to 8 for pixels of L_2.

Step 10. Find the average of these two angles obtained from L_1 and L_2 as the final estimate of the skew angle.

3. Results and discussion

For the experiment we considered more than one hundred document images from different books, magazines and journals. They were digitized by a flatbed scanner at a resolution of 160 dpi. The documents were tilted by a prespecified angle ranging between 0 and $\pm 45°$. This angle is considered as the true skew angle. The documents were subject to both the Hough transformed based approach as well as the quick approach on pixels of L_1 and L_2. Typical results are shown in Table 1 where other estimated techniques (O'Gorman, 1993; Li et al., 1994; Yan, 1993) are also considered. Please note the abbreviation of the methods A, B, C, D, E, F in the table captions, which we shall use later on.

It may be noted from Table 1 that Hough transform on pixels selected by the proposed approach (i.e. pixels of L_1 and L_2) gives a better result than the Li et

Table 1
Mean and Standard Deviation (SD) of estimated skew angles obtained by different methods
(For each true skew angle 20 images have been tested)

True skew angle (in deg) (manual)		Mean and SD of estimated skew angles using method					
		A	B	C	D	E	F
30	Mean	29.48	29.40	30.16	29.85	29.11	29.88
	SD	0.559	0.965	0.313	0.379	0.665	0.537
20	Mean	19.91	19.25	19.66	19.52	19.16	19.53
	SD	0.777	1.013	0.366	0.349	0.422	0.918
10	Mean	10.21	10.67	10.05	9.96	10.71	10.46
	SD	0.733	0.840	0.216	0.233	0.997	0.717
5	Mean	4.71	5.24	5.18	5.03	5.74	5.01
	SD	0.155	0.687	0.436	0.269	0.295	0.489
3	Mean	3.72	2.80	2.88	3.14	3.82	3.69
	SD	0.079	0.534	0.228	0.331	0.627	0.384

A: Hough transform over total image
B: Hough transform on pixels selected by the Li et al. (1994) method
C: Hough transform on pixels selected by the proposed method, i.e., pixels of L_1 and L_2
D: Our proposed quick method
E: Docstrum method (O'Gorman, 1993)
F: Yan's (1993) method

al. (1994) method as expected because of deletion of irrelevant pixels. The docstrum method (O'Gorman, 1993) is less accurate for almost all skew angles. We noted that the accuracy of the docstrum method could be improved if the characters having ascender and descender are deleted, as done in our approach. Yan's (1993) method is quite accurate but it is computationally time consuming. Also, for its implementation some a priori knowledge about the spacing is needed to compute the correlation.

To test the time efficiency of our quick method (method D) with respect to Hough transform based methods C and A, we computed the time of execution in these methods. The angular resolution used in Hough transform was 1°. The average execution times for a document of 512 × 512 pixels on a SUN 3/60 (with Microprocessor MC68020, and SUN O.S. version 3.0) machine are 515.15, 30.07 and 10.35 seconds for methods A, C and D, respectively. Also the average execution times of methods B, E and F are 35.41, 11.95 and 40.21 seconds, respectively.

To test the skew angle estimation efficiency of the methods we have done a statistical testing of hypothesis using F-distribution obtained as follows. Let $X_1, X_2, \ldots, X_n$ be n independent observations from a Gaussian distribution with mean ϕ and standard deviation σ, i.e.,

$$\sum_{i=1}^{n} \frac{(X_i - \phi)^2}{\sigma^2} \sim \chi^2_n.$$

For independent true skew angles Y_j ($j = 1, 2, \ldots, n_1$) let X_{ji} and Z_{ji} ($i = 1, 2, \ldots, n_2$) be the independent estimated angles for the ith observation by method D and method C, respectively. We assume X_{ji} and Z_{ji} follow Gaussian distributions with standard deviations σ_1 and σ_2, respectively.

Then

$$T_1 \triangleq \frac{1}{\sigma_1^2} \sum_{j=1}^{n_1} \sum_{i=1}^{n_2} (X_{ji} - Y_j)^2 \sim \chi^2_{n_1 n_2}$$

and

$$T_2 \triangleq \frac{1}{\sigma_2^2} \sum_{j=1}^{n_1} \sum_{i=1}^{n_2} (Z_{ji} - Y_j)^2 \sim \chi^2_{n_1 n_2}.$$

Our null hypothesis is H_0: $\sigma_1 = \sigma_2$ under which

$$T \triangleq \frac{T_1}{T_2} \sim F_{n_1 n_2, n_1 n_2}.$$

We have seen from our estimated skew angles that the value of T is 1.17, where $n_1 = 5$ and $n_2 = 20$. The 5% cut off point of the F-distribution (with degrees of freedom 100 and 100) is 1.41 (obtained through interpolation). Hence, we can say that the methods C and D are statistically equally efficient.

For a complex document containing half-tone images, graphics and drawings the method would work well if the text region is substantially more than the non-text region and if H is chosen in a modified way. The areas of the bounding box of individual components are grouped in small bins to give a bar histogram representation. Since according to our assumption the text region is substantially larger, the mode of this histogram will represent the bounding box area of the text characters. The bar corresponding to the mode gives the value of H in an interval corresponding to the bar width. If the components corresponding to bounding box area within this interval are retained then almost all non-text components are removed. The proposed approach can be used on the remaining components. However, it is better to use method C rather than method D because the Hough transform is more robust to noise while the quick method depends on the right choice of *ini-line* and clustering of pixels of each text line which may not be attained always for a complex document.

Acknowledgements

Interest in the work by Dr. S.K. Parui is also acknowledged with thanks. One of the authors (Mr. U. Pal) wishes to thank CSIR, India for providing him a fellowship in pursuing his work.

References

Akiyama, T. and N. Hagita (1990). Automatic entry system for printed documents. *Pattern Recognition* 23, 1141–1154.

Baird, H.S. (1987). The skew angle of printed documents. *Proc. Society of Photographic Sci. Eng.* (Rochester, NY) 40, 21–24.

Hashizume, A., P.-S. Yeh and A. Rosenfeld (1986). A method of detecting the orientation of aligned components. *Pattern Recognition Lett.* 4, 125–132.

Hinds, S.C., J.L. Fisher and D.P. D'Amato (1990). A document skew detection method using run-length encoding and the Hough transform. *Proc. 10th Internat. Conf. on Pattern Recognition*, Hedgue, Vol. 1, 464–468.

Hou, H.S. (1983). *Digital Document Processing*. Wiley, New York.

Li, D.S., G.R. Thoma and H. Wechsler (1994). Automatic page orientation and skew angle detection for binary document images. *Pattern Recognition* 27, 1325–1344.

O'Gorman, L. (1993). The document spectrum for page layout analysis. *IEEE Trans. Pattern Anal. Mach. Intell.* 15, 1162–1173.

Pal, U. and B.B. Chaudhuri (1995). Computer recognition of Printed Bangla Script. *Internat. J. System Sci.* 26, 2107–2123.

Pavlidis, T. and J. Zhau (1992). Page segmentation and classification. *Computer Vision, Graphics, and Image Processing* 54, 484–496.

Srihari, S.N. and Govindaraju (1989). Analysis of textual images using the Hough transform. *Machine Vision Appl.* 2, 141–153.

Yan, H. (1993). Skew correction of document images using interline cross-correlation. *Computer Vision, Graphics, and Image Processing* 55, 538–543.