An Approach to Extracting the Target Text Line from a Document Image Captured by a Pen Scanner [*]

Zhen-Long BAI and Qiang HUO
Department of Computer Science and Information Systems,
The University of Hong Kong, Pokfulam Road, Hong Kong, China
(Email: zlbai@csis.hku.hk, qhuo@csis.hku.hk)

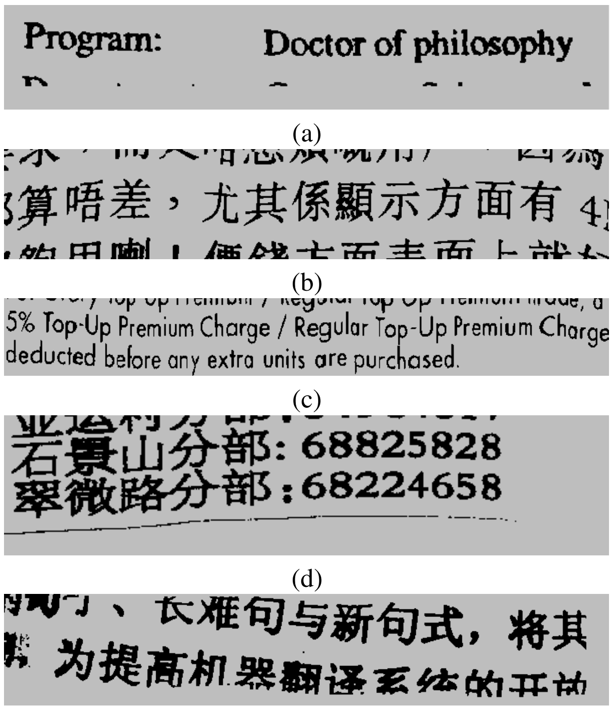
Abstract

In this paper, we present a new approach to extracting the target text line from a document image captured by a pen scanner. Given the binary image, a set of possible text lines are first formed by nearest-neighbor grouping of connected components (CC). They are then refined by text line merging and adding the missed CCs. The possible target text line is identified by using a geometric feature based score function and fed to an OCR engine for character recognition. If the recognition result is confident enough, the target text line is accepted. Otherwise, all the remaining text lines are fed to the OCR engine to verify whether an alternative target text line exists or the whole image should be rejected. The effectiveness of the above approach is confirmed by experiments on a testing database consisting of 117 document images captured by C-Pen and ScanEye pen scanners.

1. Introduction

In this paper, we study the problem of how to extract a target text line from a document image captured by a pen scanner used in products such as C-Pen [3], ScanEye [4], etc. The extracted text line will be fed to a Chinese/English OCR engine developed in our lab [6, 2, 5] for character recognition. Depending on the nature of the captured image, we define the target text line as follows:

- For image examples as shown in Figs. 1(a)&(b), there is only one line with complete characters that is naturally defined to be the target line;
- For the image example as shown in Fig. 1(c), there are several lines with complete characters, among which one is in the middle of the image, thus can be defined as the target line;



Program: Doctor of philosophy

算唔差，尤其係顯示方面有 4

5% Top-Up Premium Charge / Regular Top-Up Premium Charge
deducted before any extra units are purchased.

石景山分部: 68825828
翠微路分部:68224658

、长难句与新句式，将其
为提高机器翻译系统的

Figure 1. Examples of Pen Scanner Images

- For the image example as shown in Fig. 1(d), there are also several lines with complete characters, but none of them is clearly in the middle of the image. It is difficult even for human to judge which line is the target line. Therefore, we let the OCR result speaks, namely, the line with the highest recognition confidence will be treated as the target line.

We wish our text line finder is able to extract the complete target line from the above type of document images, and to reject any image that does not contain a text line with complete characters as shown in Fig. 1(e), or the one

[*]This work was supported by grants from the CRCG of HKU. Contribution made by Zhi-Dan FENG on this work is gratefully acknowledged.

that can not be recognized reliably by our OCR engine. Although there exist many good techniques for extracting text lines from a printed document page scanned by a traditional flatbed scanner (e.g., [11, 9, 1, 10, 7, 8]), it seems none of them can fully achieve the goal for our specific application here, mainly because of the following unique characteristics a pen scanner image might have:

- Due to the small scanning window of the pen scanner, a pen scanner image includes only a very limited number of text lines, with a total number of characters much smaller than that was expected in a traditional document page;
- The scanned character line might be skewed, undulated, and even curved;
- Groups of characters might be separated by a big space (e.g., Fig. 1(a)), yet they still need be treated as belonging to the same line.

We thus developed a new text line extraction approach for our specific application. It is noted that although the overall strategy of our approach is new, we do borrow many component techniques developed by others. In the following, we describe in detail how our approach works.

2. Our Approach

2.1. Preprocessing

Given the binary document image captured by a pen scanner, salt and pepper noises are first reduced by filtering the image with an approach similar to the one in [10]. Then, connected components (CC) are found. The CCs on the left and right boundaries are typically the fragmented characters, thus are removed *permanently*. The patches of black pixels touched to the boundaries are identified by using a run-length smoothing algorithm (RLSA) [11] and are also removed *permanently*. In the remaining CCs, we identify a subset of *special CCs* that might include large noise speckles, underlines, border lines of tables and forms, graphics, images, etc. We used CC's size and CC's black pixel density to make the decision. Apparently, some useful CCs might be labeled wrongly as the *special CCs*. So, the *special CCs* are removed *tentatively* at this stage, and will be re-examined in a later processing step when more information becomes available to make a better decision.

2.2. Forming Possible Text Lines

After the above preprocessing step, we sort all the remaining CCs from left to right based on the x-positions of the centers of their bounding boxes. We then use the following procedure to form the possible text lines:

Step 1: Initially, mark all the CCs as unassigned.

Step 2: Starting from an unassigned CC,

- Form a path of unassigned CCs by grouping recursively its right nearest-neighbor (NN) CC till
  - the right-side boundary is reached, or
  - the *cost of grouping the CC pair* is larger than a threshold C_T, or
  - the angle, ϕ, between the bounding box centers of the two grouping CCs is larger than a threshold ϕ_T;
- If the number of CCs in the above CC path is larger than a threshold N_T, then accept the path and mark all the CCs in the path as assigned;

Step 3: Repeat **Step 2** until all the CCs are processed. Consequently, a set of initial CC paths are formed.

Step 4: For each pair of neighboring CC paths along vertical direction,

- Merge them to form a new path, if
  - the *overlap degree*, OD, is larger than a threshold OD_T, and
  - the *smallest vertical distance*, DY_{min}, is less than a threshold DY_T;
- Skip, otherwise.

Step 5: For each unassigned CC,

- Assign it to the CC path to which its nearest-neighbor "assigned CC" belongs, if
  - their grouping cost is less than a threshold CA_T;

Step 6: For each *special CC* removed tentatively in the preprocessing step,

- Remove it permanently, if
  - its width is larger than a threshold, and its height is smaller than a fraction of the height of its neighboring CC path, or
  - its height is significantly larger than the height of its neighboring CC path, or
  - its width is larger than a threshold, and its black pixel density is significantly smaller than the average black pixel density of its neighboring CC path;
- Add it to the CC path to which its nearest-neighbor "assigned CC" belongs, otherwise.

Step 7: Treat each CC path as a possible text line.

In the above procedure, we define the *cost function for grouping a pair of CCs*, CC_i and CC_j, as follows:

$$C_{ij} = \alpha_x \cdot d_x + \alpha_y \cdot d_y + \alpha_h \cdot d_h \quad (1)$$

where d_x, d_y are the horizontal and vertical distances respectively between the bounding box centers of two CCs; d_h is the absolute value of the two CC's height difference; α_x, α_y, and α_h are three weighting coefficients. For a given CC, its nearest-neighbor CC is the one with the smallest cost function between them. The angle between two CCs is defined as $\phi = \arctan(\frac{d_y}{d_x})$.

For a CC path, CCP_i, consisting of n_i CCs, we sort the CCs in descending order according to their heights. We define CCP_i's *height*, H_i, as the height of kth CC, where $k = 25\% n_i$. Furthermore, we use y_{ij} to denote the vertical position of the bounding box center of jth CC. We define CCP_i's *vertical center position* as

$$Y_i = \frac{1}{n_i} \sum_{j=1}^{n_i} y_{ij} \ . \quad (2)$$

For a pair of CC paths, CCP_i and CCP_j, we can then define their *overlap degree*, OD_{ij} as

$$OD_{ij} = \frac{\frac{1}{2}(H_i + H_j) - \min(|Y_i - Y_j|, \frac{1}{2}(H_i + H_j))}{\frac{1}{2}(H_i + H_j) + |Y_i - Y_j|} \ . \quad (3)$$

Furthermore, for each CC in CCP_i, if there is a horizontally overlapped CC in CCP_j, we calculate and record their vertical distance d_y. Among all such recorded distances d_y's, suppose DY_{min} is the smallest value. We define the *smallest vertical distance* between CCP_i and CCP_j as DY_{min}.

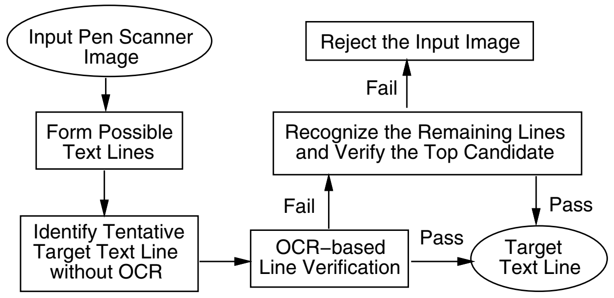
2.3. Extracting the Target Text Line or Rejecting the Image

After the above processing steps, a set of M possible text lines (labeled top down by line numbers $1, 2, \cdots, M$) are obtained from the given binary image. For ith text line, we define, in the following, a score function based on some geometric features to measure the possibility of the line being a target line as defined in the introduction section:

$$S_i = \beta_1 \cdot n_i + \beta_2 \cdot H_i - \beta_3 \cdot |Y_i - Y_{mid}| - \beta_4 \cdot |i - \frac{1}{2}M| \quad (4)$$

where n_i is the number of CCs in the text line, H_i and Y_i are the height and the vertical center position of the text line respectively, and Y_{mid} is the value of the vertical center of the whole pen scanner image. Again, $\beta_1, \beta_2, \beta_3, \beta_4$ are weighting coefficients that need be set appropriately. We found the following setting works well in practice:

$$\beta_1 = [\frac{1}{M} \sum_{i=1}^{M} n_i]^{-1} \ ,$$



Input Pen Scanner Image
Form Possible Text Lines
Identify Tentative Target Text Line without OCR
OCR–based Line Verification
Pass
Target Text Line
Fail
Recognize the Remaining Lines and Verify the Top Candidate
Pass
Fail
Reject the Input Image

Figure 2. The Flow Chart of our Target Text Line Extraction Approach.

$$\beta_2 = [\frac{1}{M} \sum_{i=1}^{M} H_i]^{-1} \ ,$$

$$\beta_3 = [\frac{1}{M} \sum_{i=1}^{M} |Y_i - Y_{mid}|]^{-1} \ ,$$

$$\beta_4 = [\frac{1}{M} \sum_{i=1}^{M} |i - \frac{1}{2}M|]^{-1} \ .$$

The tentative target text line, l_t, is identified as follows:

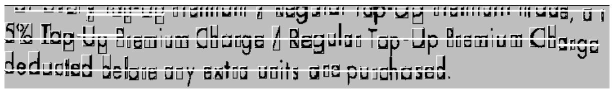
$$l_t = \arg\max_i S_i \ .$$

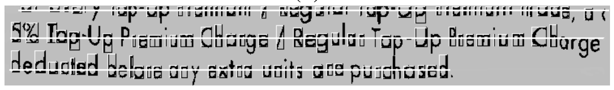
The text line l_t is then fed to our Chinese/English OCR engine for character recognition [6, 2, 5]. As described in [2], during the recognition process, our OCR engine also evaluates a confidence measure for each recognized character to show how reliable the recognition result is. From these character-level confidence measures, we can also define the text line level confidence measure to judge how reliable the line is recognized. One simple confidence measure for text line i is as follows:

$$CM_i = \frac{1}{N_i} \sum_{k=1}^{N_i} CM_i(k) \ , \quad (5)$$

where N_i is the number of characters recognized in the text line, $CM_i(k)$ is the character-level confidence measure for kth character.

For the above tentative target text line, l_t, if its line-level confidence measure CM_{l_t} is larger than a threshold CM_T, we can confirm that the line is the target line we are seeking and accept the recognition result. Otherwise, all the remaining $M - 1$ text lines will also be fed to the OCR engine for recognition. We choose the line with the highest line-level confidence measure as a new tentative target text line. If its confidence measure is larger than the threshold CM_T, the line is confirmed as the target line. Otherwise, we reject the whole pen scanner image. Fig. 2 shows the flow chart of our target text line extraction approach.


(a)


(b)

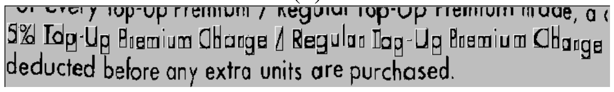
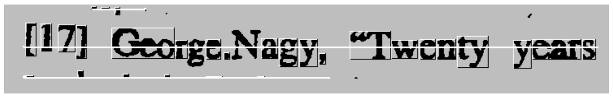
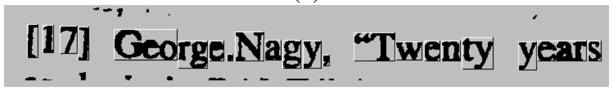
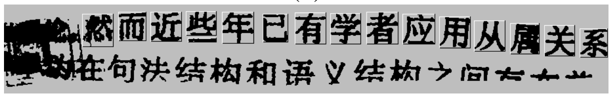

(c)

Figure 3. Illustrative Examples of Line Finding Results: (a) The possible CC paths; (b) Text lines after the merging of CC paths; (c) The identified target text line.


(a)


(b)


(c)

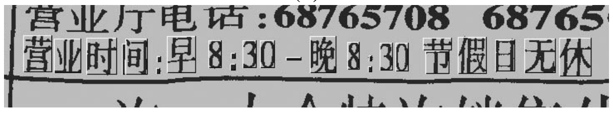

(d)

Figure 4. Examples Illustrating the Importance of Steps 5&6 in Line-Finding Procedure.

3. Experiments and Results

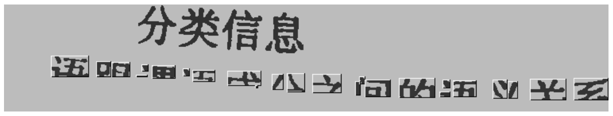
3.1. Some Illustrative Examples

To show how our approach works, in the following, we use some examples to illustrate certain points.

For the pen scanner image example in Fig. 1(c), after Steps 1-3 of the procedure described in Section 2.2, 6 CC paths can be formed as illustrated in Fig. 3(a), where each CC path is highlighted by overlaying a white line at the CC path's vertical center position. After the merging of CC paths in Step 4, 3 possible text lines are formed as illustrated in Fig. 3(b). For this specific example, Steps 5-6 do not affect the line-forming result. By using the score function in Eq. (4), the middle line is identified as a tentative target line as illustrated in Fig. 3(c). It is then fed to the OCR engine for recognition. The verification test of the line-level confidence measure is passed, thus the line is confirmed as the target text line.

Fig. 4(a) gives another example of the line-finding result after Steps 1-4. It is noticed that the CC representing character "N" is not assigned during the previous processing steps. After Step 5, this CC is added into the middle path as illustrated in Fig. 4(b), thus the complete text line is formed. Two more examples are illustrated in Figs. 4(c)(d) to show the impact of the Step 6. Here, the special CCs are confirmed thus removed successfully. Again, all the lines highlighted in Figs. 4(b)(c)(d) are identified successfully as the target lines by using the score function in Eq. (4).

Fig. 5 gives an example to show the importance of using the line-level OCR confidence measure for target line identification. Although the line of fragmented characters illustrated in Fig. 5(a) is identified as a tentative target line by using the score function in Eq. (4), it is rejected by verifying


(a)

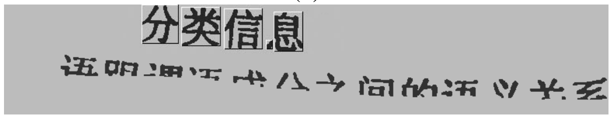

(b)

Figure 5. Examples Illustrating the Importance of Using the Line-Level OCR Confidence Measure for Target Line Identification.

the line-level confidence measure. Consequently, the alternative line is recognized and accepted after the verification of the line-level confidence. In this way, we can improve the usability of this kind of products. As for the image example illustrated in Fig. 1(e), it can be successfully rejected by using the line-level OCR confidence measure.

3.2. Benchmark Test

In order to verify the efficacy of our approach for target line extraction, we performed a benchmark test. To form a testing set, we collected 60 and 57 document images by using C-Pen 10 [3] and ScanEye [4] pen scanners respectively. All images are scanned from printed English/Chinese books, journals, newspapers, etc. To use the

Table 1. A Summary of Benchmark Testing Results for Line Segmentation and Target Line Extraction.

Number of Pen Scanner Images	Total Number of Lines	Correctly Formed Lines	Wrongly Deleted Lines	Wrongly Split Lines	Wrongly Merged Lines	Correctly Segmented Images	Number of Correctly Identified Target Lines	
							without OCR	with OCR
59 (C-Pen)	113	112	0	1	0	58	56	58
57 (ScanEye)	121	120	0	1	0	56	56	56

pen scanner, one just grips it as holding a traditional highlighter and glides it *naturally* over a line of text. As a result, according to the criteria defined in introduction section, there is only one C-Pen image need be rejected, namely the one illustrated in Fig. 1(e). There are 3 ScanEye images belong to the category as illustrated in Fig. 1(d). For these images, if the line-finder can identify any line of complete characters, we treat it as a correctly identified target line. For the remaining images, there are no ambiguity for human to define the target line thus the ground truth can be established accordingly.

Given the above testing data, it is impossible to verify the garbage rejection capability of our target line extraction algorithm. We leave this for future study after we collect deliberately more garbage samples. At this stage, we only use 59 C-Pen images and 57 ScanEye images that contain target lines to test the accuracy of the target line identification. The verification threshold CM_T has been set as $\frac{2}{3}$. With this value, we did not observe any false rejections. We summarize in Table 1 the benchmark testing results for target line identification. There are 1 C-Pen image and 1 ScanEye image that are wrongly segmented, both due to the wrong splitting of a target line. After using the OCR-based line verification, target lines from 2 more C-Pen images are identified correctly.

4. Discussions and Conclusion

In this paper, we have described an innovative approach to extracting the target text line from a document image captured by a pen scanner. Its effectiveness has been confirmed in a preliminary benchmark testing. It is noted that in our approach, there are several control parameters need be set appropriately. All of the above results are obtained by using a fixed set of values for these control parameters. Due to the space limitation, we are not able to elaborate on how we choose and set these control parameters here, although we do come out these values via some experiments and fine-tuning. As future works, we plan to collect more data to perform a larger scale benchmark test. With more data, we can also perform a more meaningful study of the sensitivity of the control parameters on the performance of line identification and garbage image rejection. In this way, we can hopefully identify the possible remaining weak points in our current approach for further improvement.

References

[1] H. S. Baird, S. E. Jones, and S. J. Fortune, “Image segmentation using shape-directed covers,” in *Proc. 10th ICPR*, 1990, pp.820-825.

[2] Z.-D. Feng and Q. Huo, “Confidence guided progressive search and fast match techniques for high performance Chinese/English OCR,” in *Proc. ICPR-2002*, August 2002, pp.III-89-92.

[3] http://www.cpen.com/

[4] http://www.penpower.com.tw/

[5] Q. Huo and Z.-D. Feng, “Improving Chinese/English OCR performance by using MCE-based character-pair modeling and negative training,” in *Proc. ICDAR-2003*, August 2003.

[6] Q. Huo, Y. Ge, and Z.-D. Feng, “High performance Chinese OCR based on Gabor features, discriminative feature extraction and model training,” in *Proc. ICASSP-2001*, May 2001, pp.III-1517-1520.

[7] K. Kise, A. Sato and M. Iwata, “Segmentation of page images using the area Voronoi diagram,” *Computer Vision and Image Understanding*, Vol.70, No.3, pp.370-382, 1998.

[8] J.-S. Liang, I. T. Phillips, and R. M. Haralick, “A statistically based, highly accurate text line segmentation method”, in *Proc. 5th ICDAR*, 1999, pp.551-554.

[9] G. Nagy and S. Seth, “Hierarchical representation of optically scanned documents,” in *Proc. 7th ICPR*, 1984, pp.347-349.

[10] L. O’Gorman, “The document spectrum for page layout analysis,” *IEEE Trans. on PAMI*, Vol.15, No.11, pp.1162-1173, 1993.

[11] K. Y. Wong, R. G. Casey, and F. M. Wahl, “Document analysis system”, *IBM J. Res. Develop.*, Vol.26, No. 6, pp.647-656, 1982.