Alignment of Curved Text Strings for Enhanced OCR Readability

T. Kasar*

Laboratoire LITIS EA-4108, Universite de Rouen, FRANCE 76800

A. G. Ramakrishnan

Department of Electrical Engineering, Indian Institute of Science, INDIA 560012

Abstract

Conventional optical character recognition systems, designed to recognize linearly aligned text, perform poorly on document images that contain multi-oriented text lines. This paper describes a novel technique that can extract text lines of arbitrary curvature and align them horizontally. By invoking the spatial regularity properties of text, adjacent components are grouped together to obtain the text lines present in the image. To align each identified text line, we fit a B-spline curve to the centroids of the constituent characters and normal vectors are computed all along the resulting curve. Each character is then individually rotated such that the corresponding normal vector is aligned with the vertical axis. The method has been tested on images that contain text laid out in various forms namely arc, wave, triangular and combination of these with linearly skewed text lines. It yields 97.3% recognition accuracy on text strings where state-of-the-art OCRs fail before alignment.

Keywords: Curved text alignment, connected component, B-splines, multi-oriented text

IJCVSP
International Journal of Computer Vision and Signal Processing

ISSN: 2186-1390 (Online)
http://www.ijcvsp.com

Article History:
Received: 1 March 2013
Revised: 10 June 2013
Accepted:23 July 2013
Published Online: 23 July 2013

1. INTRODUCTION

Text represents the most important information in a document image. It provides semantic information that may be used to describe the image content. Texts printed in magazines, advertisements, and packing cartons are occasionally oriented arbitrarily for artistic and other purposes. State-of-the-art optical character recognition (OCR) packages are not designed to handle such images that contain text laid out in a curvilinear fashion where the skew of the characters in the text string changes gradually and no two characters may have the same skew. Though there has been a lot of research on document analysis and recognition, there is little work on processing documents that contain text lines with arbitrary curves.

Existing OCR systems perform well only in recognizing texts that are linearly aligned as is evident from Figure 1. While the horizontally aligned text is easily detected and recognized, curved text poses a challenge to recognition. In most existing OCR systems, a skew correction process is often performed prior to recognition, should a need arise. Most skew estimation techniques assume the presence of long and straight text lines. This assumption may not be true for images, which contain short text lines that are oriented in arbitrary directions. These issues call for specialized pre-processing techniques that estimate and correct the skew of individual characters before feeding such a document to an OCR system.

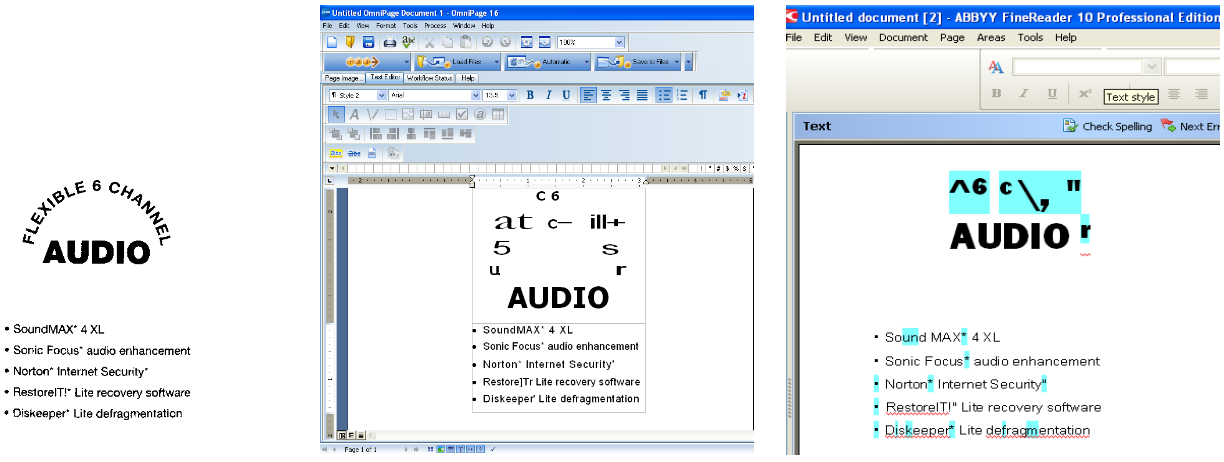
2. Review of related work

Documents that have long parallel text lines require a simple global skew detection and correction. There are a host of well-performing techniques that use projection profile [1, 2], Hough transform [3, 4, 5], principal component analysis [6, 7], nearest neighbor clustering [8, 9, 10] or a combination of them [11] to de-skew such document images.

Pal et al. [12] reported a maiden attempt to handle multiple skew in Bangla and Devanagari script documents using the inherent characteristics of these two scripts. Both the scripts have a headline that connects characters into

*Corresponding author

Email addresses: thotreingam.kasar@inv.univ-rouen.fr (T. Kasar), ramkiag@ee.iisc.ernet.in (A. G. Ramakrishnan)


FLEXIBLE 6 CHANNEL
AUDIO
• SoundMAX* 4 XL
• Sonic Focus* audio enhancement
• Norton* Internet Security*
• RestoreIT!* Lite recovery software
• Diskeeper* Lite defragmentation
Untitled OmniPage Document 1 - OmniPage 16
C 6
at c– ill+
5 s
u r
AUDIO
• SoundMAX' 4 XL
• Sonic Focus' audio enhancement
• Norton' Internet Security'
• Restore]Tr Lite recovery software
• Diskeeper' Lite defragmentation
Untitled document [2] - ABBYY FineReader 10 Professional Edition
^6 c\, "
AUDIO r
• Sound MAX* 4 XL
• Sonic Focus* audio enhancement
• Norton* Internet Security*
• RestoreIT!" Lite recovery software
• Diskeeper* Lite defragmentation

(a) (b) (c)

Figure 1: Performance of state-of-the-art OCR systems on curved text strings. (a) Binary input image containing a curved text string (b) Output of Nuance Omnipage Professional 16 OCR (Trial version) (c) Output of ABBYY FineReader 10 Professional OCR (Trial version). While the horizontally aligned text is easily detected and recognized, the curved text line poses difficulty in segmentation as well as recognition.

a word. The method searches for the upper envelope of each connected component (CC) to obtain the headlines. A clustering procedure is performed on the detected headlines and each of the resulting clusters gives an estimate of the skew of individual text line. Though the method can handle multiple skew, it requires that the document contain long straight text lines and is heavily dependent on the characteristics of the script. It is not applicable to majority of scripts since they do not have a headline. The method is also restricted to documents with skew angle less than 45°.

In [13], the authors proposed a new technique for skew correction in documents that contain several areas of text with different skew angles. Adjacent connected components are grouped using a nearest neighbor approach to form words which are further grouped into text lines. Then, the top-line and baseline for each text line are estimated using linear regression. Finally, the local skew angle of each text area is estimated and corrected independently to horizontal or vertical orientation. However, the method is again limited to de-skewing straight text lines only.

Zhang and Tan [14] use connected component analysis and regression technique to restore curved text lines that arise around the spine of scanned pages of a book. The method first identifies the shaded region and binarizes the image using a modification of Niblack's method to remove the shade. Then, the connected components in the clean area are clustered to obtain text lines, which are modeled by straight reference lines using linear regression. A bottom-up approach is applied to cluster the connected components in the shaded area into warped text lines, and polynomial regression is used to model the warped text lines with quadratic reference curves. The method assumes that the document with moderate skew is scanned in such a way that the text lines are horizontal. It also requires that there are partially straight text lines in the image.

Uchida et al. [15] proposed an interesting approach that could identify the skew angles of individual characters in a document image using an instance-based matching method. Unlike other skew correction techniques, the method does not rely on the local straightness of the text lines and/or character strokes. It can handle documents where the characters do not form long straight lines. The method computes rotation variants and invariants for each character and stores them as instances. Skew is estimated at the connected component level. It identifies the character category, estimates local skew from the stored instances of the identified category and finally determines global skew employing a voting strategy. The font image and the rotation angle for which the match score is maximum yield the character category and the local skew angle. However, the method still assumes a single skew for the whole page and also requires that the documents have fonts similar to the ones that are stored as instances.

A number of methods [16, 17, 18] are proposed to extract curled textline from hand-held camera-captured document images. Such images exhibit 3D deformation and calls for specialized dewarping techniques to flatten the page.

Two methods [19, 20] addressed the alignment of curved text but yielded only partial success. In [19], partial images of character strings are cut out from a color document and assuming that the sequence of the characters constituting the text line is known, quadratic functions are used to approximate the curvature of the string. The type of curvature of the string is determined using heuristics and a suitable correction scheme is applied. The method assumes only one text string per image and cannot handle documents containing multiple text lines. Only linearly skewed and arc-form text are considered and it cannot handle other text layouts such as 'triangular' or 'wave' or a combination of different forms.

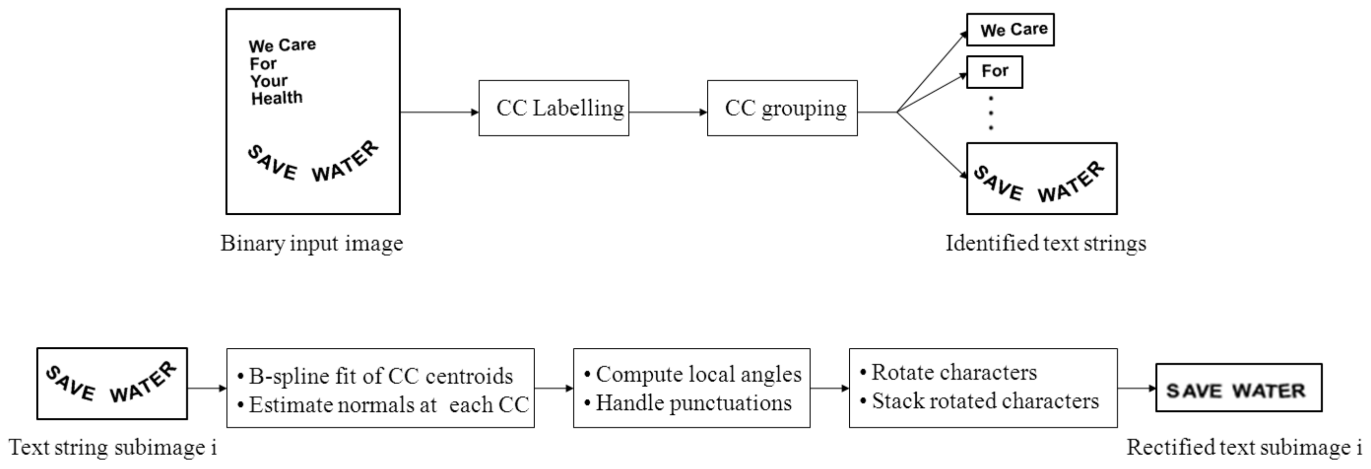


Figure 2: Block diagram of the proposed text alignment method. Text strings are first identified and then individually processed to align them horizontally.

In [20], a method is proposed to transform arc-form text to linear-form-text using concentric ellipsoidal contours. The method assumes that the text in the document is either circular or elliptical in shape and is limited to only the upper half circle or ellipse. The use of ellipsoidal contours also leads to distortion of the characters during the arc-to-linear transformation. This calls for a further de-tilting step. This post-processing step works only for upper case Latin script. Documents with multiple text lines and multiple text orientations cannot be handled. It also focusses only on the alignment of individual characters and does not account for the inter-word gap that is essential to obtain meaningful text.

In view of the limitations of existing methods and the inability of commercial OCRs to recognize with curved text, we propose a versatile method [21] to align text laid out in a variety of layouts such as arc, wave, triangular or their combination with linearly-skewed text lines. It also addresses another important aspect of maintaining the inter-character and inter-word gaps so that the rectified text string can be 'meaningfully' recognized. The method renders curved text a linear form so that it can be recognized using off-the-shelf OCRs.

3. Proposed methodology

In this paper, we propose a novel method to extract and align multiple text strings in documents having arbitrary curvature. It consists of two processing stages namely the text string identification step that associates adjacent CCs into text strings and the alignment step that processes each text string individually and transforms it into a form suitable for OCR. A schematic block diagram of the proposed method is shown in Figure 2. We assume that the input image to the text alignment block contains only clean binarized text extracted from documents images. Text detection is a different problem and we do not deal with it here.

3.1. Grouping of CCs into text strings

Since text normally exhibits spatial regularity, we seek to obtain text strings by grouping adjacent CCs based on their spatial proximity and size and orientation regularity. An 8-connectivity based component labeling is performed on the binary image to obtain a set of 'M' disjoint components $\{\mathcal{CC}_j\}, j = 1, 2, \cdots, M$. The heights of characters in a text string are generally more 'stable' than their widths and do not vary even if they are represented in italics or bold. Making use of this property during the search for and grouping of similar adjacent CCs, the local orientation of the text line is first determined from the angle between the centroids of a reference CC and its nearest neighbor (NN). If the angle lies in the intervals $[45°, 135°]$ or $[225°, 315°]$, the orientation of the text is assumed to be vertical; otherwise, it is horizontal. The estimated angle and orientation guide the subsequent NN search and grouping. Depending on whether the estimated text orientation is horizontal or vertical, the NN-search distance is made proportional to the height or width, respectively of the reference CC denoted by $\mathcal{CC}_{ref}$.

For a given $\mathcal{CC}_{ref}$ and its nearest neighbor $\mathcal{CC}_{nn1}$, we find $\mathcal{CC}_{nn2}$, which is the subsequent NN of $\mathcal{CC}_{nn1}$. The resulting triplet is analyzed for similarity of sizes and angles and is grouped together whenever all the following conditions are satisfied:

$$|\phi^{ref}_{nn1} - \phi^{ref}_{nn2}| < T_1 \tag{1}$$

$$T_2 \le \frac{\delta_{ref}}{\delta_{nn1}}, \frac{\delta_{ref}}{\delta_{nn2}} \le T_3 \tag{2}$$

$$\mathrm{D}^{ref}_{nn1}, \mathrm{D}^{nn1}_{nn2} < T_4\, \delta_{ref} \tag{3}$$

where ϕ^{ref}_{nni} denotes the anti-clockwise angle of the line joining the centroids of the reference component and its i^{th} neighbor with respect to the horizontal, the parameters $\delta_{ref}, \delta_{nn1}$ and δ_{nn2} represent the heights or widths of $\mathcal{CC}_{ref}$, $\mathcal{CC}_{nn1}$ and $\mathcal{CC}_{nn2}$, respectively depending on the search direction. D^{ref}_{nn1} and D^{nn1}_{nn2} denotes the Euclidean

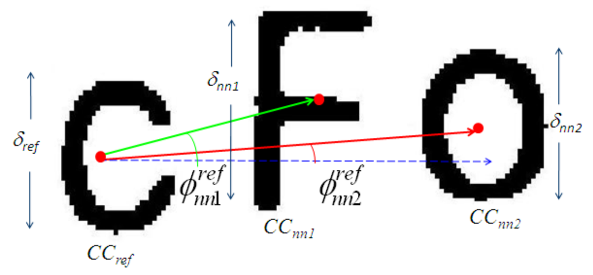


Figure 3: The parameters that guide the successive nearest neighbor search for grouping them into text strings.

distances between the centroids of ($\mathcal{CC}_{ref}$, $\mathcal{CC}_{nn1}$) pair and ($\mathcal{CC}_{nn1}$, $\mathcal{CC}_{nn2}$) pair respectively. These parameters are illustrated in Figure 3. Equation 1 checks that the difference between the angles of the two neighbors with respect to the reference component lies within a threshold value. Equation 2 tests the similarity in their sizes while Equation 3 restricts the search distance for locating the next neighbor. Since components of a text string exhibit some degree of spatial, size and orientation regularity in general, the parameters T_1, T_2, T_3 and T_4 work well for values in the range [20 30], [0.25 0.5], [1.5 2.0] and [2 3] respectively. In this work, these heuristic parameters are set to 25, 0.35, 1.75 and 2.5, respectively. The search process is repeated by making $\mathcal{CC}_{nn1}$ as the next reference component $\mathcal{CC}_{ref}$. This process is continued till all the CCs are considered and we obtain the individual text lines $\{\mathcal{T}^k\}, k = 1, 2, \cdots, K$ where K is the total number of detected text lines.

3.2. Fitting a B-spline curve to the centroids

B-splines are piecewise polynomial functions that can provide local approximations of contours of arbitrary shapes using a small number of parameters [22]. We employ B-splines in our work because they capture the smoothness inherently present in any text layout. Since polynomial fitting results in numerical problems for vertically aligned text, the usual practice is to swap the x and y coordinates before curve fitting [19]. The issue still remains for an image that contains horizontal as well as vertical text lines. B-splines can be used to represent curves of any shape thereby making it an ideal choice for handling images that contain multiple text lines with different curvatures.

For each identified text string $\mathcal{T}^k$, the centroids of the constituent characters are identified and represented as follows:

$$\mathbf{C}^k_j = \{(C^k_{x,j}, C^k_{y,j})\}, \quad j = 1, 2, \cdots, N^k \tag{4}$$

where 'N^k' is the number of characters in the k^{th} text string. These points serve as the control points for fitting B-spline curves. The i^{th} point in the resulting curve is denoted by:

$$\mathbf{p}^k_i = \{x^k_i, y^k_i\} \tag{5}$$

The order of the spline curve is chosen to be 4 so that the resulting curve is smooth and offers robustness to the zig-zag pattern of the control points that arises due to the presence of ascenders and descenders. It was observed that increasing the order of the spline does not have any significant impact on final result of alignment.

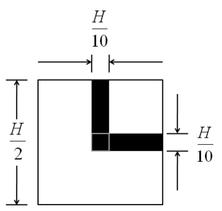


Figure 4: Structuring element used to group punctuation marks and compound characters. The size of the structuring element is proportional to the height of the character being processed.

3.3. Computing normals to the curve at each CC

The normal vector at the i^{th} point of the curve is then computed using the following relation.

$$\mathbf{n}^k_i = \begin{bmatrix} cos(\frac{\pi}{2}) & -sin(\frac{\pi}{2}) \\ sin(\frac{\pi}{2}) & cos(\frac{\pi}{2}) \end{bmatrix} \times \frac{1}{2}\left(\frac{\left(\mathbf{p}^k_i - \mathbf{p}^k_{i-1}\right)}{\left\|\mathbf{p}^k_i - \mathbf{p}^k_{i-1}\right\|_2} + \frac{\left(\mathbf{p}^k_{i+1} - \mathbf{p}^k_i\right)}{\left\|\mathbf{p}^k_{i+1} - \mathbf{p}^k_i\right\|_2}\right)$$

Here, the subscript i denotes the position on the spline curve at which the normal vector is computed and $\|\cdot\|_2$ denotes the L_2 norm.

3.4. Horizontal alignment of text

The normal vectors, thus locally determined, give a fairly good estimate of the orientation of the individual characters. Thus, we can obtain the rectified text string by simply rotating each character such that the normal vector is aligned vertically. The required angle of rotation for any character is computed as:

$$\theta^k_i = (90^\circ - \angle \mathbf{n}^k_i) \quad i = 1, 2, \cdots, N^k \tag{6}$$

The values of θ^k are indicative of the type of orientation of the text string $\mathcal{T}^k$. Here, a positive/negative value of θ implies rotation in an anti-clockwise/clockwise direction. For a linearly skewed text string, the standard deviation of the θ^k is 'small'. In such a case, the whole text string may be rotated by a global angle which is computed as the median value of θ^k. On the other hand, if θ^k varies progressively from positive to negative values or vice versa, once or multiple times, the text string is curved. In this case, each character is individually rotated.

3.5. Handling punctuations

Though the method can deal with arbitrary text line orientations by estimating and correcting the skew of each character individually, the way the CCs are grouped into text lines ignores small components such as punctuation marks and the '·' associated with the letters 'i' and 'j'. In order to group punctuation marks, we design a structuring element as shown in Figure 4, whose size is proportional

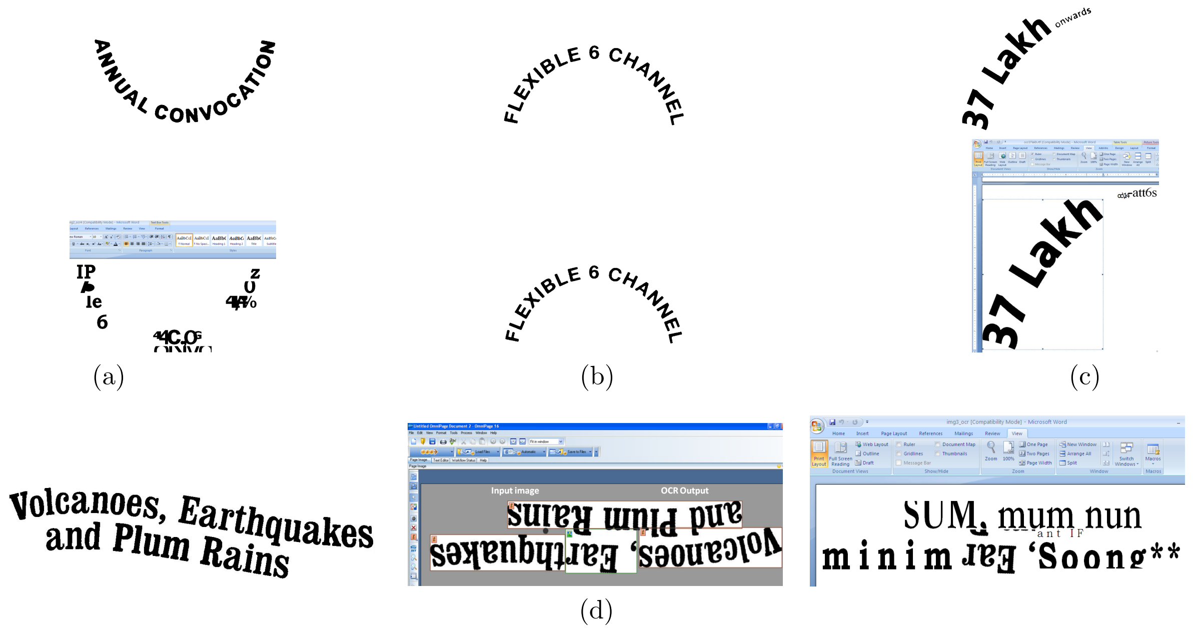


Figure 5: OCR readability on images with curved text lines. (a) Erroneous OCR output (b) No text detected (c) The OCR output is a table with an image in the first column and erroneous text in the second column (d) Image inverted by software and the corresponding erroneous OCR output.

to the height H of the character under consideration and the black regions indicate 'ON' pixels.

Each character in a text string is dilated using the structuring element and we check for the existence of other CCs in the dilated region that do not belong to any of the identified text strings. If any such CCs are encountered, they are combined with the character under consideration. The size of the structuring element is adaptively set to half the height of the character being processed.

The output image is generated by stacking the rotated characters in a left to right direction such that the horizontal spacing between the characters is proportional to the corresponding inter-character centroid distances. It may mentioned that the character rotation is carried out using bilinear interpolation. The inter-character spacing ($S^k_{j,j+1}$) between the component $\mathcal{CC}_j$ and $\mathcal{CC}_{j+1}$ is computed as follows:

$$S^k_{j,j+1} = D^k_{j,j+1} - \left(\frac{W^k_j + W^k_{j+1}}{2}\right) \qquad (7)$$

where $D^k_{j,j+1} = \sqrt{(C^k_{x,j} - C^k_{x,j+1})^2 + (C^k_{y,j} - C^k_{y,j+1})^2}$ is the Euclidean distance between the centroids of the j^{th} and $(j+1)^{th}$ components and W^k_j represents the width of the component $\mathcal{CC}_j$.

Due to the presence of ascenders and descenders, the rotated characters need to be positioned with proper offset values in the vertical axis. The spline curve represents the curvature of the text string and is used for aligning the rotated characters. The mid-point of all the pixels of each character intersected by the fitted curve is determined. Each rotated character is arranged such that this computed mid-point is aligned with the horizontal axis. However, it may be observed that 'small' vertical misalignments still remain due to the zig-zag pattern of the centroids of the characters in a text string.

4. Experiments and results

There is no standard image database available for curved text. So, in order to test the performance of the method, we have collected a set of images containing 100 curved text strings. Even though our database is small, it contains all the possibilities of text orientations and layout styles such as arc, wave, triangular and a combination of these with linearly-skewed text lines. We use Nuance Omnipage professional 16 (trial version) OCR software to evaluate the performance of our method.

Some example outputs of applying OCR directly on the input images are shown in Figure 5. Without the text alignment preprocessing step, the OCR software yields only erroneously recognized characters in most cases (see 5(a)) or even fails to detect any text at all (see 5(b)) in some images. Figure 5(c) shows a case where a curved text string is detected as a 2-column table with a partial image in one column and text in the other. Figure 5(d) shows a particular case where the input image is automatically inverted by the software before recognition and subsequently identifies 3 text blocks and 1 image block. The resulting OCRed output is therefore fully erroneous. It may be observed that even though the input binary images are noise free, segmentation of these images into text lines, words and characters is difficult owing to complex text layouts. Clearly, a document input with straight text lines seems to be necessary for current OCR systems to work reliably.

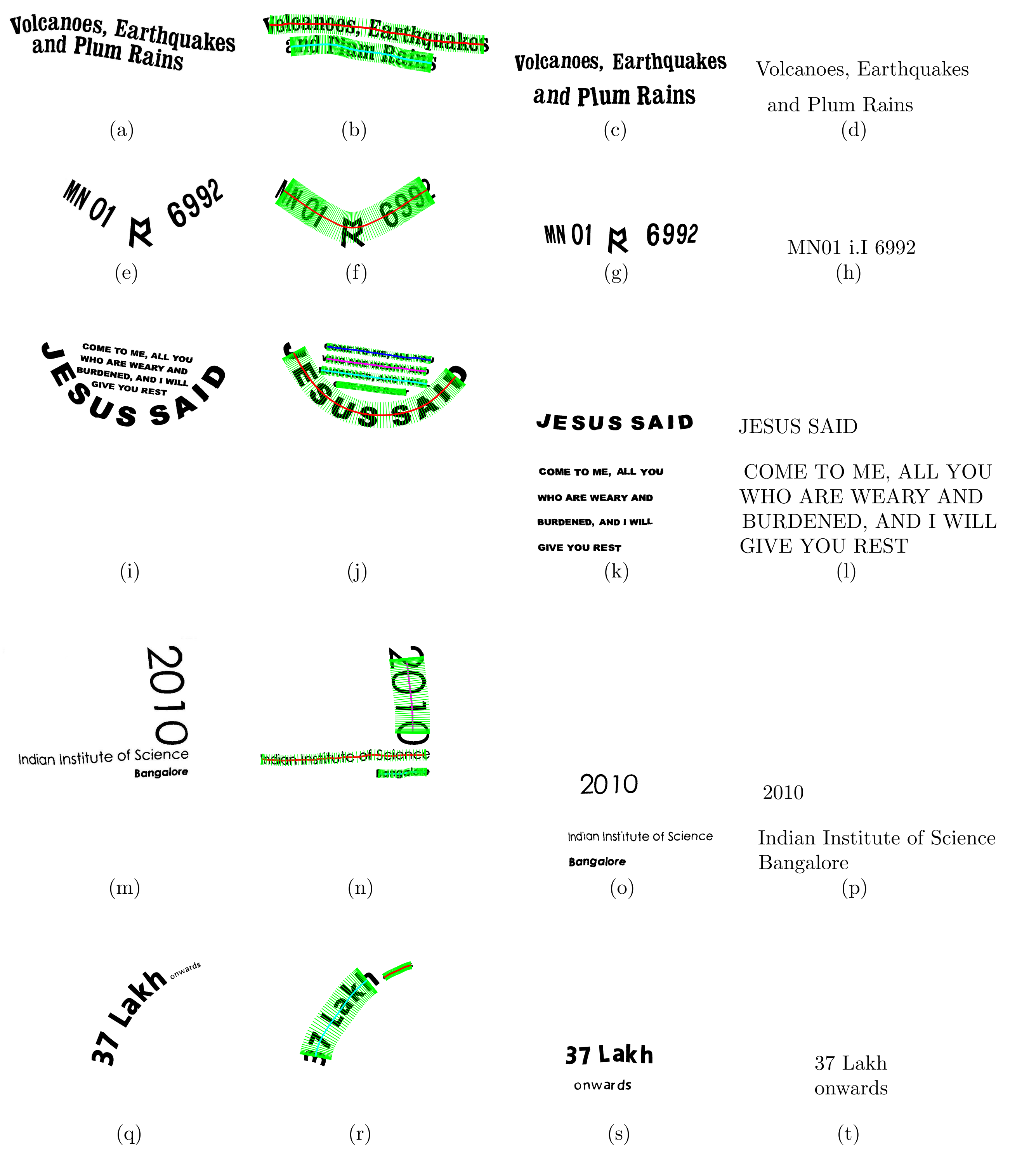


Figure 6: (a, e, i, m, q) Input images containing arbitrary text orientations. (b, f, j, n, r) Identified text strings and the estimated normal vectors. (c, g, k, o, s) Rectified images of the individual text strings. (d, h, l, p, t) Corresponding OCR outputs.

Figure 6 shows the versatility of the method in aligning various text layouts such as wave, triangular, arc or their combination with linearly skewed text lines. Our method successfully aligns curved text without any prior knowledge of the orientation of the text string. While the OCR software cannot recognize any of the curved text strings, the recognition accuracy improves significantly when fed with the output images of the proposed text alignment technique. In Fig 6(h), the character R of the rectified word image is wrongly recognized. This may be attributed to its unconventional font style. Since the OCR software yields only erroneous results on the raw input images, they are not included in the figure. The overall recognition accuracy obtained on the whole dataset is 97.3%.

Table 1: Comparison of our method with the method proposed in [20] using Nuance Omnipage Professional 16 (Test images from [20] with permission).

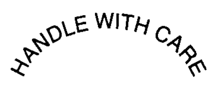
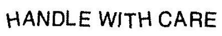
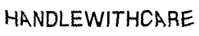
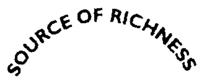
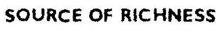
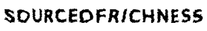
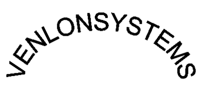
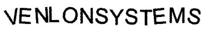
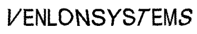
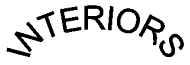
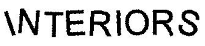
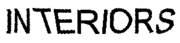
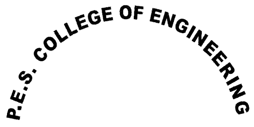
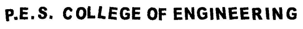
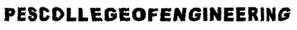
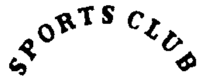
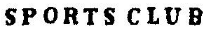
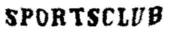
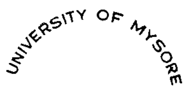
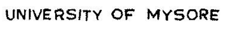
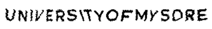
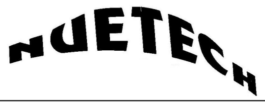
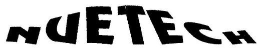
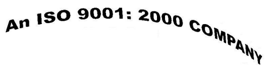
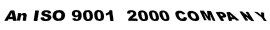
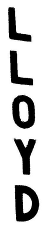
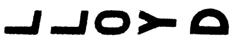
Input image	Output image obtained using our method and corresponding OCR outputs	Output image of method [20] and corresponding OCR outputs
	 HANDLE WITH CARE	 HNNDLEWITHCARE
	 SOURCE OF RICHNESS	 SOURCEDFRICHNES5
	 VENLONSYSTEMS	 VENLONSYSTEMS
	 INTERIORS	 IN'TERIDRS
	 P.E.S. COLLEGE OF ENGINEERING	 PESCOLLEGEOFENGINEERING
	 SPORTS CLUB	 SPORTSCLUO
	 UNIVERSITY OF MYSORE	 UNIVERS\7YOFMYSORE

Table 1 compares the outputs of our method with that of the method proposed in [20]. While our method yields accurate OCR outputs, the outputs of method [20] exhibit deformations leading to an increased error rate. It may be observed that our method not only aligns curved text strings successfully, but also maintains the inter-character spacings which is essential to obtain meaningful words. The method proposed in [20] ignores the inter-character spacings and combines all the words into a single character string. Their method is applicable only to uppercase English letters that are arranged in an arc-form. Further, it cannot handle the presence of multiple text lines in the same image. Our method overcomes all these limitations and since it involve only character rotation, it is free from character deformation resulting in a better OCR performance.

Table 2: Some failure cases of the proposed method.

Input image	Rectified image	OCR output before alignment	OCR output after alignment
		aretlEttk$_4$	NwaEltiv'.
		POI iso 9001 ta. 2000 tom, [illegible]	An ISO 9001 2000 CONtAnOtv4i.
		L L 0 Y D	J J C) >mu CI

Some instances of typical OCR errors encountered by our method are shown in Table 2. These may be attributed to the presence of unconventional font styles, perspective distortion and deformation, which are in general very difficult for any OCR system. In the third example, the orientation of the text line is vertical but the characters are upright. Our method automatically assumes that the direction of the characters is normal to the direction of the text line, resulting in each of the character being rotated by 90 degrees.

5. Conclusions and future work

Just like the skew detection and correction steps in conventional OCRs, alignment of curvilinear to rectilinear text is an indispensable pre-processing step in the analysis of newer document types that contain multi-oriented text strings. We have proposed a novel method for the alignment of curved character strings so that off-the-shelf OCRs can be used to recognize them. Our method can handle multiple text lines printed in various layout styles. The effectiveness of the method is amply illustrated by our experiments with various types of orientations of the text strings. While state-of-the-art OCRs fail on the input images, our method of text alignment ensures a recognition accuracy comparable to that of unskewed text.

However, due to the presence of acsenders and descenders in lower case letters, the positions of the centroid exhibit a zig-zag pattern resulting in small errors in the local skew estimate of some characters. This is observed to be within the skew tolerance of the OCR system. The method assumes that the text string as a whole, along with its constituent characters is curved. It assumes only rotation and not major distortions of the characters. It does not handle rotation of the individual characters independent of the orientation of the text string. Upright characters placed on a wavy boundary cannot be handled. Thus, there is a lot of scope for enhancing the accuracy and robustness of the method to complex backgrounds.

Our method does not assume any script-specific characteristics and can be adapted for any script. However, the current approach for identification of the text strings needs to be adapted for other scripts such as Indic or Chinese that have compound characters composed of more than one CC. Our future work is to augment the method with a trained classifier for robust detection of curved multi-script text from real-world scene images.

References

[1] A. Bagdanov, J. Kanai, Projection profile based skew estimation algorithm for JBIG compressed images, in: Proceedings of Intl. Conf. Document Analysis and Recognition, 1997, pp. 401–405.

[2] H. Baird, The skew angle of printed documents, in: Proceedings of Conf. Society of Photographic Scientists and Engineers, Vol. 40, 1987, pp. 21–24.

[3] U. Pal, B. Chaudhuri, An improved document skew angle estimation technique, Pattern Recognition Letters 17 (8) (1996) 899–904.

[4] S. N. Srihari, V. Govindaraju, Analysis of textual images using the hough transform, Machine Vision and Applications 2 (3) (1989) 141–153.

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

[6] O. Okun, M. Pietikainen, J. Sauvola, Document skew estimation without angle range restriction, Pattern Recognition 2 (1999) 132–144.

[7] T. Steinherz, N. Intrator, E. Rivlin, Skew detection via principal component analysis, in: Proceedings of Intl. Conf. on Document Analysis and Recognition, 1999, pp. 153–156.

[8] A. Hashizume, P. Peh, A. Rosenfeld, A method for detecting the orientation of aligned component, Pattern Recognition Letters 4 (8) (1986) 125–132.

[9] Y. Lu, C. L. Tan, A nearest-neighbor chain based approach to skew estimation in document images, Pattern Recognition Letters 24 (14) (2003) 2315–2323.

[10] L. O'Gorman, The document spectrum for page layout analysis, IEEE Trans. PAMI 15 (1993) 1162–1173.

[11] K. Mahata, A. Ramakrishnan, Precision skew detection through principal axis, in: Proceedings of Intl. Conf. Multimedia Processing and Systems, 2000, pp. 186–188.

[12] U. Pal, M. Mitra, B. Chaudhuri, Multi-skew detection of indian script documents, in: Proceedings of Intl. Conf. Multimedia Processing and Systems, 2001, pp. 292–296.

[13] P. Saragiotis, N. Papamarkos, Local skew correction in documents, Intl. Jl. Pattern Recognition and Artificial Intelligence 22 (4) (2008) 691–710.

[14] Z. Zhang, C. Tan, Correcting document image warping based on regression of curved textlines, in: Proceedings of Intl. Conf. Document Analysis and Recognition, Vol. 1, 2003, pp. 589–593.

[15] S. Uchida, M. Sakai, M. Iwamura, S. Omachi, K. Kise, Skew estimation by instances, in: Proceedings of Intl. Workshop Document Analysis Systems, 2008, pp. 201–208.

[16] C. Wu, G. Agam, Document image de-warping for text/graphics recognition, in: Proceedings of Structural Syntactic and Statistical Pattern Recognition, 2002, pp. 348–357.

[17] A. Ulges, C. Lampert, T. M. Breuel, Document image dewarping using robust estimation of curled text lines, in: Proceedings of Intl. Conf. Document Analysis and Recognition, Vol. 2, 2005, pp. 1001–1005.

[18] S. S. Bukhari, F. Shafait, T. M. Breuel, Coupled snakelet model for curled textline segmentation of camera-captured document images, in: Proceedings of Intl. Conf. Document Analysis and Recognition, 2009, pp. 61–65.

[19] H. Hase, M. Yoneda, T. Shinokawa, C. Y. Suen, Alignment of free layout color texts for character recognition, in: Proceedings of Intl. Conf. Document Analysis and Recognition, 2001, pp. 932–936.

[20] T. Vasudev, G. Hemanthkumar, P. Nagabhushan, Transformation of arc-form-text to linear-form-text suitable for OCR, Pattern Recognition Letters 28 (2007) 2343–2351.

[21] T. Kasar, A. G. Ramakrishnan, A method to extract and align text of an image captured and a system thereof, Indian Patent Office Reference. No: 109/CHE/2011.

[22] D. Rogers, J. Adams, Mathematical Elements for Computer Graphics, McGraw-Hill, 1990.