Script-free text line segmentation using interline space model for printed document images

Minwoo Kim, Il-Seok Oh
Division of Computer Science and Engineering
Chonbuk National University
Jeonju, Korea
{bghunter, isoh}@chonbuk.ac.kr

***Abstract*—This paper proposes a model-based text line segmentation algorithm for machine-printed document images. The model is based on geometric configuration which uses the interline spaces rather than the text lines. The paper proposes an objective function whose maximization leads to the optimal solution. The proposed interline space model provides the primary advantage of script-free nature. Additionally the model is versatile due to its abilities of processing both horizontally and vertically written documents and inferring the semantic of reading order. The experiments performed with various document images in Latin, Korean, Chinese, and Japanese scripts have proven the aforementioned advantages and have shown the noise tolerance.**

Keywords-component; text line segmentation; interline space; geometric matching; reading order; model-based approach

I. INTRODUCTION

In the DAR (document analysis and recognition) fields, comparably more studies have been done for documents written in Latin script than other scripts [1, 2]. It is partly because Western languages are based on Latin script and their alphabets have rather uniform shape and alignment. On the contrary, non-Latin scripts have totally different roots from each other. For example, Oriental languages such as Chinese, Korean, Japanese, and Indian have their own scripts and bear little similarity. It is not unusual that non-Latin scripts solve DAR problems by borrowing conventional methods developed for Latin script. In the cases that the problem is script-independent, the adopted method may work well for non-Latin scripts too. However some problems are of script-dependent nature. The text line segmentation is one of those problems because the strokes have different shapes and relationships, and they are distributed in a text line differently from script to script [3].

Most of researches on the layout analysis of machine-printed documents use the approach of first detecting text blocks and then segmenting them into text lines rather than detecting text lines directly from input page images [4]. Many people believe that once the page image has been decomposed into text blocks, text lines can be easily segmented by a simple projection analysis. However this approach works well only for constrained situations. In an open environment, it is defeated by diverse types of frame noises and skews. Thus actual systems have additional processes such as for noise removal and skew correction [5, 6]. However it is a very difficult job to develop a system that every stage of a series in processes accomplishes its own mission successfully.

Breuel proposed the constrained text line finding algorithm that attempts to detect the text lines directly from page images [7]. The algorithm is based on the fact that alphabets of Latin script have one or two connected components aligned consistently on baseline and descender line. Fig. 1(a) shows the configuration. The approach searches a set of optimal solutions (text lines) using a least-square matching. The whole procedure is governed by a branch-and-bound searching algorithm. Since this one-step process generates a set of accurate text lines in an explicit form, it has many advantages that extra process of removing graphics is not needed and skew angle is easily estimated. A rigorous comparison of six well-known algorithms has been performed in [8]. The experimental results showed that the Breuel's method has the state-of-the-art performance. The method has been proven to be superior to other methods in processing heterogeneous documents, like ones containing various font sizes. However the model used is limited to Latin script because it is formulated in terms of baseline and descender line as shown in Fig. 1(a).

This paper proposes a model-based text line segmentation algorithm. The model is based on geometric configuration which uses the interline spaces rather than the text lines. The proposed interline space model provides the primary advantages of script-free nature. The model gives another advantage that it can process both horizontally and vertically written documents. Furthermore it is easy job to infer the reading order of documents from the outputs of the proposed model. The experiments performed with various document images in Latin, Korean, Chinese, and Japanese scripts have proven the advantages. Noise tolerance of the proposed model is also analyzed.

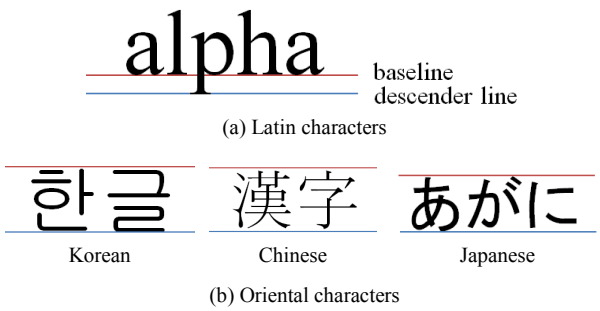
II. INTERLINE SPACE MODEL

A. Motivations

Fig. 1 illustrates character strings of four different scripts. In Latin script of Fig. 1(a), all characters are well aligned along the baseline and descender line. And all the alphabets except i and j have one connected component. This

1520-5363/11 $26.00 © 2011 IEEE
DOI 10.1109/ICDAR.2011.272

observation allows a *intra-line* model which uses distribution information of connected components within a text line.



(a) Latin characters

(b) Oriental characters

Figure 1. Shape characteristics of Latin and Oriental scripts

Most of characters from Oriental languages are composed of several connected components as shown in Fig. 1(b). The positions of connected components vary somewhat arbitrarily between the top and bottom bounding lines. It is clear that the intra-line model cannot process Oriental scripts properly. The interline space provides a powerful information which is useful in segmenting adjacent text lines in document images. Because page images in any script have text lines separated by interline spaces, the *interline space* model would work well for any script. Additionally since the interline model uses explicitly the top and bottom of text lines, it is expected to be more robust to various artifacts than the intra-line model which uses only bottom lines.

B. Formulation

We will describe our interline space model. The model uses the bounding boxes of connected components as primitive objects. Fig. 2 shows an example situation. A bounding box provides two points called as *floor* point and *ceil* point. They are represented by f_i and c_i when they come from i-th bounding box. In the figure, f_i and c_i are depicted by black and empty circles, respectively.

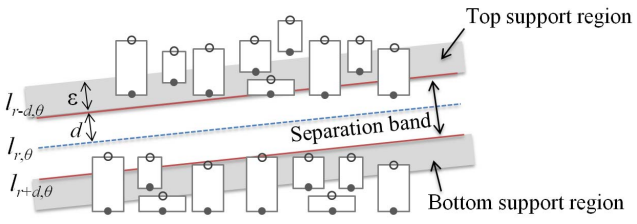


Figure 2. Elements of the proposed interline space model

The origin of coordinate system is located at top and leftmost point of the input page image. The interline space model is defined by three parameters $\Theta=(r, \theta, d)\in R^3$. The first two parameters r and θ represent the center line passing the interline space between two text lines. The parameter d represents half the width of the interline space. The center line is denoted by $l_{r,\theta}$. The top and bottom lines are denoted by $l_{r-d,\theta}$ and $l_{r+d,\theta}$, respectively.

Let us denote a set of floor points by $F=\{f_1, f_2, \ldots, f_n\}$ and a set of ceil points by $C=\{c_1, c_2, \ldots, c_m\}$. Given F and C, the objective function which evaluates a quality of $\Theta=(r, \theta, d)$ is defined by (1).

$$Q(\Theta|F,C)=\sum_{j=1,n} q(f_j, l_{r-d,\theta}) + \sum_{j=1,m} q(c_j, l_{r+d,\theta}) \quad (1)$$

The function $Q(\Theta|F,C)$ evaluates a score related to how well the value of Θ explains or fits the situation represented by F and C. The function $q(p,l)$ identifies whether the point p will *support* or *repel* the line l and evaluates a force of supporting or repelling. The function is defined by (2).

$$q(p,l)=\begin{cases} 0, \ if \ \ p \ is \ neutral. \\ 1-dist(p,l)/\varepsilon, \ if \ \ p \ supports \ l. \\ -dist(p,l)/\varepsilon, \ if \ \ p \ repels \ l. \end{cases} \quad (2)$$

Fig. 2 explains relationship between the point p and line l in the function $q(p,l)$. Let us assume $p\in C$. In the case that p lies within the support region defined by the line l as shown in Fig. 2, p is decided to support l. If p lies within the separation band, it is decided to repel l. The other case is neutral. For the point $p\in F$, the same rule is applied.

Until now, we have described the objective function for Θ which is a *point* in the solution space. However, in the initial stages of the process searching for optimal solutions, Θ is not a point but a *range*. So the objective function is reformulated as follow in order to calculate the upper bound of quality.

$$Q(\tilde{\Theta}|F,C)=\sum_{j=1,n} \max_{(r,\theta,d)\in\tilde{\Theta}} q(f_j, l_{r-d,\theta}) + \sum_{j=1,m} \max_{(r,\theta,d)\in\tilde{\Theta}} q(c_j, l_{r+d,\theta}) \quad (3)$$

Also it has been assumed that the sets F and C were placed around the support regions. However in the initial stage, F and C contain all of floor and ceil points coming from several text lines. The searching algorithm presented in Section 3 will handle the situation by refining step by step the sets F and C.

III. Searching Algorithm

The Algorithm "page segmentation" illustrates whole procedure of obtaining the optimal solution. This searching algorithm is the same as [7, 9]. The model and objective function are replaced with the ones defined in Section 2.B. The searching algorithm uses the data structure, heap as priority queue and works with first element in the heap which is always the best candidate solution. (The value of $Q(\tilde{\Theta}|F,C)$ of a potential solution $\tilde{\Theta}$ is used as priority.) So the algorithm bounds the inferior solutions. The algorithm divides the solution into two (i.e., branch) and inserts them into heap. So the algorithm is said to be branch-and-bound.

For efficiency purpose, the algorithm keeps a set of ceil/floor points associated with a candidate solution. It is named as matchlist and denoted by M in the Algorithm. In the initial stage of algorithm, M is coarse. It is refined step by step.

It starts with a parameter range of the allowed skew angle and the allowed interline space size. The r_{min} and r_{max} are set to be 0 and height of input image, respectively. After inserting the initial solution into the heap, the algorithm initiates the loop in the line 7. The loop terminates when the heap is empty.

Algorithm: page segmentation

Input: a document page image I[1..row][1..col]
Output: S // a set of Θ, each representing an interline space
1. Finds a set of connected components in I[][];
2. Compute sets of floor and ceil points, *F* and *C*;
3. $\widetilde{\Theta} = [(r_{\min}, r_{\max}), (\theta_{\min}, \theta_{\max}), (d_{\min}, d_{\max})]$; //initial range
4. M=[*F*,*C*];
5. heap.push([$\widetilde{\Theta}$,M]) ; // heap is initially empty
6. S= Φ ; // initialize solution set to be empty;
7. while (not heap.empty()) {
8. [$\widetilde{\Theta}$,M]= heap.pop();
9. if (is_point([$\widetilde{\Theta}$,M])) {
10. Transform $\widetilde{\Theta}$ into Θ and insert it into S;
11. Remove points associated with M from all the matchlists in the heap;
12. }
13. else {
14. Bisect $\widetilde{\Theta}$ into $\widetilde{\Theta}_1$ and $\widetilde{\Theta}_2$ and recomputed M_1 and M_2 for them;
15. if ($Q(\widetilde{\Theta}_1 | M_1) > threshold$) heap.push([$\widetilde{\Theta}_1$,$M_1$]);
16. if ($Q(\widetilde{\Theta}_2 | M_2) > threshold$) heap.push([$\widetilde{\Theta}_2$,$M_2$]);
17. }
18. }

In the line 9, the candidate solution is tested whether it is a point or a range. If all of three ranges in $\widetilde{\Theta} = [(r_1, r_2), (\theta_1, \theta_2), (d_1, d_1)]$ are small enough, the solution is decided to be a point and it is taken as a solution. Otherwise (i.e., it is still a range), the solution $\widetilde{\Theta}$ is bisected into two sub-ranges $\widetilde{\Theta}_1$ and $\widetilde{\Theta}_2$. The bisection is done with respect to one of three parameters, (r, θ, d). The one with the largest gap is chosen. The newly spawn solution is tested whether it is viable, i.e., it can ultimately represent an actual interline space. The solution that passed the test is inserted into the heap.

IV. Experimental Results

The experiments used a collection of machine-printed document images in four different scripts. The collection is shown in Table 1. The images were scanned in 300dpi. Since Oriental document images are not available in public databases, we collected them for the experiments.

TABLE I. Set of Document Images

	Korean	**Chinese**	**Japanese**	**Latin**	**Total**
Number of documents	26	20	27	22	95
Number of text lines	1381	1058	1100	1699	5238

Table 2 presents performance data of the intra-line model proposed by Breuel [7] and the interline model proposed in this paper. The evaluation was done using the performance criterion in [10] in which the error rate, *err* is defined as follows. In the formula, *G*, *M*, *O*, and *U* represent a set of text lines in ground truth, text lines missing, over-segmented, and under-segmented, respectively. The symbol |.| represents set size.

$$err = \frac{|M \cup O \cup U|}{|G|} \quad (4)$$

The intra-line model works fairly well for Latin script as expected. However a very low performance has been obtained for other scripts, especially with respect to over-segmentation. It is inevitable that the intra-line model commits this type of error because Oriental scripts generate lots of floor points which are not absorbed by baseline and descender line. The remaining points form extra text lines. Fig. 3 illustrates those ghost text lines.

TABLE II. Performance of Intra-Line and Interline Models

	intra-line model [7]				**Interline model (proposed)**			
	M	O	U	err	M	O	U	err
Latin	0.00	0.24	0.00	0.24	1.35	0.00	0.00	1.35
Korean	0.29	83.06	0.00	83.35	0.00	0.14	0.00	0.14
Chinese	0.66	64.93	0.00	65.60	0.66	0.28	0.00	0.62
Japanese	0.00	68.91	0.00	68.91	0.36	0.45	0.00	0.82

(M: missing, O: over-segmentation, U; under-segmentation, *err*: error rate) (unit: %)

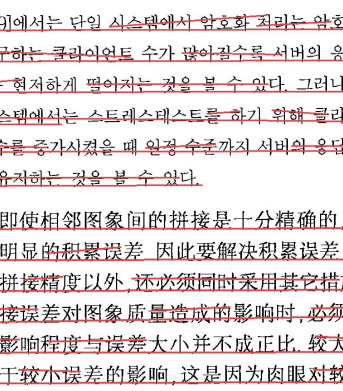


Figure 3. Ghost text lines generated by intra-line model

Table 2 also presents performance data obtained by the interline model proposed in this paper. The model works fairly well for all the scripts. Fig. 4 shows an example of missing errors. During algorithm processing, the current solution encloses the word ‘shape’. The ceil/floor points coming from the word will repel the solution since the points lie within the separation band as Fig. 2 and (2) explain. They contribute negative values to the score. However since the word is short, its negative effect is not enough. Ultimately the current solution is not further bisected, and it is taken as final solution. Since all of the missing errors in the experiment are of this type, the error can be easily overcome by applying a post-processing module.

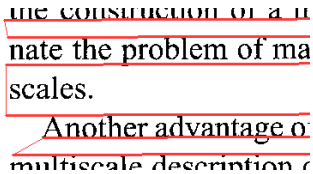
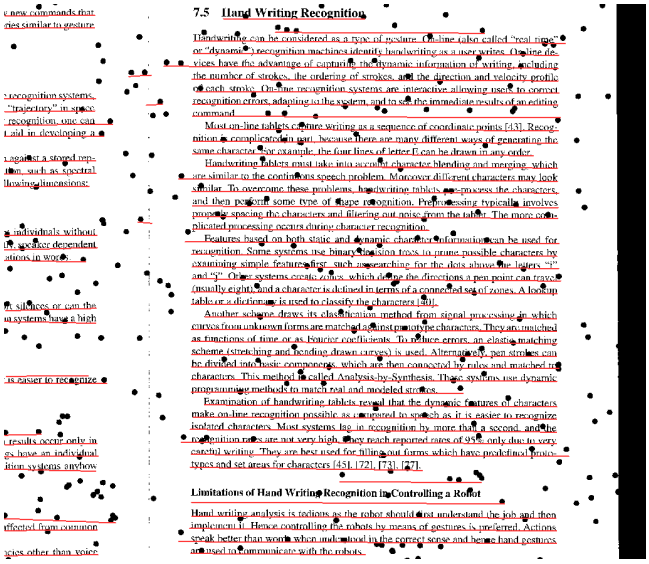
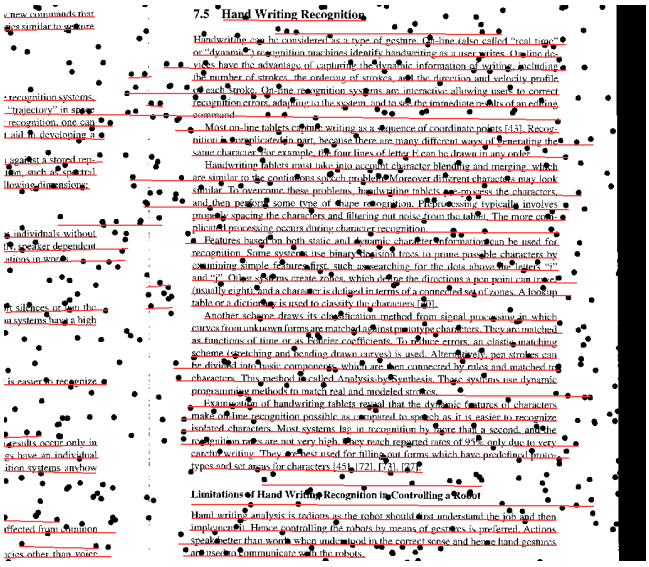


Figure 4. Missing errors generated by interline model

We analyzed the tolerance of the proposed interline model to noise effects. The experiment added artificial noisy bounding boxes by controlling the noise rate. Fig. 5(a) and 5(b) are images which the noise bounding boxes are added by 15% and 30% of original connected components, respectively. Though erroneous text lines appear, the true text lines are still well found. Based on this observation, we say that the proposed model is robust to noises.



(a) 15% noise (about 500 noise boxes added)



(b) 30% noise (about 1000 noise boxes added)

Figure 5. Evaluation of robustness to noises

V. Discussions on Versatility of The Interline Model

This section discusses two abilities of the proposed interline space model. The conventional intra-line model is deficient in these abilities. Some kind of documents is written vertically, such as old Korean newspaper shown in Fig. 6. The proposed interline model can handle the image by modifying the initial parameter range in line 3 of the Algorithm in Section 3. On the contrary, the conventional intra-line model should be modified in the fundamental level since the distribution of bounding boxes of connected components changes severely.

Figure 6. Results by interline model for vertically written document image (old Korean newspaper)

The semantic of reading order of multi-column documents can be readily inferred from the outputs of the interline model. For this purpose, the output is converted into a directed graph that a text line is a. Fig. 7 shows the graph. By checking the matchlist of the nodes, the link between nodes can be made. In the case of nodes b, c, and e, the b's matchlist overlaps with both c's and e's matchlists. So b has two children of c and e. In this manner, DAG (directed acyclic graph) can be constructed. Using DAG, semantic of reading order can be readily inferred.

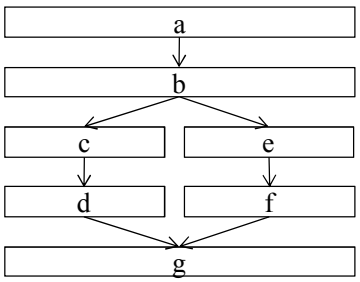


Figure 7. Inferring semantic of reading order from interline model's outputs

VI. CONCLUSIONS

This paper presented a new interline space model which extends the conventional intra-line model. The outputs of the interline model can be converted into one of the intra-line model, but the inverse is not the case. This fact gives the interline model abilities which the intra-line model is deficient. Since the proposed interline model is script-free, it is effective for multi-lingual documents and for documents in any kind of scripts. Additionally the model is versatile due to its ability of processing both horizontally and vertically written documents. Furthermore it is easy job to infer the reading order of documents from the outputs of the proposed model. One of future works is to extend the model to be nonlinear. Use of SVM classifier is an approach which is being currently studied.

ACKNOWLEDGMENT

This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2011-0004600).

This work was supported by the second stage of Brain Korea 21 Project in 2011.

REFERENCES

[1] G. Nagy, "Twenty years of document image analysis in PAMI," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 22, No. 1, January 2000.

[2] R. Kasturi, L. O'Gorman, and V. Govindaraju, "Document image analysis: A primer," Sadhana, Vol. 27, No. 1, pp. 3–22, 2002.

[3] D. Ghosh, T. Dube, and A. P. Shivaprasad, "Script recognition—A review," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 32, No. 12, pp. 2142-2161, December 2010.

[4] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. "Geometric layout analysis techniques for document image understanding: A review," Technical Report, IRST, 1998.

[5] J. van Beusekom, F. Shafait, and T. M. Breuel, "Combined orientation and skew detection using geometric text-line modeling," International Journal on Document Analysis and Recognition, Vol. 13, No. 2, pp. 79-92, June 2010.

[6] Z. Liu, H. Zhou, and N. Yang, "Semi-supervised learning for text-line detection," Pattern Recognition Letters, Vol. 31, No. 11, pp. 1260-1273, August 2010.

[7] T. M. Breuel, "Two geometric algorithms for layout analysis," In Workshop on Document Analysis Systems, pp.188-199, August 2002.

[8] F. Shafait, D. Keysers, and T. M. Breuel, "Performance evaluation and benchmarking six-page segmentation algorithms," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 30, No. 6, June 2008.

[9] T. M. Breuel, "Implementation techniques for geometric branch-and-bound matching methods," Computer Vision and Image Understanding, 2003.

[10] S. Mao and T. Kanungo, "Empirical performance evaluation methodology and its application to page segmentation algorithms," IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 23, No. 3, pp. 242-256, March 2001.