

AN AUTOMATED APPROACH TO REMOVE LINE FROM TEXT BYPASSING RESTORATION STAGE

Amjad Rehman Khan^{#1}, Fajri Kurniawan^{*2}, Dzul kifli Mohamad^{*3}

[#]Department of Computer Science, Bahauddin Zakaryia University Multan Pakistan
¹amjadbz2003@yahoo.com

^{*}Department of Computer Graphics and Multimedia, Universiti Teknologi Malaysia
 Skudai, Johor Bahru, Malaysia 81310

²kfajri2@siswa.utm.my

³dzulkifli@utm.my

Abstract—In composite document image, handwritten and printed text is often found to be overlapped with printed lines. The problem becomes critical for obscure and broken lines at multiple positions. Consequently, line removal is unavoidable pre-processing stage in the development of robust object recognizers. Moreover, the restoration of the handwritten area after removal of lines still persists to be a problem of interest. This paper presents a new approach to detect and remove unwanted printed line inherited in the text image at any position without characters distortion to avoid restoration stage. The proposed technique is based on connected component analysis. Experiment is conducted using single line images that scanned and extracted manually from several documents and forms. It is demonstrated that our approach is equally suitable to deal with line removal in printed and handwritten text written in any language circumvent restoration stage.

Keywords—Underline detection, line removal, connected components analysis, character restoration.

I. INTRODUCTION

Lines are frequently used in many documents such as bank checks, mail order forms, tax forms or admission forms [1]. Among the others, text overlapping with lines poses serious problems for OCR performance. The dilemma becomes critical for composite documents containing printed and handwritten text overlapped with lines [3]. Detection and removal of these factors through pre-processing techniques can be helpful to improve recognition rates [2]. The work reported in this paper deals with the problem of how to detect and remove lines in a composite document overlapped with printed and handwritten text. Beside that, following line detection and removal restoration of broken characters still persists to be problem of interest. Some examples of single text line images with overlapped line are illustrated in Fig. 1 that are taken manually scanning several documents and form that contain lines. In the simplest case, the line in Fig. 1(c) is just underline. The lines in Fig. 1(a),(b) are overlapped with some characters and broken as well, thus are named as overlapped lines. The Fig 1(f) presents handwritten text overlapped with line and finally Fig. 1(g) show some Chinese

text with line. By playing with one of the best commercial OCR products on the market, namely Fine Reader 7.0 [18], we observed that:

- 1) It can remove most of the touched, broken and curved underlines
- 2) It does not work very well for difficult cases as in Fig. 1(b),(d), where there exist some black pixels under the underline, or line is both curved and broken
- 3) It failed to extract line that present in the middle of the text
- 4) Finally, it can't deal with line in the script as in Fig. 1 (e),(f), (g). Although good techniques obviously have been developed for underline removal in OCR by some leading companies, but still they can deal with underline in the printed text only.

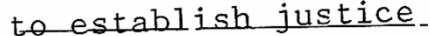
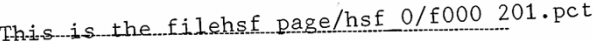
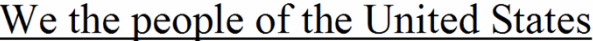
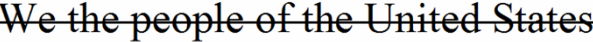
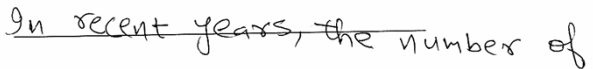
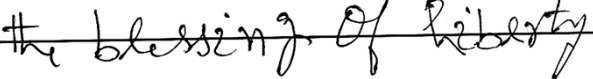
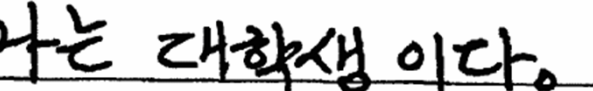
a	
b	
c	
d	
e	
f	
g	

Fig 1. Examples of Latin and Chinese single text line images with some form of line in printed and script writing

II. REVIEW OF THE RELATED WORK

Several approaches have been proposed for underline removal in the literature. Most of them detached underline from binarized image by the dilation and erosion operators of the mathematical morphology [3, 4, 5, 6]. Dilation was applied until all the lines longer than a fixed threshold are removed from the underline region. On the other hand this operator shattered the characters and therefore it became difficult to recognize. Hence erosion was applied to recover the lost parts of the characters. However, broken characters could not restore correctly [7]. Govindaraju and Srihari [9] achieved underline removal by using the “good continuity criterion”. The criterion first detects the smooth strokes in the image and then identifies the spine of the image as the smooth stroke with maximum length and finally is removed. However, this approach worked out on thinned images and therefore it requires a preliminary time consuming process. Secondly, global properties of textual word shapes and those of interfering strokes were used to separate them. Yu and Jain [13] have proposed a method for line removal and character restoration using Block Adjacency Graph representation of the input binary image. The horizontal form lines were located by finding long straight lines based on the block adjacency graph. Form line separation and character reconstruction were also implemented from this graph.

Yoo et al., [14] have classified the various types of junction points at the point of contact or crossing over of the characters and the line. After line removal, the junction points are detected and restored based on their classification type. Koerich and Ling [15] have detected and removed the lines using horizontal projection profile (HPP). The removed regions are rectified by checking the neighbours for every pixel that could be fitted into the erased line. Based on whether the neighbouring pixels satisfy certain condition or not, they decide to leave the pixel on or off.

Blumenstein et al., [12] introduced new pre-processing techniques for underline removal and restoration based on horizontal black pixel runs. It is assumed that word stroke thickness will be similar to the thickness of the underlines present in the word. However this assumption is not true in all cases particularly for printed documents/forms (see figure 2.3). Though, underline removal and restoration did not perform well on some of the more erratic underlines that were present in some word images. Therefore remainders of undetected underlines were removed manually to facilitate further processing [12].

In some algorithms such as proposed in [8], broken characters are restored. If result that character recognizer is performed with the restored characters is wrong, restored characters are sent back to the restoration algorithm stage. In such methods, the processing time was increased because it has feedback paths. In addition, characters are sometimes recognized incorrectly such as ‘h’ and ‘b’ [3].

Bai and Huo [11] used strategies of connected component and bottom edge analysis to detect underline in printed text. Prior to removal of detected underline, an OCR engine is used to recognize and verify the input text line. However the

approach dealt with underline in printed text and failed in script line removal.

Recently, Arvind et al., [16] detect multiple printed lines with varying thickness present in the word image using horizontal projection profile. Restoration of the smashed characters is performed by using Bresenham line drawing algorithm [17]. However, technique can not deal with restoration of printed characters and skewed images. To conclude, techniques developed so far are:

1. Computationally expensive as consists of two stages: line removal and restoration of smashed characters.
2. Deal with underline removal in printed text rather line removal.
3. Can not deal with line removal in script writing.

Keeping in view problems in the commercial tools and techniques reported in the literature, we develop a solution to detect and remove line in the composite documents. In the following, we describe in detail how our approach works and report the results of a set text lines that consist of 568 images to demonstrate the effectiveness of proposed approach.

III. PROPOSED APPROACH

Given a binary image of a text line, our approach works as follows: First, an underline detection module is applied to detect whether there exists any line. Afterward, a line removal module is used to remove different kinds of lines without shattered some part of characters pixels.

A. Line Detection Module

Firstly, to detect unwanted line, procedure starts by tracing left most foreground pixel. The traced component is analysed by checking connected pixels (without drastic change) from left to right along with record of length to save “t/T”, “J” bar. Connected components are considered line if its length is greater than quarter of the width of word image ($\frac{1}{4}w$). Prior to removal of the detected line, junctions are examined by tracing pixels up, down and diagonally with some threshold in order to avoid character distortion.

1) Detection via Connected Components Analysis

The precise analysis is a prerequisite to detect unwanted lines and junctions. A junction point is defined as a contact point of characters with line as shown in Figure 1. In implementation there are multiple lines with varying thickness present at any position of the printed/ handwritten word image. Prior to the removal of the line, junctions are examined at each pixel to avoid characters smash-up and to skip restoration stage.



Fig. 1 Line and junction points' detection

The proposed approach is based on connected components analysis. Initially, the image is cleaned of all noise elements such as spurious dots and lines. The procedure starts by detecting unwanted line and checking of junction points prior to removal of the line to avoid characters smash-up.

Let say an image denoted by P where $P \in \{0,1\}$, (1 represent foreground pixel)

$$p_{i,j} \in P, \begin{cases} i = 1, 2, \dots, h \\ j = 1, 2, \dots, w \end{cases}$$

Where h, w are height and width of P respectively.

Define origin as $O = \{o \in P \mid o = 1\}$. Take origin point $o_{a,b} \in O$, a and b is left most of P .

(i) Define

$L = \{l \in \{p_{i,j}\} \mid p_{i,j} = 1, p_{i,j+1} = 1, j = 1, 2, \dots, a, a \leq w\}$ Start from $o_{a,b}$ trace line (L) to right direction allowing one pixel upward and downward continually.

(ii) calculate $length(L) = \begin{cases} length(L) + 1, p_{i,j} = 1 \\ length(L), p_{i,j} = 0 \end{cases}$

(iii) If $length(L) < \frac{1}{4}w$ then do step (i).

B. Line removal module

Connected components of the detected line are simply converted from foreground to background pixels except at junction points to avoid restoration stage. Checking of junction up, down and diagonal is limited by threshold that is calculated from thickness (t) of the lines.

Define U as diagonal connected pixels

$$U = \left\{ u_k \in \{p_{i,j}\} \mid p_{i+r, j+r} = 1, -\left\lfloor \frac{t}{2} \right\rfloor \leq r \leq \left\lfloor \frac{t}{2} \right\rfloor, k = 1..t \right\}$$

If $p_{i,j} \in L$ and $\begin{cases} p_{i-1,j} \neq 1 \\ p_{i+1,j} \neq 1 \end{cases}$ and $length(U) < thick$

then $p_{i,j} = 0$ (changing foreground to background)

IV. EXPERIMENTAL RESULTS AND ANALYSIS

To verify the effectiveness of our approach, a benchmark test is performed. All testing images are single text line images that scan manually from documents and forms that contain of lines. Our benchmark test results are summarized in Tables 1. There is no objective method to estimate the effectiveness of pre-processing techniques, the most common way is by sight, even so, opinions can differ [10]. Therefore the judgment of the correctness of line removal is made by author's visual inspection of the processed images. Figure 2 exhibits results of line removal for all images in Figure 1.

From table 1 it is observed all text images with underline are correctly detected and removed, line detection for text overlapped with line is 98.26% and 95.29%. Likewise results for broken line in text are 97.61% and 90.24% respectively. As the performance does not reach perfectly; it is because

some pure text images are wrongly output with some pixels being incorrectly removed as line(s). However this reason is more acceptable for script line removal. The overall line removal accuracy is 92.42%. This confirms the effectiveness of our approach for line detection and removal from printed text and script writing except minor negative effect on original image.

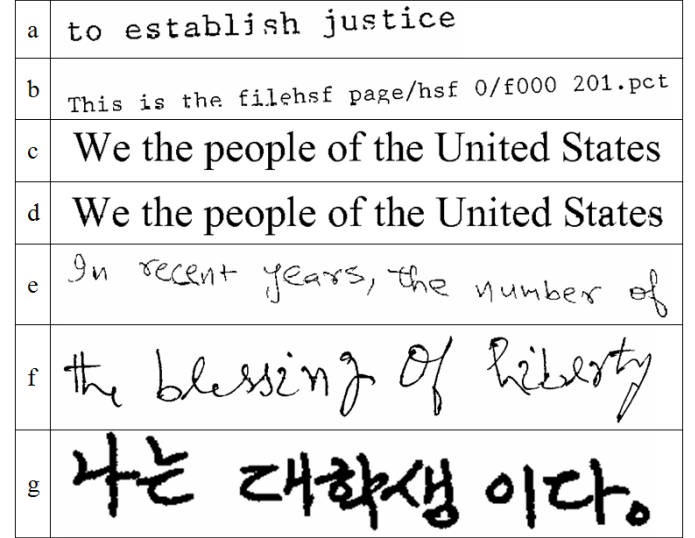


Fig 2. Results of line removal from printed and script for images in Fig 1.

Table 1. Benchmark test results for images with some lines

	Line in Printed text (342)		
	Underline (127)	Overlap line (173)	Broken line (42)
Line detected	127 (100%)	170 (98.26%)	41 (97.61%)
Correctly removed	127 (100%)	162 (95.29%)	37 (90.24%)
	Line in Script writing (226)		
	Underline (73)	Overlap line (119)	Broken line (34)
Line detected	73 (100%)	113 (94.95%)	31 (91.17%)
Correctly removed	73 (100%)	97 (85.84%)	29 (93.54%)

V. CONCLUSION

This paper presented new approach to detect and remove an unwanted line from printed text and script without any character strokes distortion that of course avoids restoration stage. Our approach has been confirmed to deal with touch, overlapped, broken, dashed line detection and removal. This was our first step towards document analysis and recognition. In future, we shall explore new techniques for line segmentation, word segmentation, character segmentation and finally character recognition.

REFERENCES

- [1] G. Dimauro, S. Impedovo, G. Pirlo, and A. Salzo, "Removing Underlines from Handwritten Text: An Experimental Investigation,

- Progress in Handwriting Recognition," A.C. Downton & S. Impedovo, *World Scientific Publishing*, pp. 497-501, 1997.
- [2] A.W. Senior and A.J. Robinson, "An Off-line Cursive Handwriting Recognition System," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 309-321, 1998.
- [3] J.Y. Yong, M.K. Kim, S.W. Bana and Y.B. Kwon, "Line Removal and Restoration of Handwritten Characters on the Form Documents," *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, vol. 1, pp. 128-131, 18-20 Aug. 1997.
- [4] X. Ye, M. Cheriet and C.Y. Suen, "A generic method of cleaning and enhancing handwritten data from business forms," *International Journal on Document Analysis and Recognition*, vol. 4, pp. 84-96, 2001.
- [5] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.
- [6] R. Charles, Giardina, R. Edward, Dougherty, *Morphological Methods in Image and Signal Processing*, Prentice Hall, Inc, pp. 264-279, 1988.
- [7] D. Guillevic and C.Y. Suen, "Cursive Script Recognition: A Fast Reader Scheme," *Proceedings of the 3rd International Conference on Documents Analysis and Recognition*, pp. 311-314, 1993.
- [8] D. Wang and S.N. Srihari, "Analysis of Form Images," *Proceeding of the International Conference on Document Analysis and Recognition*, pp. 181-186, 1991.
- [9] V. Govindaraju and S.H. Srihari, *Separating Handwritten Text from Interfering Strokes, From Pixels to Features III-Frontiers in Handwriting Recognition*, S. Impedovo, J.C. Simon (eds.), North-Holland Publication, pp. 17-28, 1992.
- [10] E. Kavallieratou, N. Fakotakis and G. Kokkinakis, "A Slant Removal Algorithm," *Pattern Recognition*, vol. 33, no. 7, pp. 1261-1262, 2000.
- [11] Z.L. Bai and Q. Huo, "Underline Detection and Removal in a Document Image Using Multiple Strategies," *Proceedings of the 17th International Conference on Pattern Recognition (ICPR'04)*, pp. 1051-4651, 2004.
- [12] M. Blumenstein, C.K. Cheng and X.Y. Liu, "New Preprocessing Techniques for Handwritten Word Recognition," *Proceedings of 2nd International Conference on Visualization, Imaging and Image Processing*, ACTA. Press, Calgary, pp. 480-484, 2002.
- [13] B. Yu, A.K. Jain, "A generic system for form dropout", *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 11, pp. 1127-1132, 1996.
- [14] J. Yoo, M. Kim, S.Y. Han, Y. B. Kwon, "Line removal and restoration of handwritten characters on the form documents", *Proc. 4th Int. Conf. on Document Analysis and Recognition*, pp. 128-131, 1997.
- [15] A.L. Koerich, L.L. Ling, "A system for automatic extraction of the user-entered data from bank checks", *Proc. of Int. Symposium on Computer Graphics, Image processing and Vision*, pp. 270-278, 1998.
- [16] K.R. Arvind, J. Kumar and A.G. Ramakrishnan, "Line Removal and Restoration of Handwritten Strokes," *International Conference on Computational Intelligence and Multimedia Applications*, vol. 3, pp. 208-214, 2007.
- [17] J.D. Foley, A.V. Dam, S.K. Feiner, J.F. Hughes, *Computer Graphics: Principles and Practice in C*, 2nd Edition, Addison-Wesley, Pearson Education.
- [18] <http://www.abbyy.com>.