2011 International Conference on Document Analysis and Recognition

Determining Document Skew Using Inter-Line Spaces

Boris Epshtein
Google Inc.[1]
1600 Amphitheatre Parkway, Mountain View, CA
borisep@google.com

We present a novel method of determining a global text page orientation. The method is based on Hough transform, but, unlike the existing methods, it does not use the letters themselves and relies on establishing the orientation of inter-line regions. The method is robust and also works even when a single line of text is present. Experimental evaluation is shown, comparing the method to other methods proposed in the literature.

Keywords-document analysis; skew; orientation; Hough transform; inter-line regions

I. INTRODUCTION

The Optical Character Recognition (OCR) gains new fields of application nowadays, due to the introduction of OCR apps for mobile phones, such as Google Goggles, WordLens and iBing Vision. In these apps, the pages of text are typically photographed by a mobile device and then read automatically. One notable problem in such OCR systems is the need of full 360 degrees orientation detection, in order to bring the text lines to horizontal orientation. If, in the older OCR algorithms, developed with the goal of reading scanned documents, preprocessing algorithms were sometimes geared towards removing a small amount of skew, today we need systems capable of detecting any rotation. Another difference from the traditional OCR is the requirement to work reliably on very small amounts of text, sometimes as small as one word. Additionally, not all texts are organized in familiar structures of columns and paragraphs, which makes the use of traditional text layout segmentation algorithms problematic. We present a novel algorithm for a text skew detection based on the well-studied Hough transform technique, but with a surprising twist: instead of estimating the direction of the text lines, we estimate the direction of the white spaces between lines. In the current work we assume that all the text lines on the page have approximately the same direction (i.e. the page is roughly parallel to the image plane). We also do not try to disambiguate between normal and upside-down positions of the text, so we will count the rotation angles differing by π radians as equal. As shown in the Experiments section, the algorithm is robust, accurate, works on small amounts of text such as one line and outperforms traditional Hough transform-based algorithm and projection-based algorithm.

II. PREVIOUS WORK

There exist several families of skew detection algorithms. [1] We are summarizing them, as well as their pluses and limitations, in the current section. For a detailed survey of the skew detection algorithm, interested readers are referred to [8].

1. Hough transform

Methods of this family include [1, 2, 3, 4, 10, 11, 12]. In the simplest form, a Hough transform-based method starts with a binary image, and applies the Hough transform to every black pixel in the image. The mapping form a Cartesian *(x, y)* pixel coordinates to the parametric polar space *(ρ, θ)* is considered. For every selected point with the coordinates *(x, y)* one is added to every cell along the sinusoidal curve in the *(ρ, θ)* space defined by the equation

$$\rho = x\sin\theta + y\cos\theta \qquad (1)$$

We call the *(ρ, θ)* space the Hough accumulator. After all the selected points have been enumerated, maxima in the Hough space are determined and corresponding lines (parallel to the text lines) found. Then the skew angle can be found by summing along the ρ dimension and finding the angle corresponding to the maximal sum. A number of improvements to this basic strategy have been developed by researchers over the years. For example, method [3] applies a hierarchical Hough transform to the centers of connected components, representing letters. Other methods [12] create a segmentation of text page into rectangular blocks, corresponding to paragraph, and apply Hough transform to the edges of the paragraphs. These methods actually benefit from long text lines and work less reliable on small amounts of text. Also, narrow columns of text can be problematic, since lines of all directions can be composed of letters in that case. Additionally, methods relying on paragraph structure [7] seem to work less robust on texts lacking such layout. Our method also belongs to this group, but substantially improves on the robustness of the methods.

2. Projection-based algorithms.

The algorithms of this family [5, 6, 7, 14, 16] typically examine several candidate directions of text lines. Pixel

[1] The work was done while the author was employed by Microsoft Corp.

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.15

values (or, alternatively, the squared gradients) are summed along the directions and stored in bins. If the direction coincides with the true direction of the text lines, the projection will look like a series of peaks and deep valleys, corresponding to the text lines and spaces between lines. If the direction does not exactly coincide with the skew angle, the peaks will be more shallow. because of the overlaps of the projections of the text lines. This property (the sharpness of peaks and depth of valleys) can be expressed computationally by a number of measures, like the entropy of the histogram. Then the direction with the best value of the measure is selected. The limitation of this algorithm is its running time: if the text lines are long, even slight difference between true skew angle and candidate angle will lead to the "smearing" of the peaks and valleys on the projection. Because of that, many candidate angles should be examined. Several improvements with smaller running time exist, but they typically work for a limited range of skew angles. Another limitation of this approach is the poor handling of multi-column text, which can lead to a catastrophic error of $\pi/2$ radians in direction estimation.

3. Other algorithms

Several other types of algorithms have been suggested in the literature, such as clustering the directions between feature points [13], cross-correlation between text lines [15], Fourier analysis, etc.

III. The Algorithm

In this section, we first give an informal description of the algorithm, followed by the formulation of the algorithm.

Imagine all possible straight lines, running through a page of text. We discard lines that are not sufficiently close to the text blocks. From the rest of the lines, we give bigger weight to those that run only in the white space and lesser (sometimes even negative) weight to the lines that cross the letters. As a result, lines parallel to the text and passing through the white inter-line space will have big weight, and can be easily detected by a Hough transform. Since we are interested in the orientation of such lines and not in their distance to the center of the page, we can sum the values of Hough cells along the ρ axis and then find the prevalent orientation θ.

We follow with the formal description of the algorithm.

The proposed skew detection algorithm takes a grayscale image of a text page as an input and proceeds along the following steps:

1. Binarize the input image. In our system, we use a local binarization algorithm - a version of Sauvola's algorithm [9]. This step creates a binary image B, with pixel values equal 1 for text and 0 for background. Actually, our binarization algorithm handles both black texts on white background and black texts on white background simultaneously on the same page, but the detailed explanation of its details is out of scope of this paper.

2. Extract the connected components from the image B. Each connected component is considered a letter candidate (or several letters glued together, or a text string, in connected scripts).

3. Filter the connected components according to a set of geometric rules. For example, too small connected components (in our system, less than 5x5 pixels) are deleted. The rest of the rules are dealing with filtering out the components with an aspect ratio out of acceptable bounds, too big components or components with the ratio between the perimeter and square root of area out of acceptable bounds. The filtered out components are then deleted from image B, i.e. replaced with the background color.

4. Create a mask M from the image B using the morphological operation of dilation. The radius of dilation is set to be one third of the median size (expressed by the bigger side of the bounding box) of the participating connected components. The value of a pixel in M is 1 if it is within a radius r from a foreground pixel (of value 1) in image B and 0 otherwise.

5. Consider the value of each pixel in B.

- If the pixel has value 0 and the corresponding pixel in M has value 1, add 1 to every corresponding cell of a Hough accumulator (eq. 1).
- If the pixel has value 1, subtract a value δ from each corresponding cell of a Hough accumulator. In our system, $\delta = 8$. All the parameter values in the algorithm were determined by optimizing the mean error on a small training set (20 images of texts), different from the test set used in experiments.
- Otherwise, do nothing

Do this for every pixel in B.

6. Compute a threshold T from the values of Hough cells. In our system, we set T to be the 99th percentile of the distribution of the values in Hough cells.

7. Go over all the Hough cells and sum up all the values along the ρ dimension whose values exceed T, thus creating a projection onto the θ dimension.

8. Find the maximum of the projection. Output the corresponding value θ as the result of the algorithm.

Consider again all possible straight lines that can pass through a page. If the line mostly passes through the margins, or in any other fashion is far from text, it will not contribute to the Hough accumulator, since the values of the corresponding pixels in both M and B will be 0. If the line is not completely lying in the white space between lines, it will necessarily cross the letters, and therefore the corresponding Hough cells will not accumulate much votes, since a lot of δ's will be subtracted from them. Therefore, the lines coinciding to the white space between lines will get most of the votes in the Hough accumulator, and they all will share the direction θ, which we will be able to identify by thresholding and summing along the ρ dimension .

Let us additionally note that the Steps 2 and 3 are optional, designed to give the system a better robustness for noise, and, in some circumstances and imaging conditions,

may be omitted, depending on the area of application of the OCR system.

IV. Experimental Results

In order to evaluate the performance of the algorithm, we performed several experimental tests on both real and synthetic data.

A. Experiments on real images database.

For this experiment, we have collected a database of 500 images of documents - magazine pages, newspapers, books, etc., photographed using an iPhone 3GS camera. On each image the bounding boxes of all the words were marked, and the ground truth skew direction was computed from them. To do that, we took for each text line the direction between two points: the middle of the left border of the leftmost bounding box and the middle of the right border of the rightmost bounding box. After that, all the text line directions were averaged. Note that this ground truth estimation technique is imprecise, and does not work well for big fonts and short text lines.

We computed the skew direction as explained in the Section III. Additionally, we ran a previously reported in the literature projection-based algorithm [16] and the traditional Hough-based algorithm that takes into account the letters instead of white inter-line intervals. The results, showing average difference between ground truth and computed skew angle for all three algorithms, are shown in Table 1. Examples of the database images and the detected orientations are shown in Figure 1.

Table 1. Mean and variance of errors (in radians) on the database of 500 real images for three algorithms.

Algorithm	Our method	Projections	Traditional Hough
Mean error	0.007499	0.030221	0.135413
Error variance	0.004883	0.032231	0.182685

The high error mean and variance in the case of benchmark algorithms are due to high number of catastrophic errors, i.e. cases where the detected text orientation differs from the ground truth by more than $\pi/10$. Table 2 shows the numbers of catastrophic errors for all three algorithms.

Table 2. The number of catastrophic errors on the database of 500 real images for three algorithms.

Algorithm	Our method	Projections	Traditional Hough
Number of catastrophic errors	1	14	43

The examples of catastrophic errors are shown in Figure 2. The full distribution of errors for the proposed algorithm is shown on Figure 4. When the only image producing the catastrophic error is removed from the dataset, the mean and variance of error for our proposed algorithm becomes even more impressive (Table 3).

Table 3. The minimal, maximal, mean error and error variance for the proposed algorithm on the dataset with one catastrophic error removed.

Mean error	0.004375
Error variance	0.000013
Minimal error	0.000000
Maximal error	0.029234

As can be seen from the results, the algorithm, although run on a different, more difficult database, achieves state of the art performance for skew detection.

B. Results on synthetic images.

For this experiment, we produced 8 images with horizontal text of different sizes and layouts (Figure 3). We added blur and additional Gaussian noise to all images, in order to make task more difficult. Then we ran our orientation computation on images rotated by different angles, from 0 to π radians, with a step of 0.05 radians. The results, showing average difference between ground truth and computed skew angle for the proposed algorithm are shown in Table 4.

Table 4. The minimal, maximal, mean error and error variance for the proposed algorithm on the dataset of 8 images of horizontal texts.

Image	Mean error	Error Variance	Maximal error	Minimal error
1	0.002192	0.000002	0.004585	0.000037
2	0.006460	0.000006	0.010779	0.002274
3	0.002186	0.000002	0.004412	0.000037
4	0.003734	0.000007	0.010239	0.000086
5	0.016156	0.000006	0.020440	0.011935
6	0.008749	0.000128	0.066433	0.000135
7	0.027616	0.000006	0.032141	0.023193
8	0.002305	0.000002	0.005346	0.000037

V. Conclusion and future work

We have presented an algorithm for skew detection in arbitrary rotated documents. The algorithm is shown to be robust, handling any orientation and producing less catastrophic errors than classical Hough-based algorithm and projection-based algorithm. Two possible extensions are better handling of projective transformations and handling pages region by region, which is needed for proper processing of curved pages.

References

[1] S. Srihari, V. Govindaraju, "Analysis of textual images using the Hough Transform", Machine Vision and Applications, vol. 2, 1989, pp. 141-153

[2] D. Le, G. Thoma, H. Wechsler "Automatic page orientation and skew angle detection for binary document images", Pattern Recognition 27, 1994 pp. 1325-1344

[3] B. Yu, A. Jain, "A robust and fast skew detection algorithm for generic documents", Pattern Recognition 29(10), 1996, pp. 1599-1629

[4] U. Pal, B. Chaudhuri, "An improved document skew angle estimation technique", Pattern Recognition Letters, Vol. 17, 1996, pp. 899-904

[5] H. Hou, "Digital document processing", Wisely, New York, 1983

[6] T. Akiyama, N. Hagita "Automated entry system for printed documents", Proceedings of Conference Society of of Photographic Scientists and Engineers, Rochester, New York 1987, pp. 14-21

[7] T. Pavlidis, J. Zhou, "Page segmentation by white streams", ICDAR, 1991, pp. 945-953

[8] J. Hull, "Document Image Skew Detetion: Survey and Annotated Bibliography", Document Analysis Systems II, World Scientific, Singapore, Jonathan J. Hull and Suzanne L. Taylor (editors) 1998, pp. 40-64.

[9] J. Sauvola, M. Pietikainen, "Adaptive document image binarization," Pattern Recognition 33(2),

[10] N. Nandini, K. Srikanta Murthy, G. Hemantha Kumar, "Estimation of Skew Angle in Binary Document Images Using Hough Transform", World Academy of Science, Engineering and Technology, 42, pp. 44-50

[11] A. Amin, S. Fischer, "A Document Skew Detection Method Using the Hough Transform", Pattern Analysis And Applications, 2000, vol. 3, pp. 249-253

[12] Y. Ham, H. Chung, I. Kim, R. Park "Automated Analysis of mixed documents consisting of printed korean alphanumeric texts and graphic images", Optical Engineering, 33, 6, pp. 1845-1853

[13] A. Hashizume, P.S. Yeh, A. Rozenfeld, "A method of detecting the orientation of aligned components" Pattern Recognition Letters, 1996, pp. 125-132

[14] H. Baird, "The sked angle of printed documents" Proc. of SPSE 40th Symposium on Hybrid Imaging Systems, 1987, pp. 739-743

[15] H. Yan, "Skew correction of document images using interline cross-correlation", Computer Vision, Graphics and Image Processing, 55, 1193, pp. 538-543

[16] W. Postl, "Detection of linear oblique structures and skew scan in digitized documents", Proceedings of the 8th International Conference on Pattern Recognition, 1986, pp. 687-689

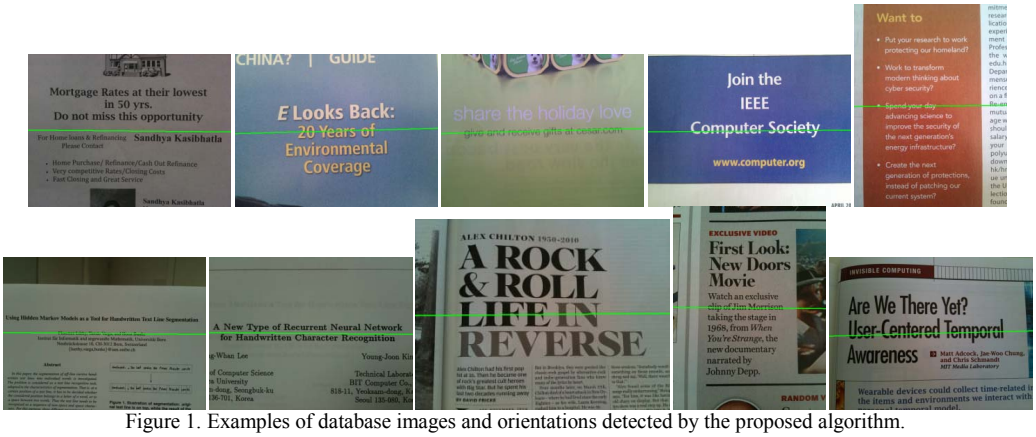

Figure 1. Examples of database images and orientations detected by the proposed algorithm.

Figure 2. Examples of the catastrophic errors produced by the algorithms. (**a**) The only catastrophic error produced by the proposed algorithm. (**b**) Examples of catastrophic errors produced by the Projections method. (**c**) Examples of the catastrophic errors produced by the Traditional Hough method. Blue lines: ground truth orientation, green line: orientation computed by the proposed method, red line: orientation computed by the Projection method, yellow line: orientation computed by the Traditional Hough method.

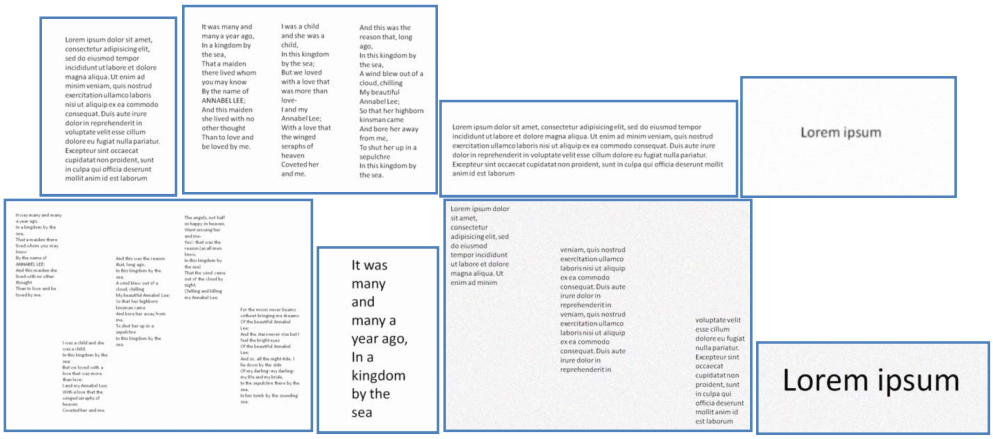

Figure 3. Images with horizontal text used in the second experiment.

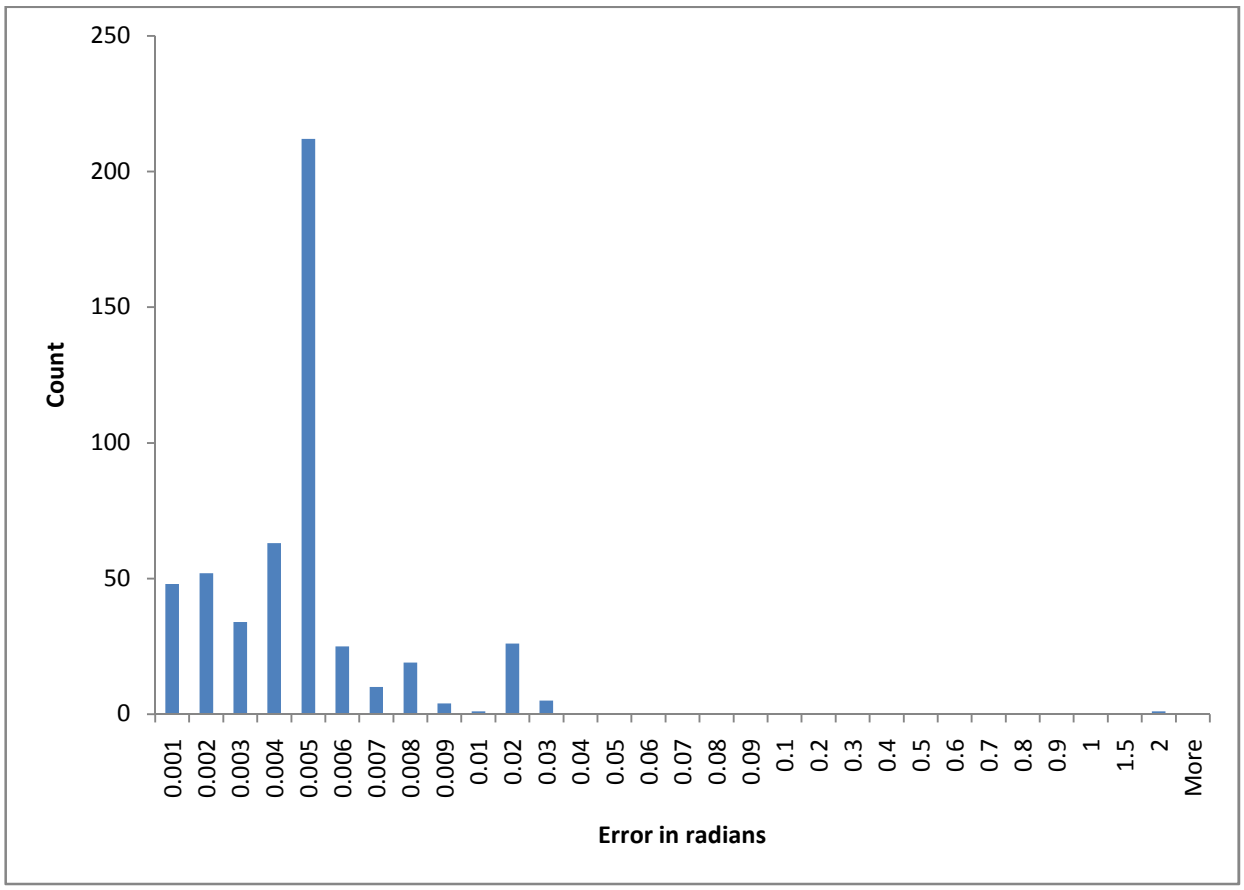

Figure 4. Distribution of errors on the dataset of 500 real images