Text-Line Extraction as Selection of Paths in the Neighbor Graph

Koichi Kise[1], Motoi Iwata[1], Andreas Dengel[2], and Keinosuke Matsumoto[1]

[1] Department of Computer and Systems Sciences
College of Engineering, Osaka Prefecture University
1-1 Gakuencho, Sakai, Osaka 599-8531, Japan
{kise,iwata,matsu}@ss.cs.osakafu-u.ac.jp
[2] German Research Center for Artificial Intelligence (DFKI, GmbH)
P.O. Box 2080, 67608 Kaiserslautern, Germany
Andreas.Dengel@dfki.de

Abstract. This paper presents a new method of text-line extraction which can be applied to tilted non-rectangular pages. The method is characterized as follows. As the representation of physical structure of a page, we propose the neighbor graph which represents neighbors of connected components. The use of the area Voronoi diagram enables us to extract neighbors without predetermined parameters. Based on the neighbor graph, the task of text-line extraction is considered to be the selection of its paths appropriate as text-lines. We apply simple iterative selection of edges with local examination so as to reduce the computational cost. From experimental results for 50 pages with rectangular and non-rectangular layout, we discuss advantages and limitations of our method.

1 Introduction

Layout analysis is the process of extracting document components such as text-blocks, text-lines, figures and tables as well as identifying layout structure among them. Since the layout analysis is a complex task, it is often decomposed into several sub-tasks. Text-line extraction is an important sub-task, since a large number of documents mainly consist of text-lines.

Existing methods of text-line extraction can be classified from the viewpoint of a class of layout to be processed. An important and the most investigated class would be *rectangular* layout. The layout is rectangular if all document components can be circumscribed by non-overlapping upright rectangles. For this class of layout, various methods such as smearing and projection have been proposed. Some of the methods attain higher accuracy as well as efficiency by fitting algorithms to this class.

In recent years, however, the number of pages beyond this class has been growing as publishing techniques progress. Pages in recent magazines and journals, etc. sometimes include tilted text-lines so as to make them stand out from

S.-W. Lee and Y. Nakano (Eds.): DAS'98, LNCS 1655, pp. 225–240, 1999.

horizontal ones. In order to extract text-lines from such pages, it is necessary to utilize methods independent of layout as well as skew of text-lines.

Attempts have therefore been made to reduce the assumptions on layout of pages. For black-on-white pages, connected component analysis is often applied to obtain primitives (connected components), and they are merged to form text-lines. Existing methods with connected component analysis are characterized by (1) a representation of physical structure among connected components, and (2) a strategy for merging connected components based on the representation. Some representative methods are as follows: Hough transform with the ρ–θ space[1], distance/angle filtering with k-NN (nearest neighbors)[2], relaxation with low resolution images[3], simulated annealing with pairs of adjacent connected components [4].

In this paper, we propose a new approach for text-line extraction with the help of the *area Voronoi diagram*[5] developed in the field of computational geometry. The area Voronoi diagram enables us to obtain neighbor relations between connected components efficiently. In our method, the neighbor relations are represented by a graph called the *neighbor graph* whose vertices and edges correspond to connected components and neighbor relations, respectively. Based on the neighbor graph, the task of text-line extraction is considered to be the selection of its *paths* appropriate as text-lines. In order to keep the computational cost low, the method selects paths by iterative extension of edges based only on local examination. From experimental results for 50 images with rectangular and non-rectangular layout, we discuss advantages and limitations of the method.

2 Representation of Physical Structure

2.1 Area Voronoi Diagram

The physical structure of a binary document image can be represented by *neighbor* relations among connected components. In this paper, we employ the neighbors of connected components defined by the *area Voronoi diagram*.

The area Voronoi diagram is a generalization of the ordinary (point) Voronoi diagram[5]. While the point Voronoi diagram is generated from a set of *points*, the area Voronoi diagram is generated from a set of *non-overlapping figures*.

The definition of the area Voronoi Diagram is as follows. Let $G = \{g_1, ..., g_n\}$ be a set of non-overlapping figures in the 2-dimensional plane, and $d(p, g_i)$ be the Euclidean distance between a point p and a figure g_i defined by

$$d(p, g_i) = \min_{q \in g_i} d(p, q) \ , \tag{1}$$

where q is a point on the contour of g_i and $d(p, q)$ is the Euclidean distance between p and q. Then a Voronoi region $V(g_i)$ and the area Voronoi diagram $V(G)$ are defined by

$$V(g_i) = \{p | d(p, g_i) \leq d(p, g_j), \forall j \neq i\} \ , \tag{2}$$

$$V(G) = \{V(g_1), ..., V(g_n)\} \ . \tag{3}$$

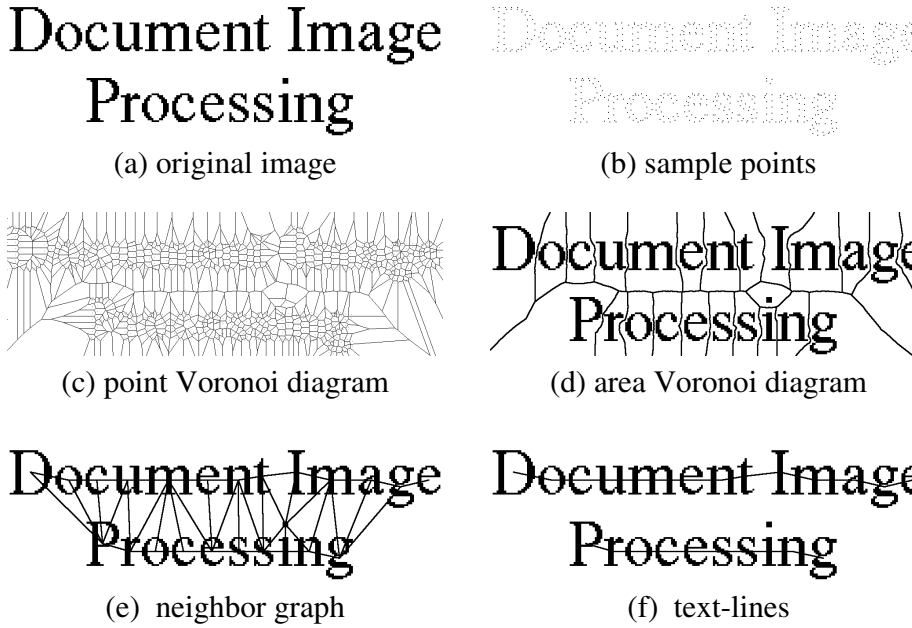


Fig. 1. The Area Voronoi diagram and the neighbor graph.

The boundaries of Voronoi regions are called *Voronoi edges*.

It is known that the area Voronoi diagram can be approximately constructed from the point Voronoi diagram in the following manner[5]:

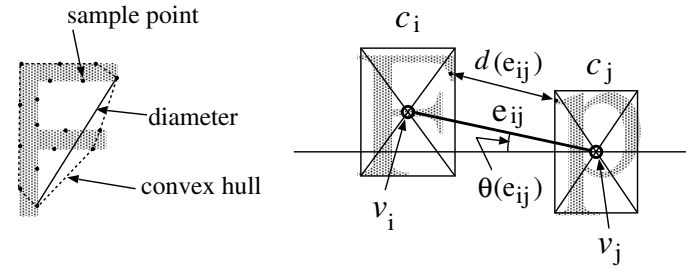
step 1 Generate the point Voronoi diagram from the points $P = P_1 \cup ... \cup P_n$ where P_i be a set of points lying on the contour of g_i.
step 2 Delete Voronoi edges generated from points on the same figure.

Although a Voronoi edge in the area Voronoi diagram may be a complex curve, that of the approximated version consists of line segments, which are the approximation of the curve. In this paper, the approximated version is referred to as the area Voronoi diagram for simplicity.

The above algorithm enables us to construct the area Voronoi diagram directly from a binary image[6]. By applying labeling, contour following and sampling procedures to a binary image, points on contours of connected components are obtained as shown in Fig. 1(b). The point and area Voronoi diagrams are constructed using these points as shown in Fig. 1(c) and (d), respectively.

2.2 Neighbor Graph

The area Voronoi diagram can be interpreted as the representation of neighbor relations among connected components based on the Euclidean distance: a pair



(a) Features of a vertex (b) Features of an edge

Fig. 2. Features.

of connected components which share a Voronoi edge on their Voronoi regions are neighbors with each other.

In our method, the neighbor relations are explicitly represented by a graph called the neighbor graph. The neighbor graph $\langle V, E \rangle$ is a graph in which a vertex $v_i \in V$ corresponds to a connected component c_i, and an edge $e_{ij} \in E$ between vertices v_i and v_j corresponds to a Voronoi edge shared by connected components c_i and c_j. In other words, an edge e_{ij} represents the neighbor relation between connected components c_i and c_j. The neighbor graph for the area Voronoi diagram in Fig. 1(d) is shown in Fig. 1(e).

It is worth noting that, in ordinary cases, a text-line can be represented by a *path* of a neighbor graph as shown in Fig. 1(f). In such cases, therefore, all we have to do to extract text-lines is to select the appropriate paths from the neighbor graph.

2.3 Features

In order to select the appropriate paths, we utilize the features shown in Fig. 2.

Let $P = \{p_1, ..., p_n\}$ be sample points on the contour of a connected component c. The features of a vertex v (a connected component c) are defined using these points as follows:

area $a(v)$ **:** The area $a(v)$ is the area of the convex hull of P.

diameter $D(v)$ **:** The diameter $D(v)$ is the distance between the farthest pair of points in P:

$$D(v) = \max_{p_i, p_j \in P} d(p_i, p_j) \ . \tag{4}$$

The area is utilized, for example, to distinguish characters from larger connected components in figures. The diameter is useful to distinguish characters from thin ruled lines whose areas are close to those of characters.

Features of an edge e_{ij} are illustrated in Fig. 2(b).

distance $d(e_{ij})$ **:** The distance $d(e_{ij})$ is defined by

$$d(e_{ij}) = \min_{p_1 \in P_1, p_2 \in P_2} d(p_1, p_2) \ . \tag{5}$$

where P_1 and P_2 are the sets of sample points on contours of connected components c_1 and c_2, respectively.

angle $\theta(e_{ij})$ **:** Let v_i and v_j be the centroids of the minimum upright bounding boxes of connected components c_i and c_j, respectively. The angle $\theta(e_{ij})$ [degree] is the angle of the line segment $\overline{v_i v_j}$ to the horizontal line.

3 Text-Line Extraction

3.1 Overview

The method of text-line extraction consists of three steps: (1) generation of the neighbor graph, (2) extraction of seeds and (3) extension of seeds. Figure 3 shows results of these steps.

The first step is to generate the neighbor graph from an input binary image. All the features described in Sect. 2.3 are also calculated in this step.

Text-lines are extracted at the succeeding steps based on the assumptions that:

(A1) Every text-line is represented by a path in the neighbor graph, [1] i.e., $(v_1, e_{12}, v_2, ..., e_{m-1,m}, v_m)$ where $v_i \in V$, $e_{ij} \in E$, and $v_i \neq v_j$ if $i \neq j$,

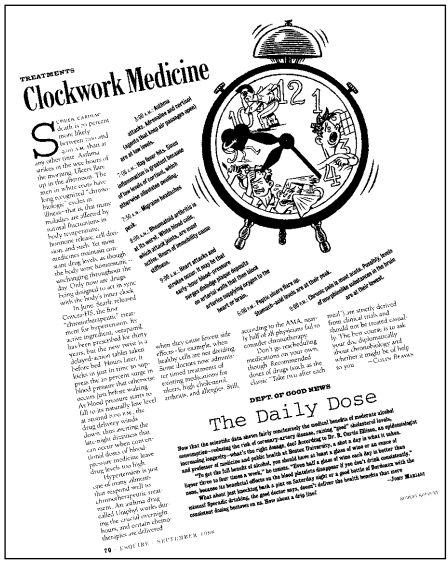
(A2) Every text-line consists of connected components of approximately equal size and shape,

(A3) Every text-line is linear.

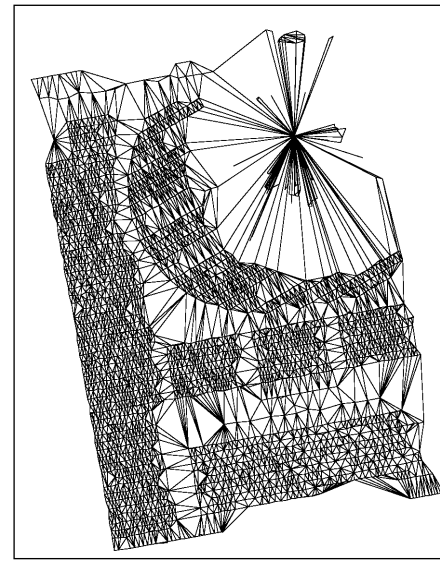
In order to get a clue to extract text-lines, we extract *seeds* from the neighbor graph at the second step. A seed is a path of the neighbor graph which seems a part of a text-line with confidence. Since a seed contains no loops and branches, we can estimate the orientation of a text-line.

At the last step, seeds are extended to obtain complete text-lines based on the estimated orientation. To extend seeds reliably without the help of time-consuming process such as relaxation, we apply simple iterative extension with a varying criterion. Since seeds are short and thus unreliable at early stages of the iteration, we apply a strict criterion. At later stages, on the other hand, a criterion is relaxed to obtain complete text-lines. An edge to be merged with a seed is selected based on local examination of edges which connect with a seed at its end. Details of each step are described below.

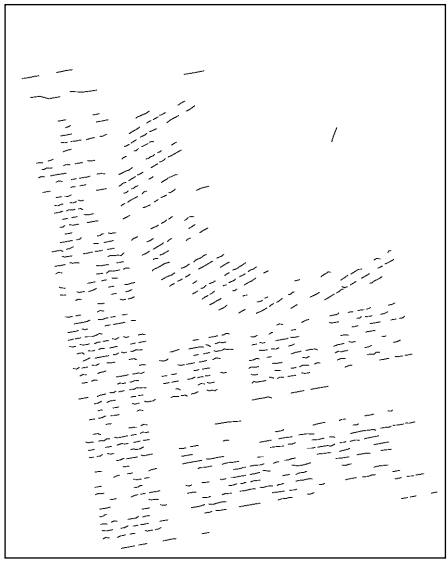
[1] To be precise, a text-line is represented not by a path, but by a connected subgraph of the neighbor graph, since a path excludes small dots of characters (e.g., i, j). In most cases, however, such dots can be easily included in a text-line by, for example, constructing a convex hull of connected components in a path.



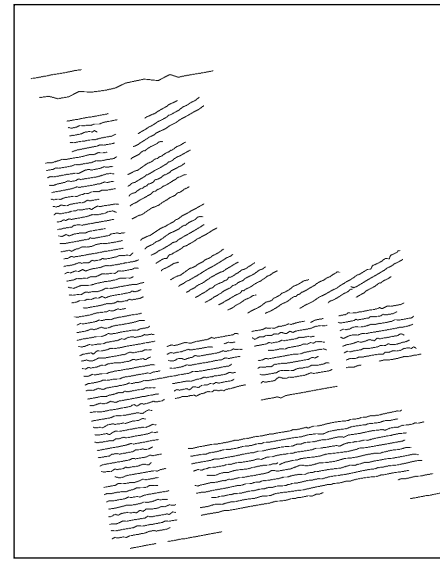
(a) original image



(b) neighbor graph



(c) seeds



(d) extracted text-lines

Fig. 3. Results of each step. The image (a), which is artificially tilted by 10°, contains 5433 connected components. The number of edges in (b), (c) and (d) is 8343 (100%), 1231 (15%) and 2575 (31%), respectively. The total computation time was 6.36 sec. with a Pentium II 300MHz PC.

3.2 Generation of the Neighbor Graph

Pixels on contours of connected components are selected with a sampling rate R. These pixels are utilized as sample points for the construction of the area Voronoi diagram. In this process, a connected component is eliminated as noise if its area is less than or equal to T_n. Next, the area Voronoi diagram is transformed into the neighbor graph. Figure 3(b) illustrates an example of the neighbor graph.

3.3 Extraction of Seeds

This step consists of the following three sub-steps: initial filtering of edges, extraction of seed candidates and identification of seeds.

Initial Filtering of Edges The neighbor graph generally includes superfluous edges which violate the assumption (A2). In this sub-step, such edges are deleted. An edge e_{ij} between vertices v_i and v_j is deleted from the neighbor graph if it satisfies one of the following equations:

$$\frac{\min(a(v_i), a(v_j))}{\max(a(v_i), a(v_j))} \leq T_a \ , \tag{6}$$

$$\frac{\min(D(v_i), D(v_j))}{\max(D(v_i), D(v_j))} \leq T_D \ , \tag{7}$$

where T_a and T_D are thresholds. After the deletion of superfluous edges, isolated vertices are also deleted.

Extraction of Seed Candidates In general, a connected component is closest to that in the same text-line. Thus a group of connected components close with one another can be a candidate of a seed.

Since seeds should be a part or a whole of a text-line, we first select edges whose distances are less than or equal to the estimated inter-line gap T_{ds} of body text regions. Figure 4 illustrates the frequency distribution for the edges in Fig. 3(b). There exist two apparent peaks P_1 and P_2 near the origin: P_1 and P_2 correspond to inter-character and inter-line gaps, respectively. Thus the value of T_{ds} is set to the distance for the peak P_2.

Then the selected edges are connected to form seed candidates. An edge e_{ij} between vertices v_i and v_j is tested in the ascending order of the distance $d(e_{ij})$ as follows:

1. If both v_i and v_j are contained in no seed candidates, make a seed candidate (v_i, e_{ij}, v_j).
2. If only one of the vertices v_i, v_j is contained at an end of a seed candidate, connect the edge e_{ij} to the end of the candidate. For example, when a vertex v_j is contained at an end of a seed candidate $(v_j, e_{jk}, ...)$, update the candidate to $(v_i, e_{ij}, v_j, e_{jk}, ...)$.
3. Otherwise, skip e_{ij}.

The extraction terminates when no edge can be selected from the selected edges.

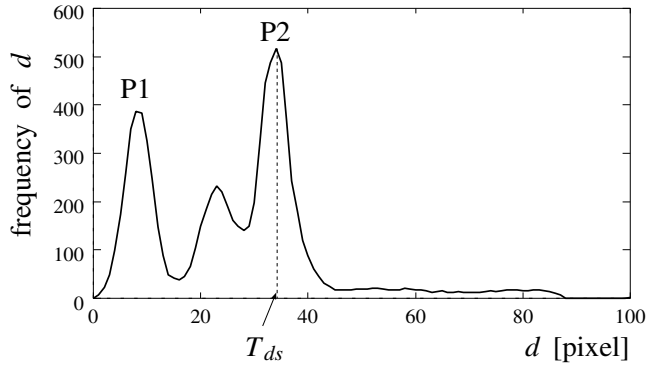


Fig. 4. Frequency distribution of the distance d.

Identification of Seeds Next, seeds are selected from the candidates. A candidate r is selected as a seed if all of the following conditions are satisfied:

1. It consists of more than one edge.
2. $\mathrm{Var}_\theta(r) \leq T_{v\theta}$ and $\mathrm{Var}_d(r) \leq T_{vd}$, where $\mathrm{Var}_\theta(r)$ and $\mathrm{Var}_d(r)$ are the variance of distances and that of angles of edges in r, respectively, and $T_{v\theta}$, T_{vd} are thresholds.

The first condition indicates that a seed candidate consisting of a single edge is less reliable. The second condition represents that connected components in a seed should be arranged approximately linear as well as at approximately even intervals.

After the selection, the following features of a seed s are calculated:

distance $d(s)$ **:** $d(s) = \sum_{e \in E_s} d(e)/|E_s|$, where E_s is a set of edges in a seed.
angle $\theta(s)$ **:** Consider a line segment which connects both ends of a seed s. $\theta(s)$ [degree] is the angle of the line segment to the horizontal line.

3.4 Extension of Seeds

Finally, seeds are extended iteratively using a criterion varying with the current number of times of iteration n. Figure 5 shows the algorithm of this step: at each time of iteration, each seed s is extended as much as possible by selecting an edge connecting at each end of s, and merging the selected edge with s. The function "Select_Edge(s,v_i,n)" selects an edge e_{ij} to be merged with s at a vertex v_i depending on the current number of times of iteration n. This function returns ϕ (no edge) if there exists no appropriate edge. The function "Merge_Edge(s,v_i,e_{ij})" merges the selected edge e_{ij} with the seed s at its end v_i if $e_{ij} \neq \phi$; otherwise, it returns s.

The point of the algorithm is how to select the appropriate edge to be merged with a seed s in the function "Select_Edge". In order to keep the computational cost low, it is desirable to select the edge only by local examination of edges.

```
for n = 1 to N do
  for all seed s ∈ S(the set of seeds) do
    if the seed s exists in S then
      repeat
        v1 ← a vertex at an end of s
        v2 ← a vertex at the other end of s
        s_old ← s
        e1x ← Select_Edge(s,v1,n)
        e2y ← Select_Edge(s,v2,n)
        s ← Merge_Edge(s,v1,e1x)
        s ← Merge_Edge(s,v2,e2y)
        if s_old ≠ s then
          Delete from S all seeds which share edges of s
          Add s in S
          Recalculate features of s
        end if
      until s_old = s
    end if
  end for
end for
```

Fig. 5. Extension of seeds.

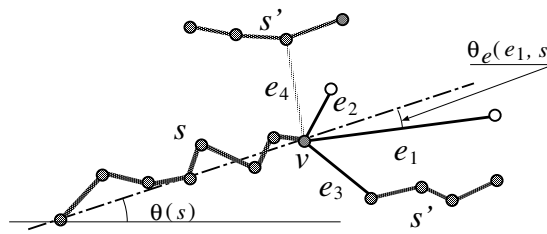


Fig. 6. Selection of K-best edges.

Figure 6 illustrates a seed to be extended. The method selects the first "acceptable" edge from the sorted list of K-best edges $(e_1, ..., e_K)$ as follows.

First, we obtain E_v, i.e., a set of all edges which are contained in no seeds and connect with a seed s only at its end v. If an edge $e \in E_v$ is between the seed s and a different seed s', e must also be at an end of s'. In the case of Fig. 6, $E_v = \{e_1, e_2, e_3\}$. Then, the list $(e_1, ..., e_K)$ is generated from the elements of E_v by sorting them in the ascending order of the difference of angles

$$\theta_e(e_i, s) = |\theta(e_i) - \theta(s)| \ . \tag{8}$$

The number of elements K is less than or equal to $K_{\max}$. In Fig. 6, the list (e_1, e_2) is obtained with $K_{\max} = 2$.

Next, we select the acceptable edge from the list. If an edge is between connected components in a word, the distance of the edge is generally small but

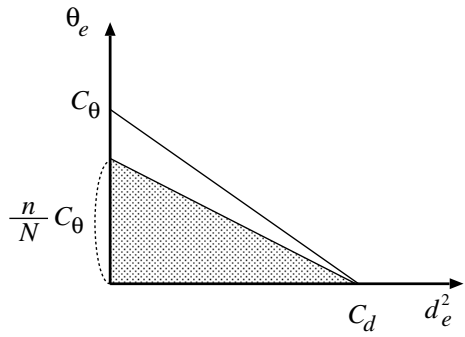


Fig. 7. d_e^2–θ_e plane.

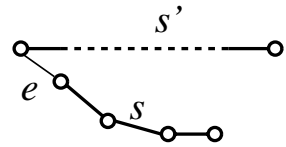


Fig. 8. Necessity of a test with a different seed s'. The edge e is acceptable from s but not from s'.

its angle may not close to that of the seed due to the presence of ascenders and descenders. On the other hand, if an edge is between words, the distance is larger but the angle is more closer to that of the seed. The criterion for the acceptable edges encodes the above nature by the following equations:

$$J(e,s,n) = \frac{\theta_e(e,s)}{\frac{n}{N}C_\theta} + \frac{d_e^2(e,s)}{C_d} \leq 1 \ , \tag{9}$$

$$d_e^2(e,s) = (d(s) - d(e))^2 \ , \tag{10}$$

where N is the total number of times of iteration, C_θ and C_d are thresholds. Figure 7 illustrates the d_e^2–θ_e plane in which an edge corresponds to a point; an edge e satisfies (9) if it is in the shaded area. An edge e for a seed s is tested whether (9) is satisfied if e connects only with the seed s. In the case that an edge e is between s and s', $J(e,s',n) \leq 1$ should also be true in order to keep the extended seed linear. Figure 8 shows the situation in which this additional test is required.

After the iteration, the method outputs as text-lines the seeds whose numbers of edges are more than or equal to M.

Table 1. Values of parameters.

step	neighbor graph		extraction of seeds				extension of seeds				
parameter	R	T_n	T_a	T_D	$T_{v\theta}$	T_{vd}	N	$K_{\max}$	M	C_d	C_θ
value	1/7	64	1/40	1/10	400	50	10	2	3	1600	50

Table 2. Experimental results.

data set	no. of text-lines	correct	error		
			frag.	over-merg.	omission
UW1	1111	89.1%	7.84%	0.63%	2.43%
Nonrect	2975	89.9%	4.00%	4.73%	1.37%
Total	4086	89.7%	5.03%	3.61%	1.66%

4 Experimental Results

4.1 Conditions

Experiments were made using a PC with Pentium II 300 MHz CPU and 256 MB real memory. The method was applied to two data sets of 300 dpi images: UW1 and Nonrect. UW1 consists of 25 page images with rectangular layout obtained from the University of Washington database 1. UW1 includes one-column (22 pages), two-column (2 pages), and three-column (1 page) pages whose layout is rectangular. We applied no skew correction to these images. Nonrect also consists of 25 images obtained by scanning various English magazines, but their layout is non-rectangular: they include document components of arbitrary shape. Nonrect includes one-column (3 pages), two-column (5 pages), three-column (13 pages), and four-column (4 pages) pages. In order to test the robustness against a severe skew, the images in Nonrect were artificially rotated counterclockwise by 10°.

The values of parameters used in the experiments are listed in Table 1. These values were determined using 50 different images: 25 images from the University of Washington database 1 and 25 images from magazines with non-rectangular layout.

4.2 Results and Discussion

Table 2 shows the results of the experiments. In this table, "no. of text-lines" and "correct" indicate the sum of the number of text-lines in each data set and the rate of correctly extracted text-lines, respectively. In the experiments, an extracted text-line is regarded as correct if it is neither fragmented nor merged with different document components, except for omission of small connected components such as hyphens, dots, commas and periods.

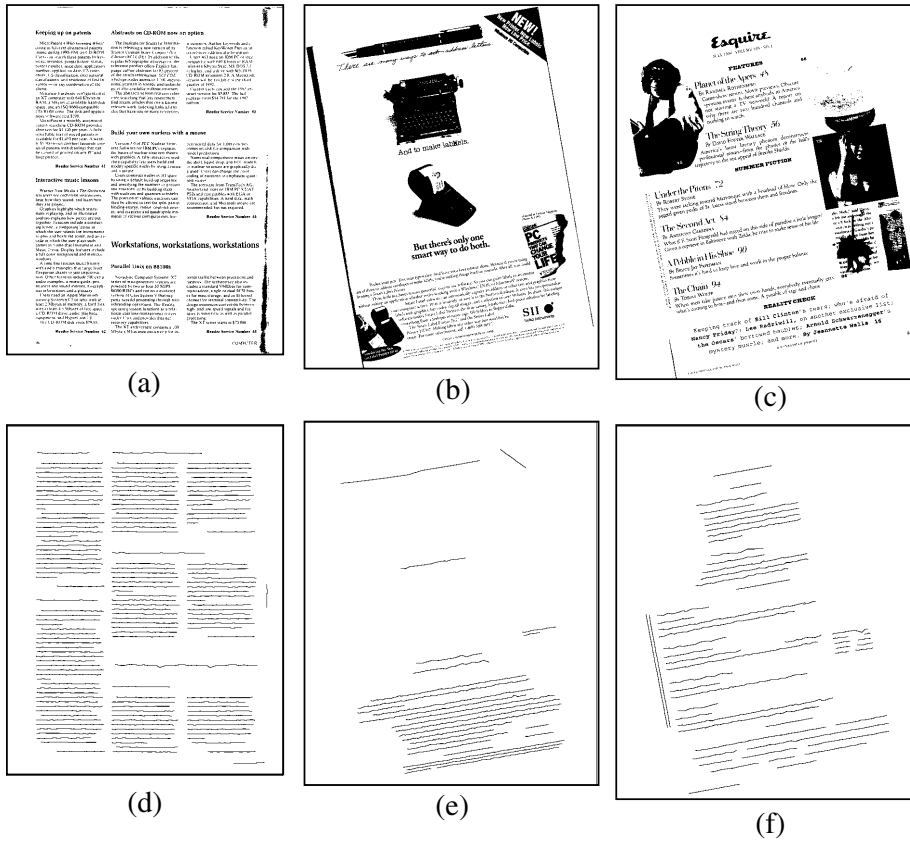


Fig. 9. Examples of processing results.

For both of the data sets, about 90% of text-lines were correctly extracted. This indicates that the method is capable of extracting text-lines independently of layout and skew.

Figure 9 illustrates examples of the processing results. Figure 9(a) belongs to UW1, and the rest are in Nonrect. Text-lines extracted from these images are shown in Figs. 9(d)–(f). Although a copy noise in Fig. 9(a), parts of figures in Fig. 9(b) and (c) were erroneously extracted, most of the extracted text-lines were correct ones. It is noteworthy that the text-lines with large characters in Fig. 9(a), the tilted text-line on the top right corner of Fig. 9(b) and those on the left edge of Fig. 9(c) were correctly extracted.

Errors can be classified into three types: fragmentation, over-merging, and omission of a whole text-line. Rates of the number of errors to the total number of text-lines are shown in Table 2, and examples of these errors are shown in Fig. 10.

(a) fragmentation (b) over-merging (c) omission

Fig. 10. Examples of errors.

Most of the fragmentation errors were due to wide inter-word gaps. A different type of fragmentation was found in text-lines which included equations and chemical symbols: long subscripts, superscripts and consecutive dots (...) severely disturbed the orientation of text-lines, so that edges to be merged could not be found. The fragmentation in UW1 occurred more frequently than that in Nonrect, since images in UW1 contained a larger number of symbols and equations.

Over-merging errors occurred more frequently for Nonrect. This was mainly because inter-column gaps were close to inter-word gaps in two images in Nonrect, one of which is shown in Fig. 10(b). It was also observed that some dotted ruled lines and parts of figures were merged with text-lines.

Omission occurred in short text-lines located on ends of paragraphs as well as headers and footers. It is difficult for the method to extract short text-lines because the chance of presence of seeds in short text-lines is less than that in long text-lines. Another reason is that the method deletes short text-lines consisting of less than or equal to M edges so as to eliminate noises.

We consider that these errors indicate the limitation of the method which extracts text-lines individually. In order to correct these errors, it is necessary to take account of text-blocks. The parallelness and the alignment of text-lines in a text-block could be fruitful features for correcting these errors.

Let us turn to the discussion about the efficiency of the method. On average, 60,000 sample points were extracted from 8,500 connected components to construct the neighbor graph consisting of 7,000 edges. 2,400 edges were selected as parts of text-lines for an image. The computation time required at each step is listed in Table 3. The process of labeling, contour following and sampling dominated the overall computation time. Although the construction of the area Voronoi diagram and the neighbor graph, and the feature extraction took the second and third longest time, respectively, each of them is about half of the time required by labeling. Thus, we consider that the method is applicable to images that can be labeled in reasonable time.

Table 3. Computation time.

labeling, contour following and sampling	3.39 sec. (49.7%)
feature extraction	1.40 sec. (20.5%)
construction of the area Voronoi diagram and the neighbor graph	1.74 sec. (25.5%)
seed extraction and extension	0.28 sec. (4.1%)
others (file I/O etc.)	0.01 sec. (0.2%)
total	6.82 sec.

Note. The ratio of time to the total is shown in parentheses.

Another important point is that the time consumed by the steps of extraction and extension of seeds was quite small as compared with other steps. This is because the method selects edges based only on the local examination.

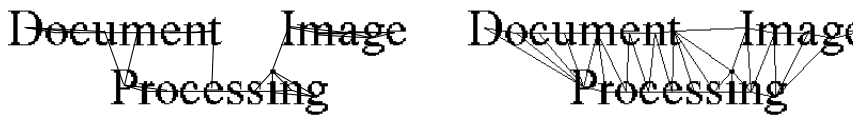
5 Related Work

In this section, we compare our method with some representative methods [1,2,3,4] which are also capable of extracting text-lines independently of layout and skew. We divide the discussion into the comparison of representations of physical structure, and that of strategies of extraction.

5.1 Representations

A representation of physical structure depends on the definition of neighbors of a connected component. A simple definition is that connected components are neighbors with each other if the distance between them is less than a threshold. Some of the methods [3,4] utilize this definition with modification for efficiency and effectiveness. However, it is necessary for such methods to determine the threshold.

As a representation without the threshold of distance, O'Gorman has utilized k-NN of connected components. Although this representation bears resemblance to the neighbor graph, an important difference is that k-NN requires a predetermined value of k. Figure 11 illustrates 3-NN and the neighbor graph for the same image, which looks like the image in Fig. 1 but the word gap between "Document" and "Image" is wider. As shown in Fig. 11(a), 3-NN does not cover the neighbor relation between "Document" and "Image". In order to obtain the neighbor relation between these two words, it is required to use k-NN with $k \geq 7$; the appropriate value of k varies depending on an image. On the other hand, the neighbor graph in Fig. 11(b) contains all neighbor relations required to extract text-lines without the use of such a parameter.



(a) 3-NN (b) neighbor graph

Fig. 11. k-NN and the neighbor graph.

5.2 Strategies

As a strategy of extraction, a variety of methods have been proposed. These include Hough transform [1], the angle/distance filtering of k-NN [2], relaxation [3], and simulated annealing [4]. In particular, relaxation and simulated annealing are effective for various text-lines, since they pursue globally optimized interpretations. In compensation for the effectiveness, however, they require a large amount of computation time. As compared with these methods, our method is fast but less accurate, since it extracts text-lines only by the local test. In order to improve the accuracy with a little loss of efficiency, it is required to expand the scope of test appropriately. A possible way would be the use of text-blocks — cooperative extraction of text-blocks [6] and text-lines.

6 Conclusion

We have presented the method of representing physical structure of pages by the neighbor graph as well as the method of extracting text-lines based on the representation. The use of the area Voronoi diagram enables us to extract the neighbor relations among connected components without the help of domain-specific parameters. The iterative extension of seeds based only on local selection of edges enables us to keep the method efficient.

From the experimental results for 50 images with a wide variety of layout and severe skew, we have confirmed that the method is capable of extracting text-lines independently of layout and skew. From the analysis of accuracy, however, it has been shown that there exist some limitations of the method which extracts text-lines with only a local evidence.

Future work is to overcome the limitations by developing a method of layout analysis which can cooperatively extract text-blocks and text-lines.

Acknowledgments

This research was supported in part by the Grant-in-Aid for Scientific Research from the Ministry of Education, Science, Sports and Culture, Japan, and The Telecommunication Advancement Foundation.

References

1. L. A. Fletcher and R. Kasturi, A robust algorithm for text string separation from mixed text/graphics images, *IEEE Trans. PAMI*, Vol. 10, No. 6, pp.910–918, 1988. 226, 238, 239
2. L. O'Gorman, The document spectrum for page layout analysis, *IEEE Trans. Pattern Anal. & Machine Intell.*, Vol. 15, No. 11, pp.1162–1173, 1993. 226, 238, 239
3. F. Hönes and J. Lichter, Layout extraction of mixed mode documents, *Machine Vision and Applications*, Vol. 7, pp.237–246, 1994. 226, 238, 239
4. K. Gyohten, T. Sumiya, N. Babaguchi, K. Kakusho and T. Kitahashi, A multi-agent based method for extracting characters and character strings, *IEICE Trans. Information & Systems, Japan*, Vol. E97-D, No. 5, pp.450–455, 1996. 226, 238, 239
5. K. Sugihara, Approximation of generalized Voronoi diagrams by ordinary Voronoi diagrams, *CVGIP: Graphical Models and Image Processing*, Vol. 55, No.6, pp.522–531, 1993. 226, 227
6. 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. 227, 239