A Unified Approach for Document Structure Analysis and its Application to Text-line Extraction

Jisheng Liang[†] **Ihsin T. Phillips**[‡] **Robert Haralick**[†]

[†] Department of Electrical Engineering
University of Washington Seattle, WA 98195

[‡] Department of Computer Science/Software Engineering
Seattle University, Seattle, WA 98122

Abstract

In this paper, we formulate the document segmentation as a partitioning problem. The goal of the problem is to find an optimal solution to partition the set of glyphs of a given document to a hierarchical tree structure where entities within the hierarchy have their physical properties and semantic labels. A unified approach is proposed for the partitioning problem. The Bayesian framework is used to assign and update the probabilities. An iterative, relaxation like method is used to find the partitioning solution that maximizes the joint probability.

We have implemented a text-line extraction algorithm using this framework. The algorithm was evaluated on the UW-III database of some 1600 scanned document image pages. For a total of 105,020 text lines, the text-line extraction algorithm identifies and segments 104,773 correctly, an accuracy of 99.76%. The detail of the algorithm is presented in this paper.

1 Introduction

Given a document image, the end result of a document segmentation algorithm, in general, produces a hierarchical structure that captures the physical structure and the logical meaning of an input document. The top of the hierarchical structure presents the entire page, and the bottom of the structure includes all glyphs on the document. Entities in the hierarchy are labeled and are associated with a set of attributes describing the nature of the entities. For example, the character set on a textual document would reside at the bottom of the hierarchy; each character would be labeled as a "glyph", and the attributes for the glyph may be the ASCII value, the font style, and the position of the character. The next level up may be words, then, text-lines, text-zones, text-blocks, and so on to the entire page.

Most known page segmentation algorithms [1]-[15] construct the document hierarchy from level to level, up and down within the hierarchy, until the hierarchical structures are built and the segmentation criteria are satisfied. Within this model, the page segmentation problem may be considered as a series of level-construction operations. That is, given a set of entities at a certain level of hierarchy, say source_level, the goal of the level-construction operation is to construct a set of entities for another level, say target_level.

In this paper, we propose a methodology for formulating and solving the document page segmentation problem. Our methodology uses the Bayesian framework. The methodology can be applied, uniformly, to any level-construction operation within the document hierarchy. To illustrate the usage of this methodology, a text-line extraction algorithm has been implemented and presented in this paper.

The remaining of this paper is organized as follows. In Section 2, we present the proposed methodology for the document segmentation problem and a general purpose algorithm derived from the methodology. In Section 3, we give, in detail, the text-line extraction algorithm which we implemented using the proposed methodology. In Section 4, we discuss how those probabilities used in the algorithm were computed. The paper summary is given in Section 6.

2 The Methodology

2.1 Document Structure Analysis Formulation

Let $\mathcal{A}$ be the set of entities at the source_level. Let Π be a partition of $\mathcal{A}$ and each element of the partition is an entity on target_level. Let L be a set of labels that can be assigned to elements of the partition. Function $f : \Pi \rightarrow L$ associates each element of Π with a label. $V : \wp(\mathcal{A}) \rightarrow \Lambda$ specifies measurement made on subset of $\mathcal{A}$, where Λ is the measurement space.

The problem can be formulated as follows: given initial set $\mathcal{A}$, find a partition Π of $\mathcal{A}$, and a labeling

function $f : \Pi \rightarrow L$, that maximizes the probability

$$\begin{aligned} & P(V(\tau) : \tau \in \Pi, f, \Pi | \mathcal{A}) \\ &= P(V(\tau) : \tau \in \Pi | \mathcal{A}, \Pi, f) P(\Pi, f | \mathcal{A}) \\ &= P(V(\tau) : \tau \in \Pi | \mathcal{A}, \Pi, f) \\ &\quad \times P(f | \Pi, \mathcal{A}) P(\Pi | \mathcal{A}) \end{aligned} \quad (1)$$

By making the assumption of conditional independence, that when the label $f(\tau)$ is known then no knowledge of other labels will alter the probability of $V(\tau)$, we can decompose the probability 1 into

$$\begin{aligned} & P(V(\tau) : \tau \in \Pi, f, \Pi | \mathcal{A}) \\ &= \prod_{\tau \in \Pi} P(V(\tau) | f(\tau)) P(f | \Pi, \mathcal{A}) P(\Pi | \mathcal{A}) \end{aligned} \quad (2)$$

The possible labels in set L is dependent on the target_level and on the specific application. For example, $l \in L$ could be text content, functional content type, style attribute, and so for.

The above proposed formulation can be uniformly apply to the construction of the document hierarchy at any level, e.g., text-word, text-line, and text-block extractions, just to name a few. For example, as for text-line extraction, given a set of glyphs, the goal of the text-line extraction is to partition glyphs into a set of text-lines, each text-line having homogeneous properties, and the text-lines' properties within the same region being similar. The text-lines' properties include, deviation of glyphs from the baseline, direction of the baseline, text-line's height, and text-lines' width, and so for.

As for the text-block segmentation, for example, given a set of text lines, text-block segmentation groups text lines into a set of text-blocks, each block having homogeneous formatting attributes, e.g. homogeneous leading, justification, and the attributes between neighboring blocks being similar.

2.2 A General Purpose Algorithm for Document Entity Extraction

Given an initial set $\mathcal{A}$, we first construct the read order of the elements of $\mathcal{A}$. Let $A = (A_1, A_2, \cdots, A_M)$ be a linearly ordered set (chain in $\mathcal{A}$) of input entities. Let $\mathcal{G} = \{Y, N\}$ be the set of grouping labels. Let A^P denote a set of element pairs, such that $A^p \subset A \times A$ and $A^p = \{(A_i, A_j) | A_i, A_j \in A \text{ and } j = i+1\}$. Function $g : A^p \rightarrow \mathcal{G}$, associates each pair of adjacent elements of A with a grouping label, where $g(i) = g(A_i, A_{i+1})$. Then, the partition probability $P(\Pi | A)$ can be computed as follows,

$$\begin{aligned} & P(\Pi | A) = P(g | A) \\ &= P(g(1), \cdots, g(N-1) | A_1, \cdots, A_N) \\ &= P(g(1) | A_1, A_2) \times \cdots P(g(N-1) | A_{N-1}, A_N) \\ &= \prod_{i=1}^{N-1} P(g(i) | A_i, A_{i+1}) \end{aligned} \quad (3)$$

Therefore, the joint probability is further decomposed as

$$\begin{aligned} & P(V(\tau) : \tau \in \Pi, f, \Pi | A) \\ &= \prod_{\tau \in \Pi} P(V(\tau) | f(\tau)) \times P(f | \Pi, A) \\ &\quad \times \prod_{i=1}^{N-1} P(g(i) | A_i, A_{i+1}) \end{aligned} \quad (4)$$

An iterative search method is developed to find the consistent partition and labeling that maximizes the joint probability of equation 4.

1. Determine initial partition

   Let $t = 0$, $\Pi^t = \{\{A_m\}\}_{m=1}^M$.

   (a) Compute $P_i^0(Y) = P(g(i) = Y | A_i, A_{i+1})$ and $P_i^0(N) = P(g(i) = N | A_i, A_{i+1})$ where $1 \leq i \leq M-1$.

   (b) Let $R \subseteq A \times A$ and $R = \{(A_i, A_{i+1}) | P_i^0(Y) > P_i^0(N)\}$. Update partition

   $$\Pi^{t+1} = \{\tau | \tau = \{A_i, A_{i+1}, \cdots, A_j\}, \text{where } (A_k, A_{k+1}) \in R, k = i, \cdots, j-1\}$$

2. Search for optimal partition adjustment

   Repeat

   - For $i = 1$ to $M-1$ Do
     - If $A_i \in U$, $A_{i+1} \in W$, $U \neq W$ Then,

       (a) Let
       $$T = U \bigcup W.$$
       and
       $$\hat{\Pi} = T \bigcup (\Pi^t - U - W)$$

       (b) Find labeling f by maximizing
       $$P_{label} = \prod_{\tau \in \hat{\Pi}} P(V(\tau) | f(\tau)) P(f | A, \hat{\Pi})$$

       (c) $\hat{P_i^t}(Y) \propto P_i^0(Y) \times P_{label}$, and $\hat{P_i^t}(N) = P_i^{t-1}(N)$.

     - If $A_i \in W$ and $A_{i+1} \in W$, where $W = \{A_k, \cdots, A_i, A_{i+1}, \cdots, A_j\}$, Then

       (a) $S = \{A_k, \cdots, A_i\}$ and $T = \{A_{i+1}, \cdots, A_j\}$
       $\hat{\Pi} = (\Pi^t - W) \bigcup S \bigcup T$

       (b) Find labeling f by maximizing
       $$P_{label} = \prod_{\tau \in \hat{\Pi}} P(V(\tau) | f(\tau)) P(f | A, \hat{\Pi})$$

       (c) $\hat{P_i^t}(N) \propto P_i^0(N) \times P_{label}$, and $\hat{P_i^t}(Y) = P_i^{t-1}(Y)$

End

- Select k such that,

$$k = \arg\max_i(\max\{\hat{P}_i^t(Y), \hat{P}_i^t(N)\})$$

- If $P_k^t\hat{(}Y) > P_k^t\hat{(}N)$, Then
  - $T = U \bigcup W$ where $A_k \in U, A_{k+1} \in W$
  - $\Pi^{t+1} = (\Pi^t - U - W) \bigcup T$

  Else, $W = \{A_i, \cdots, A_k, A_{k+1}, \cdots, A_j\}$,
  - Let $S = \{A_i, \cdots, A_k\}$ and $T = \{A_{k+1}, \cdots, A_j\}$
  - $\Pi^{t+1} = (\Pi^t - W) \bigcup S \bigcup T$
- If $P(V, f, \Pi^{t+1}|A) \leq P(V, f, \Pi^t|A)$, end and return Π^t.
  Else, let $t = t + 1$ and continue.

Our method consists of two major components – off-line statistical training and on-line segmentation. Section 3 presents our on-line algorithm of text-line and zone segmentation. Our statistical training method is given in section 4.

3 Text-line Extraction Algorithm

Figure 1 gives an overview of the text-line segmentation algorithm. Without loss of generality, we assume that the reading direction of the text-lines on the input page is left-to-right. The text-line segmentation algorithm starts with the set of the connected-components bounding boxes of a given binary image of a textual document.

Algorithm:

1. Extract & Filter Glyphs:

   We apply the standard connected-component algorithm to obtain the glyph set, $C = \{c_1, c_2, \cdots, c_M\}$. Those components that are smaller than the $threshold_{small}$ or larger than the $threshold_{large}$ are removed from C.

2. Locate Glyph Pairs:

   For each $c_i \in C$, we search for its "nearest right mate", c_j, among those "visible" right neighbors of c_i. When a right mate is found, a link is established between the pair. The definitions for the nearest right mate and the visible right neighbors are given in section 3.1. Note that, a glyph at the right-most edge of a document would not have a right mate. At the end of this step a set of text-line segments are established, $T_{segment} = \{t_1, t_2, \cdots, t_{K_1}\}$. For each linked pair, c_i and c_j, we compute the grouping probability, $P(sameline(i,j)|c_i, c_j)$. This is the estimated probability that two components with their sizes and spatial relationships lie on the same text-line.

3. Group Text-lines:

   For each $t_k \in T_{segment}$ formed in step 2, we check each link $(c_i, c_j) \in t_k$ and estimate the linking probability $P(link(i,j))$ between c_i and c_j. If $P(link(i,j) = N) > P(link(i,j) = Y)$, we disconnect (c_i, c_j) link. That is, t_k becomes two subsegments.

   During the initial partition, $P(link(i,j)) = P(sameline(i,j))$, and this step yields our initial text-line set, $T_{initial} = \{t_1, t_2, \cdots, t_{K_2}\}$.

4. Detect Base-line & X-height:

   For each $t_k \in T_{initial}$, we apply a robust line fitting algorithm to the right-bottom corners of all glyphs in t_k to obtain the base-line and the direction of t_k. The computation of base-line and X-height are given in Section 3.2.

5. Detect Page Skew:

   The median of all the computed base-lines' direction for the entire set $T_{initial}$ is taken as the page skew angle, $angle_{skew}$. If the $angle_{skew} > threshold_{skew}$, we rotate the image by $-angle_{skew}$ using the technique given in [17], and the process repeats from step 1. Otherwise, proceed to the next step.

6. Compute Text-line Probability:

   For each text-line $t_k \in T_{initial}$, We compute its probability of having the homogeneous text-line properties,

$$P(V(t_k)|textline(t_k)),$$

   where $V(t_k)$ is the measurement made on the text-line t_k.

   The observation that we make is the component distance deviation σ_{t_k} of t_k from its base-line. If $P(\sigma_{t_i}|textline(t_i)) > threshold_\sigma$ we accept t_i. Otherwise, we pick the weakest link (c_i, c_j) within t_k as the potential breaking place where we may sub-divide t_j into t_{j1} and t_{j2}.

7. Adjust Pairs linking Probability:

   To determine whether to sub-divide t_k, we compute t_{k1}'s and t_{k2}'s base-lines, and their component deviations, $\sigma_{t_{k1}}$ and $\sigma_{t_{k2}}$.

   We update the linking probability between c_i and c_j by combining their grouping probability with the text-line probability,

$$\begin{aligned} & P(link(i,j) = Y) \\ & \quad \propto \quad P(sameline(i,j) = Y|c_i, c_j) \\ & \qquad \times P(\sigma_{t_k}|textline(t_k)), \end{aligned}$$

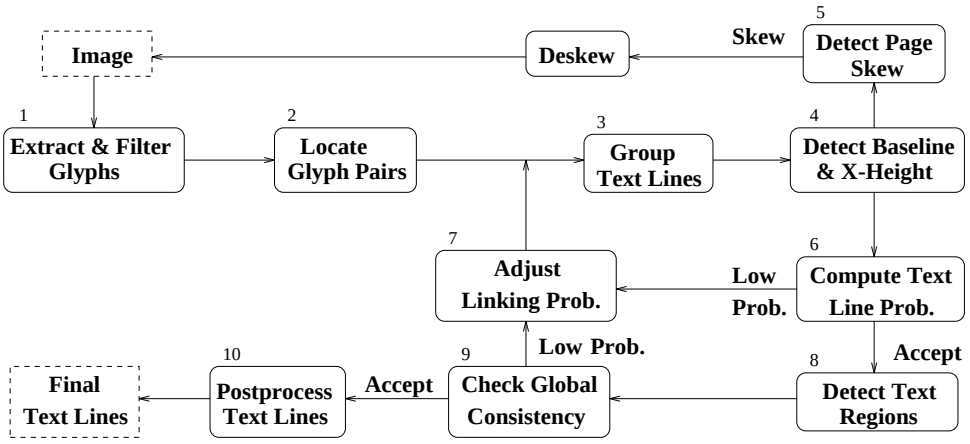


Figure 1: Illustrates the processing steps of the text-line segmentation algorithm.

and

$$\begin{aligned} & P(link(i,j) = N) \\ \propto \quad & P(sameline(i,j) = N | c_i, c_j) \\ & \times P(\sigma_{t_{k1}} | textline(t_{k1})) \\ & \times P(\sigma_{t_{k2}} | textline(t_{k2}), \end{aligned}$$

where

$$P(link(i,j) = Y) + P(link(i,j) = N) = 1.$$

If $P(link(i,j) = N) > P(link(i,j) = Y)$, the process repeats from step 3. Otherwise, proceed to the next step.

8. Detect Text Regions and Zones:

   To detect text-regions with respect to all text-lines in $T_{interim}$, we do as follows. For each text-line $t_k \in T_{interim}$, we compute it's bounding box and the three bounding box edge positions: the left, the center, and the right.

   Then, a horizontal projection profile is computed on all the text-line bounding boxes. Each text-line box constitutes one count on the profile. A horizontal cut is made where the gap within the profile satisfies our cutting criteria. The computation of the projection profile and the cutting criteria are given in detail in section 3.3.

   The result of the last step is a sequence of horizontal text-regions, $R = \{R_1, R_2, \cdots, R_r\}$. In this step, each of the region, R_i, is to be further decomposed into a sequence of text-zones by cutting R_i vertically. The top and the bottom edges of R_i become the top and the bottom edges of the text-zones. Our text-zone detection finds the left and the right edges of text-zones within R_i.

   Let $R_i = \{t_1, t_2, \cdots, t_p\}$ be a horizontal text-region, $R_i \in R$. To detect a text-zone within R_i, we compute the vertical projection profile on the left, the center, and the right positions of all text-lines $t_k \in R_i$.

   Next, we locate the bin with the max count on the profile. If the max count comes from, say, the left position of the majority of the text-lines that contribute to the max count, we say, we have detected a left-edge of a text-zone, Z_n. Let $\{t_1, t_2, \cdots, t_m\}$ be the sequence of text-lines whose left positions fall within the bin which has the max count. The left-edge of Z_n is estimated as the median of the left edge position of all text-lines within Z_n. The right edge of Z_n is computed in a similar fashion. The top and the bottom edges of Z_n are the top and the bottom edges of R_i. Then, all the text-lines within Z_n are removed from further consideration, and this step is repeated until each text-line in R is assigned to one of the detected text-zones. A complete description of this step is given in section 3.3.

9. Check Global Consistency (Splits & Merges):

   Let $Z = \{Z_1, Z_2, \cdots, Z_n\}$ be the set of the detected text-zones from the last step. Let $Z_i \in Z$ and $Z_i = \{t_1, t_2, \cdots, t_k\}$. We examine the probability, $P_{context}(\omega(t_j), \omega(Z_i))$, that t_j's attributes $\omega(t_j)$ being consistent with its neighboring text-lines within Z_i. (The computation of $P_{context}$ is given in section 3.4.)

   If $P_{context}(t_j) < threshold_{context}$, we update the linking probability for each pair within t_j, and the process repeats from step 3. Step 8 and 9 are repeated until $P_{context}(t_j) > threshold_{context}$ is satisfied for all t_j. The complete description of the global consistent check, the split and the merge procedures are given in detail in section 3.4.

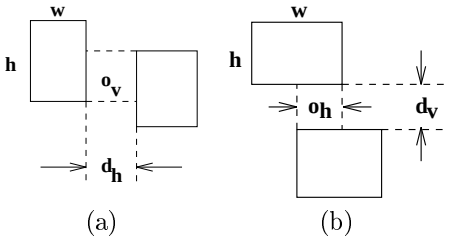
10. Postprocess Text Lines: Finally, all components which were initially put into the reserved set and those text-lines which were not included during the text-zone formation, or as the results of splitting, are now be individually examined to determine whether it could be included in any of the segmented text-lines.

Figure 2 and 3 illustrate the text line detection process. Figure 2(a) shows a set of connected component bounding boxes. The extracted initial text line segments by merging pairs of connected components are illustrated in Figure 2(b). We notice some text lines are split while some are merged across different columns. Figure 3(c) plots the extracted text regions by grouping the edges of text segments. Finally, the corrected text lines given the observations on text regions are shown in Figure 3(d).

A few cases that the algorithm failed are shown in Figure 4. A vertical merging error was shown in Figure 4(a). Figure 4(b) and (c) illustrate horizontal and vertical splitting errors due to the large spacing. A spurious error caused by warping is shown in Figure 4(d).

3.1 Mate Pairs and Grouping Probability

Let $C = \{c_1, c_2, \cdots, c_M\}$ be the set of glyphs, the connected-component set after the too small components are removed. Each glyph $c_i \in C$ is represented by a bounding box (x, y, w, h), where x, y is the coordinate of top-left corner, and w and h are the width and height of the bounding box respectively. The spatial relations between two adjacent boxes are shown in Figure 5.



(a) (b)

Figure 5: Illustrates the spatial relations between two bounding boxes that are (a) horizontally adjacent (b) vertically adjacent.

For a pair of bounding boxes a and b, the horizontal distance $d_h(a,b)$ and vertical distance $d_v(a,b)$ between them are defined as

$$d_h(a,b) = \begin{cases} x_b - x_a - w_a & \text{if } x_b > x_a + w_a \\ x_a - x_b - w_b & \text{if } x_a > x_b + w_b \\ 0 & \text{otherwise} \end{cases}$$

$$d_v(a,b) = \begin{cases} y_b - y_a - h_a & \text{if } y_b > y_a + h_a \\ y_a - y_b - h_b & \text{if } y_a > y_b + h_b \\ 0 & \text{otherwise} \end{cases}$$

The horizontal overlap $o_h(a,b)$ and vertical overlap $o_v(a,b)$ between a and b are defined as

$$o_h(a,b) = \begin{cases} x_a + w_a - x_b & \text{if } x_b > x_a,\ x_b < x_a + w_a \\ x_b + w_b - x_a & \text{if } x_a > x_b,\ x_a < x_b + w_b \\ 0 & \text{otherwise} \end{cases}$$

$$o_v(a,b) = \begin{cases} y_a + h_a - y_b & \text{if } y_b > y_a,\ y_b < y_a + h_a \\ y_b + h_b - y_a & \text{if } y_a > y_b,\ y_a < y_b + h_b \\ 0 & \text{otherwise} \end{cases}$$

Let $c_a = (x_a, y_a, w_a, h_a)$ and $c_b = (x_b, y_b, w_b, h_b)$ be two glyphs. We define c_b as a "visible" right neighbor of c_a if $c_b \neq c_a, x_b > x_a,$ and $o_v(a,b) > 0$. Let C_a be the set of right neighbors of c_a. The "nearest" right neighbor of c_a is defined as

$$\arg\min_{c_i \in C_a} (d_h(a,i) | c_i \neq c_a, x_i > x_a, o_v(a,i) > 0).$$

For each linked pair, c_a and c_b, we associate with their link with the probability, $P(sameline(a,b)|c_a, c_b)$, that indicate how probable they belong to the same text-line. Given the observations of their heights and widths, and the distance and the overlaps between the pair: h_a, w_a, h_b, w_b, $d(a,b)$, $o(a,b)$, we compute the probability that c_a and c_b belong to the same text-line as:

$$P(sameline(a,b)|h_a, w_a, h_b, w_b, d(a,b), o(a,b)).$$

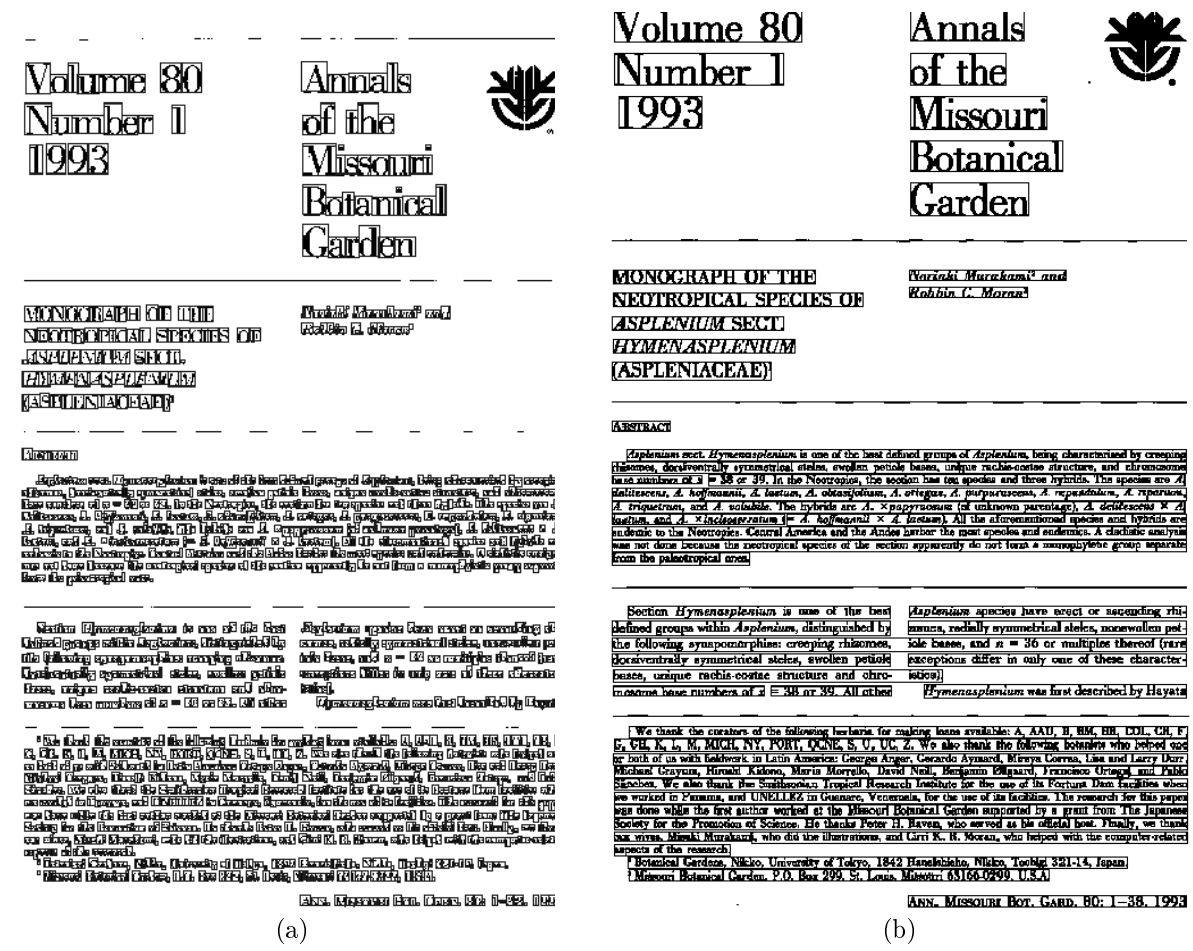
3.2 Base-line, X-height, and Skew Angle

The baseline coordinate of a text-line is estimated using a robust estimator. The robust estimation means it is insensitive to small departures from the idealized assumptions for which the estimator is optimized.

We want to fit a straight line $y(x; a, b) = a + bx$ through a set of data points, which are the bottom-right corner of glyph boxes, since ascenders are used more often in English texts than descenders. The merit function to be minimized is

$$\sum_{i=1}^{N} |y_i - a - bx_i|.$$

The median c_M of a set of numbers c_i is also the value which minimizes the sum of the absolute deviations $\sum_i |c_i - c_M|$. It follows that, for fixed b, the value of a that minimizes the merit function is $a = median\{y_i - bx_i\}$, where $b = \sum_{i=1}^{N} sgn(y_i - a - bx_i)$. This equation can be solved by the bracketing and bisection method [16].



(a) (b)

Figure 2: Illustrates a real document image overlaid with the extracted bounding boxes of (a) the connected components; and (b) the initial text line segments.

Given a set of baseline angles $\{\theta_1, \theta_2, \cdots, \theta_P\}$, the skew angle of page is estimated as

$$\theta_{page} = \text{median}\{\theta_1, \theta_2, \cdots, \theta_P\}.$$

If skew angle θ is larger than the threshold, $threshold_\theta$ the page will be rotated by $-\theta$.

For each given text-line t_i and the estimated baseline (a, b), we compute the absolute deviation of glyph from the estimated baseline

$$\sigma(t_i, a, b) = \sum_{i=1}^{N} |y_i - a - bx_i|.$$

The x-height of a text-line is estimated by taking the median of the distance from the top-left corner of each glyph box to the baseline

$$xh(t_i) = \text{median}\{d(x_i, y_i, a, b) | 1 \leq i \leq N\}.$$

Given the observations on text-line t_i, we can compute the likelihood that t_i has the property of a text-line

$$P(xh(t_i), \sigma(t_i, a, b)) | \text{textline}(t_i)).$$

3.3 Text-zone Formation

Horizontal Projection of the Text Line Boxes

Given a set of text-line bounding boxes $T = \{t_1, t_2, \cdots, t_M\}$, our goal is to group them into a sequence of horizontal text-regions $R = \{R_1, R_2, \cdots, R_N\}$. We do the following.

Let (x_i, y_i, w_i, h_i) represents the bounding box of the text-line $t_i \in T$. t_i is bounded by x_i and $x_i + w_i$.

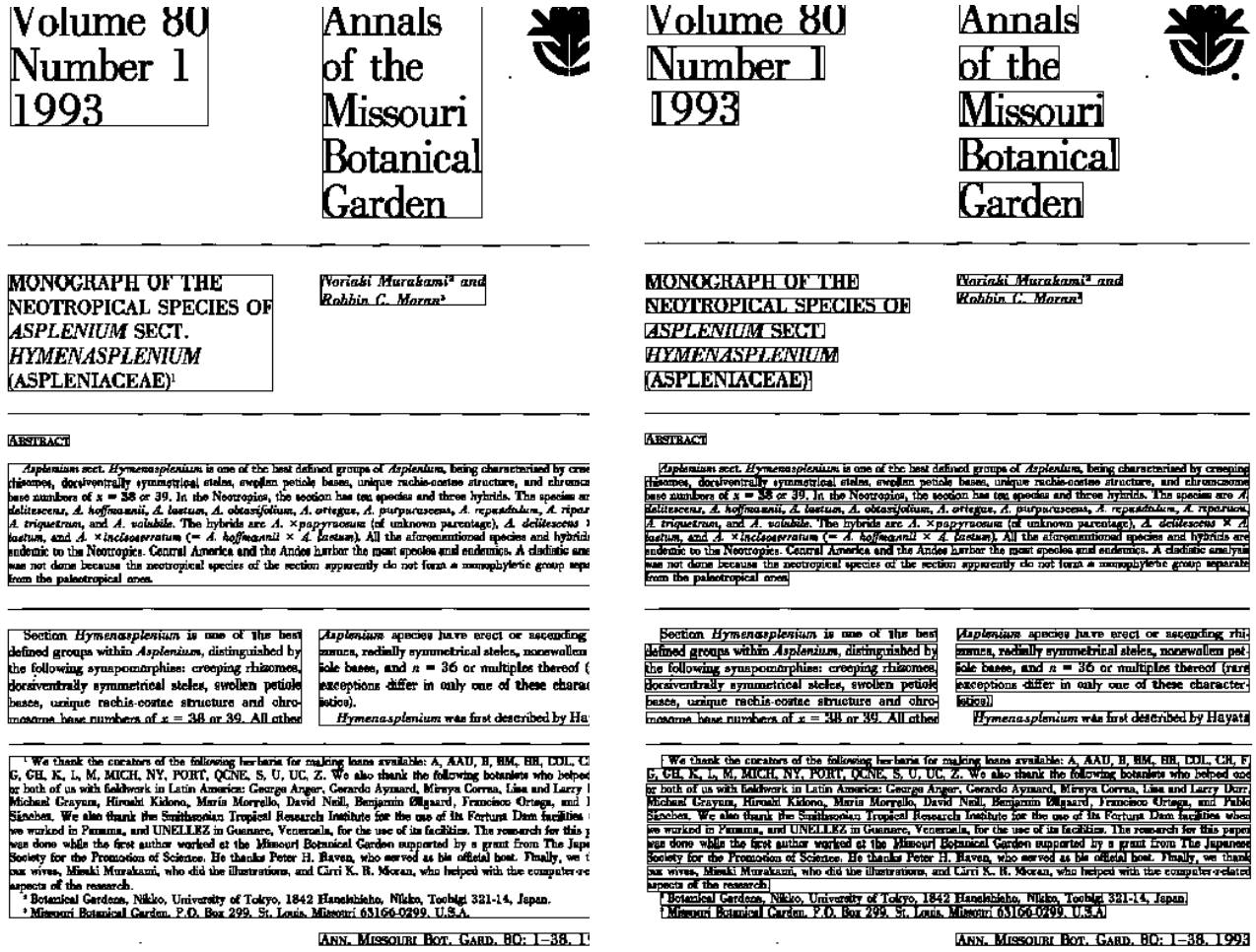
Given an entity box (x, y, w, h), its horizontal projection (Figure 6) is defined as

$$\text{horz-profile}[j] = \text{horz-profile}[j] + 1, x \leq j < x + w.$$

Vertical Projection of the Text Line Edges

The vertical projection of a set of entities is defined as

$$\text{vert-profile}[j] = \text{vert-profile}[j] + 1, y \leq j < y + h.$$



(c) (d)

Figure 3: Illustrates a real document image overlaid with the extracted bounding boxes of (c) the text regions; and (d) the corrected text lines.

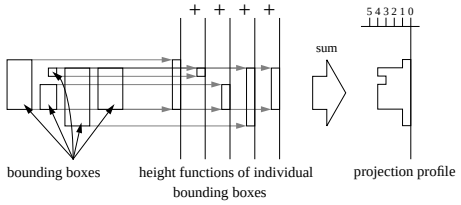


Figure 6: Illustrates the horizontal projection of bounding boxes.

Let (x_i, y_i, w_i, h_i) represents the bounding box of a text-line $t_i \in T$. We assign the left edge of t_i to be x_i, the right edge of t_i to be $x_i + w_i$, and the center of t_i to be $x_i + w_i/2$. The vertical edge projection on the three edges of the text-line bounding boxes of all $t_i \in T$ is defined as:

$$\begin{aligned} C_{left}[j] &= C_{left}[j] + 1, j = x \\ C_{center}[j] &= C_{center}[j] + 1, j = x + w/2 \\ C_{right}[j] &= C_{right}[j] + 1, j = x + w. \end{aligned}$$

Text-zone Detection Algorithm

1. Compute the horizontal projection profile of all text-line boxes.

2. Segment the page into a set of large regions, by making cut at the gaps of horizontal projection profile, where the width of gap is larger than a certain threshold. The threshold is determined by the median height of detected text-lines.

3. For each region

   (a) Compute the vertical projection count C of the left edges E_{left}, right edges E_{right} , and center edges E_{center} of text-line boxes.

   (b) Find a place which has the highest total count within its neighborhood of width w. $x = \arg_{i,j} \max(\sum_k C_{i,k}, i \in \{left, right, center\}, j - \frac{1}{2}w \leq k < j + \frac{1}{2}w)$, where w is determined by the dominant text-line height within the region.

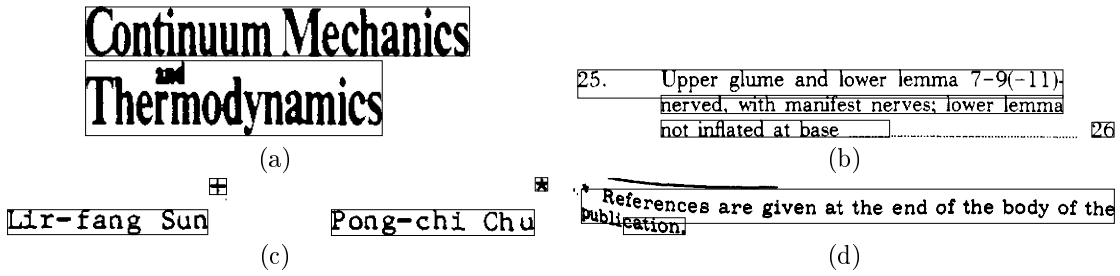


Figure 4: Illustrates examples that the text detection algorithm failed.

(c) Determine the zone edge as the median of edges E_{ik}, within the neighborhood $j - \frac{1}{2}w \leq k < j + \frac{1}{2}w$.

(d) For each edge E_{ik}, finding its corresponding edge of the other side of the box $E_{jk}, j \neq i$

(e) Determine the other edges of this zone by taking the median of E_{jk}

(f) Remove the text-line boxes enclosed by the detect zone from T

(g) If $T = \emptyset$, an empty set, we are done, otherwise, repeat this step.

If the inter-zone spacing between two adjacent zones is very small, it may cause the majority of text-lines from those two zones to merge. On the other hand, a list-item structure usually has large gaps and this causes splitting errors. In order to detect these two cases, we compute the vertical projection profile of glyph enclosed by each zone.

If there is a zero-height valley in the profile, compute the probability that the region should be split into two zones

$$P(\text{twozone}(c)|w_{gap}, n, h_m, h_l, h_r, w_l, w_r),$$

where w_{gap} is the width of profile gap, n is the total number of text-lines within the current region c, h_m is the median of text-line height within c. h_l and w_l (h_r and w_r) are the height and width of the region on the left (right) side of gap. If the probability is larger than a certain threshold, split the region at the detected gap.

Given a pair of adjacent zones, the probability that they are part of the list-item structure is:

$$P(\text{list-item}(c_l, c_r)|w_{gap}, h_l, h_r, w_l, w_r, n_l, n_r),$$

where n_l and n_r are the number of text-line within the left and right zones respectively.

3.4 Text-line Splitting and Merging

Given the detected zones, we can determine if a text-line is horizontally merged or split, or vertically merged or split.

Given the observations on a text-line $t = (c_1, c_2, \cdots, c_m)$ and its neighbors $N(t)$ within the same zone Z, we compute the probability that t is vertically consistent, merged, or split:

$$P(\text{v-consistent}(t, N(t))|h(t), h_N(t), h_t(c), h_N(c)),$$

where $h(t)$ is the height of text-line t, $h_N(t)$ is the median of text-line height in zone $N(t)$, $h_t(c)$ is the median height of glyphs in t, and $h_N(c)$ is the median height of glyphs in $N(t)$. Then, we can update the linking probability between a pair of adjacent glyphs c_i and c_j:

$$\begin{aligned} P(link(i,j)) &\propto P(\text{sameline}(i,j)|c_i, c_j) \\ &\times P(\text{v-consistent}(t, N(t))), \end{aligned}$$

where $c_i \in t$, and $c_j \in Z$.

Given a pair of adjacent text-lines t_m and t_n within the same zone, we can update the linking probability between a pair of glyph $c_i \in t_m$ and $c_j \in t_n$:

$$\begin{aligned} P(link(i,j)) &\propto P(\text{sameline}(i,j)|c_i, c_j, \text{samezone}(i,j)) \\ &\propto P(c_i, c_j|\text{sameline}(i,j))P(\text{sameline}(i,j)) \\ &\times P(\text{samezone}(i,j)|\text{sameline}(i,j)). \end{aligned}$$

Similarly, if a text-line is across two or more zones, we can update the linking probability for each pair of adjacent glyph that belong to different zones

$$\begin{aligned} P(link(i,j)) &\propto P(\text{sameline}(i,j)|c_i, c_j, \text{diffzone}(i,j)) \\ &\propto P(c_i, c_j|\text{sameline}(i,j))P(\text{sameline}(i,j)) \\ &\times P(\text{diffzone}(i,j)|\text{sameline}(i,j)). \end{aligned}$$

4 Probability Estimation

Discrete lookup tables are used to represent the estimated joint and conditional probabilities used at each of the algorithm decision steps. We first quantize the value of each variable into a finite number of mutually exclusive states. If A is a variable with states $a_1, \cdots, a_n$, then $P(A)$ is a probability distribution over these states: $P(A) = (x_1, \cdots, x_n)$

where $x_i \geq 0$ and $\sum_{i=1}^{n} x_i = 1$. Here, x_i is the probability of A being in state a_i. If the variable B

has states $b_1, \cdots, b_m$, then $P(A|B)$ is an $n \times m$ table containing numbers $P(a_i|b_j)$. $P(A, B)$, the joint probability for the variables A and B, is also an $n \times m$ table. It consists of a probability for each configuration (a_i, b_j).

We conduct a series of experiments to empirically determine the probability distributions that we used to extract text lines. A tree structure quantization is used to partition the value of each variable into bins. At each node of the tree, we search through all possible threshold candidates on each variable, and select the one which gives minimum value of entropy. The total number of terminal nodes, which is equivalent to the total number of cells, is predetermined. Finally, the bins on each variable form the cells in the space. For each joint or conditional probability distribution, a cell count is computed from the the ground-truthed document images in the UW-III Document Image Database. Rather than entering the value of each variable for each individual in the sample, the cell count records, for each possible combination of values of the measured variables, how many members of the sample have exactly that combinations of values. A cell count is simply the number of units in the sample that have a given fixed set of values for the variables. The joint probability table can be computed directly from the cell count.

A few parameters, such as those thresholds used in the algorithms. Their values are estimated. A representative sample of a domain was used and a quantitative performance metric was defined. We tuned the parameter values of our algorithm and selected the set which produces the optimal performance on the input population. Assuming the criterion function is unimodal in the parameter value within a certain range, we used a golden section search method to find the optimal value within that range.

5 Experimental Results

We applied our text-line extraction algorithm to the total of 1600 images from the UW-III Document Image Database. The numbers and percentages of miss, false, correct, splitting, merging and spurious detections are shown in Table 1. Of the 105,020 ground truth text-lines, 99.76% of them are correctly detected, and 0.08% and 0.07% of lines are split or merged, respectively. Most of the missing errors are due to the rotated text.

6 Summary

In this paper, we formulate the document segmentation as a partitioning problem. The goal of the problem is to find an optimal solution to partition the set of glyphs on a given document to a hierarchical tree structure where entities within the hierarchy are associated with their physical properties and semantic labels. A unified approach is proposed. The Bayesian framework is used to assign and update the probabilities during the segmentation. An iterative, relaxation like method is used to find the partitioning solution that maximizes the joint probability.

A text-line extraction algorithm has been implemented to demonstrate the usage of this framework. This algorithm consists of two major components – off-line statistical training and on-line text-line extraction. The probabilities used within this algorithm are estimated from an extensive training set of various kinds of measurements of distances between the terminal and non-terminal entities with which the algorithm works. The off-line probabilities estimated in the training then drive all decisions in the on-line segmentation module. The on-line segmentation module first extracts and filters the set of connected components of the input image to obtain a set of glyphs. Each glyph is linked to its adjacent neighbor to form glyph pars. Associated with each link is the pair's linking probability. The entire text-line extraction process can be viewed as an iterative re-adjustment of the pairs' linking probabilities on the glyph set. The segmentation algorithm terminates when the decision can be made in favor for each link within the final set of text-line segments.

The algorithm was tested on the 1600 pages of technical documents within the UW-III database. A total of 105020 text lines within these pages, the algorithm exhibits a 99.8% accuracy rate. Currently, we are implementing a text-block extraction algorithm, also using the proposed framework. This new algorithm is currently at the testing phrase and the prelimary result looks promosing.

References

[1] S. Srihari and W. Zack, Document Image analysis, *Proceedings of the 8th International Conference on Pattern Recognition (ICPR'86)*, pp. 434-436, July 1986, Paris, France.

[2] N. Amamoto, S. Torigoe and Y. Hirogaki, Block Segmentation and Text Area Extraction of Vertically/Horizontally Written Documents, *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 739-742, October 1993, Tsukuba, Japan.

[3] M. Okamoto and M. Takahashi, A Hybrid Page Segmentation Method, *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 743-746, October 1993, Tsukuba, Japan.

Table 1: Performance of text-line extraction algorithms.

	Total	Correct	Splitting	Merging	Mis-False	Spurious
Ground Truth	105020	104773 (99.76%)	80 (0.08%)	78 (0.07%)	79 (0.08%)	10 (0.01%)
Detected	105019	104773 (99.77%)	172 (0.16%)	37 (0.04%)	25 (0.02%)	12 (0.01%)

[4] T. Saitoh, M. Tachikawa and T. Yamaai, Document Image Segmentation and Text Area Ordering, *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 323-329, October 1993, Tsukuba, Japan.

[5] D.J. Ittner and H.S. Baird, Language-Free Layout analysis, *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 336-340, October 1993, Tsukuba, Japan.

[6] Y. Hirayama, A Block Segmentation Method for Document Images with Complicated Column Structures, *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 91-94, October 1993, Tsukuba, Japan.

[7] T. Pavlidis and J. Zhou, Page Segmentation and Classification, *CVGIP, Graphical Models and Image Processing*, Vol. 54, pp. 484-496, November 1992.

[8] L. OGorman, The Document Spectrum for Page Layout Analysis, *IEEE Transactions of Pattern Analysis and Machines Intelligence*, pp. 1162-1173, November 1993.

[9] G. Nagy and S. Seth, Hierarchical Representation of Optically Scanned Documents, *Proceedings of the 7th International Conference on Pattern Recognition (ICPR'84)*, pp. 347-349, July 1984, Montreal, Canada.

[10] H.S. Baird, Background Structure in Document Images, *International Journal of Pattern Recognition and Artificial Intelligence*, pp. 1013-1030, October 1994.

[11] F. Jones and J. Litchter, Layout Extraction of Mixed Mode Documents, *Machines Vision and Applications*, Vol. 7, No. 4, pp. 237-246, 1994.

[12] S-Y. Wang and T. Yagasaki, Block Selection: A Method for Segmenting Page Image for Various Editing Styles, *Proceedings of the 3rd International Conference on Document Analysis and Recognition (ICDAR'95)*, pp. 128-133, August 1995, Montreal, Canada.

[13] F. Esposito, D. Malerba and G. Semeraro, A Knowledge-Based Approach to the Layout Analysis, *Proceedings of the 3rd International Conference on Document Analysis and Recognition (ICDAR'95)*, pp. 466-471, August 1995, Montreal, Canada.

[14] J. Ha, R.M. Haralick and I. Phillips, Document Page Decomposition by the Bounding-Box Projection, *Proceedings of the 3rd International Conference on Document Analysis and Recognition (ICDAR'95)*, pp. 1119-1122, August 1995, Montreal, Canada.

[15] S. Chen, R.M. Haralick and I. Phillips, Extraction of Text Lines and Text Blocks on Document Images Based on Statistical Modeling, *International Journal of Imaging Systems and Technology*, Vol. 7, No. 4, pp. 343-356, Winter, 1996.

[16] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, 1992.

[17] S. Chen, R.M. Haralick and I. Phillips, Automatic Text Skew Estimation in Document Images, *Proceedings of the 3rd International Conference on Document Analysis and Recognition (ICDAR'95)*, pp. 1153-1156, August 1995, Montreal, Canada.

[18] I. Phillips, *Users' Reference Manual*, CD-ROM, UW-III Document Image Database-III, 1995.

[19] I. Phillips, S. Chen and R. Haralick, CD-ROM Document Database Standard, *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 478-483, October 1993, Tsukuba, Japan.

[20] I. Phillips, J. Ha and R. Haralick and D. Dori, The Implementation Methodology for the CD-ROM English Document Database, *Proceedings of the 2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pp. 484-487, October 1993, Tsukuba, Japan.