High Performance Document Layout Analysis

Article · May 2003

Source: CiteSeer

CITATIONS
55

READS
348

1 author:

Thomas Breuel
Google Inc.
257 PUBLICATIONS **4,323** CITATIONS

SEE PROFILE

High Performance Document Layout Analysis

Thomas M. Breuel
PARC, Palo Alto, CA, USA
tmb@parc.com

Abstract

In this paper[1], I summarize research in document layout analysis carried out over the last few years in our laboratory. Correct document layout analysis is a key step in document capture conversions into electronic formats, optical character recognition (OCR), information retrieval from scanned documents, appearance-based document retrieval, and reformatting of documents for on-screen display. We have developed a number of novel geometric algorithms and statistical methods. Layout analysis systems built from these algorithms are applicable to a wide variety of languages and layouts, and have proven to be robust to the presence of noise and spurious features in a page image. The system itself consists of reusable and independent software modules that can be reconfigured to be adapted to different languages and applications. Currently, we are using them for electronic book and document capture applications. If there is commercial or government demand, we are interested in adapting these tools to information retrieval and intelligence applications.

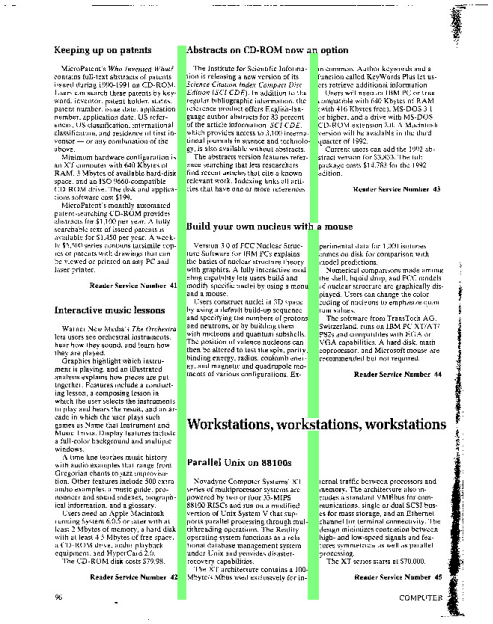
1 Introduction

Document layout analysis is a key step in converting document images into electronic form. Document layout analysis identifies key parts of a document, like titles, abstracts, sections, page numbering, and puts the text on a page into the correct reading order, which is a prerequisite for optical character recognition (OCR), as well as most forms of document retrieval.

Traditional document layout analysis methods will generally first attempt to perform a complete global segmentation of the document into distinct geometric regions corresponding to entities like columns, headings, and paragraphs using features like proximity, texture, or whitespace. Segmentation into regions are often carried out using heuristic methods based on morphology or "smearing" based approaches, projection profiles (recursive X-Y cuts), texture-based analysis, analysis of the background structure, and others (for a review and references, see [7]). Each individual region is then considered separately for tasks like text line finding and OCR. The problem with this approach lies in the fact that obtaining a complete and reliable segmentation of a document into separate regions is difficult to achieve in general. Some decisions about which regions to combine may well involve semantic constraints on the output of an OCR system. However, in order to be able to pass the document to the OCR system in the first place, we must already have identified text lines, leading to circular dependencies among the processing steps.

In our work, we use exact and globally optimal geometric algorithms, combined with robust statistical models, to model and analyze the layout of pages. By combining these algorithms carefully, we arrive at an overall approach to document layout analysis that avoid the circular dependencies of traditional methods and greatly reduces the number of parameters needed to "tune the system". The resulting document layout analysis systems are *applicable to a wide variety of languages and layouts*, and have proven to be *robust to the presence of noise* and spurious features in a page image. Below, we will first examine the individual steps and algorithms needed for this approach to document layout analysis and then describe how these algorithms are put together into an overall document layout analysis system. It can be accomplished reliably using the whitespace analysis algorithm described in this paper using a novel evaluation function.

[1] This paper is a compilation of results, figures, and descriptions from a number of previously published papers. Please see the individual sections for attributions and references.



Keeping up on patents

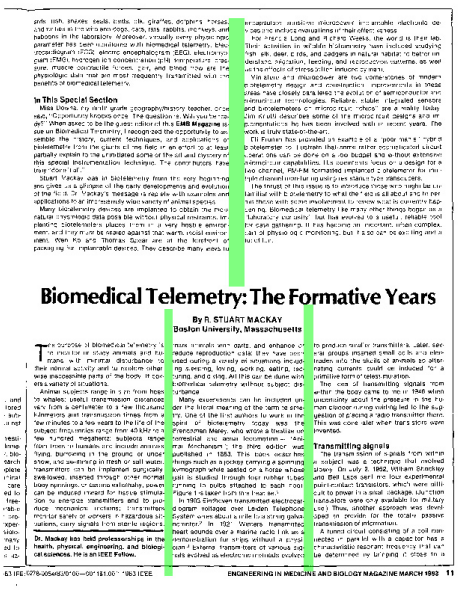
Abstracts on CD-ROM now an option

Build your own nucleus with a mouse

Workstations, workstations, workstations

Parallel Unix on 88100s

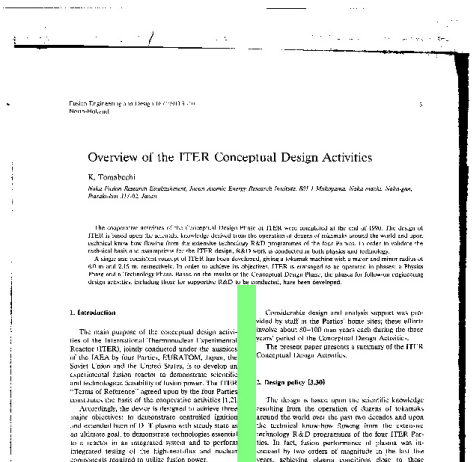
Interactive music lessons



Biomedical Telemetry: The Formative Years

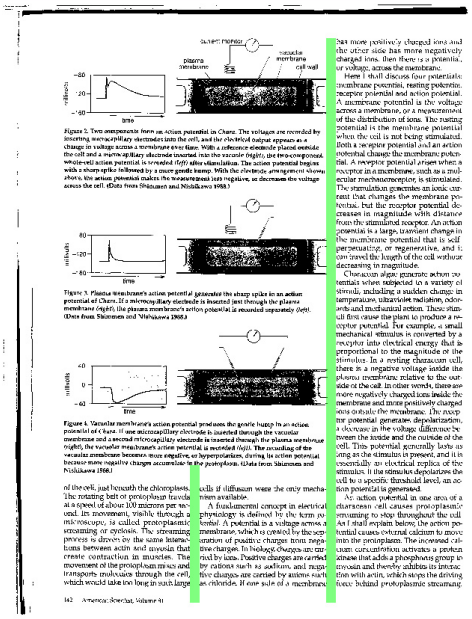
By R. STUART MACKAY

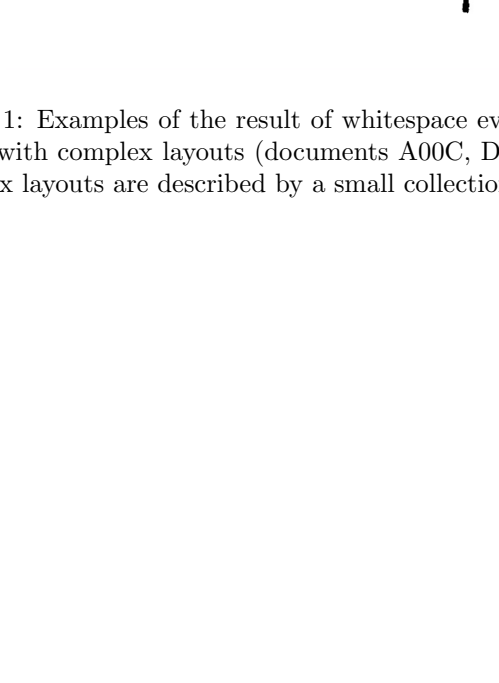
Boston University, Massachusetts



Overview of the ITER Conceptual Design Activities

K. Tomabechi





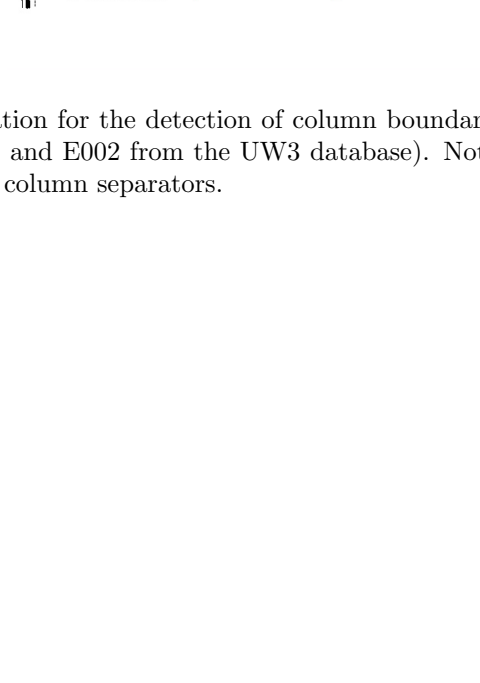


Figure 1: Examples of the result of whitespace evaluation for the detection of column boundaries in documents with complex layouts (documents A00C, D050, and E002 from the UW3 database). Note that even complex layouts are described by a small collection of column separators.

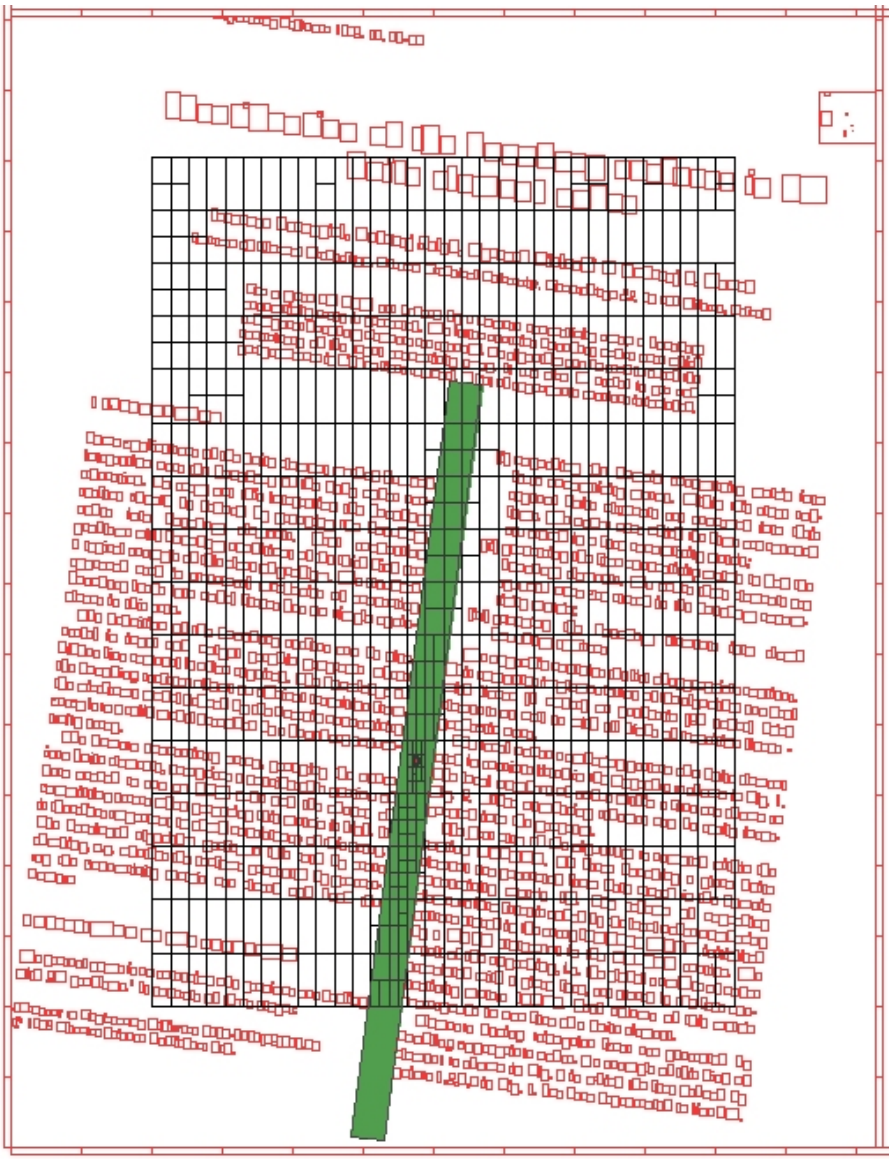


Figure 2: Globally optimal whitespace finding using non-axis aligned rectangles. The red rectangles in the background are bounding boxes for connected components in a scanned document. The large, non-aligned slender rectangle in the center is the column boundary found by the algorithm. Overlayed is a grid showing the hierarchical exploration of the parameter space carried out by the algorithm.

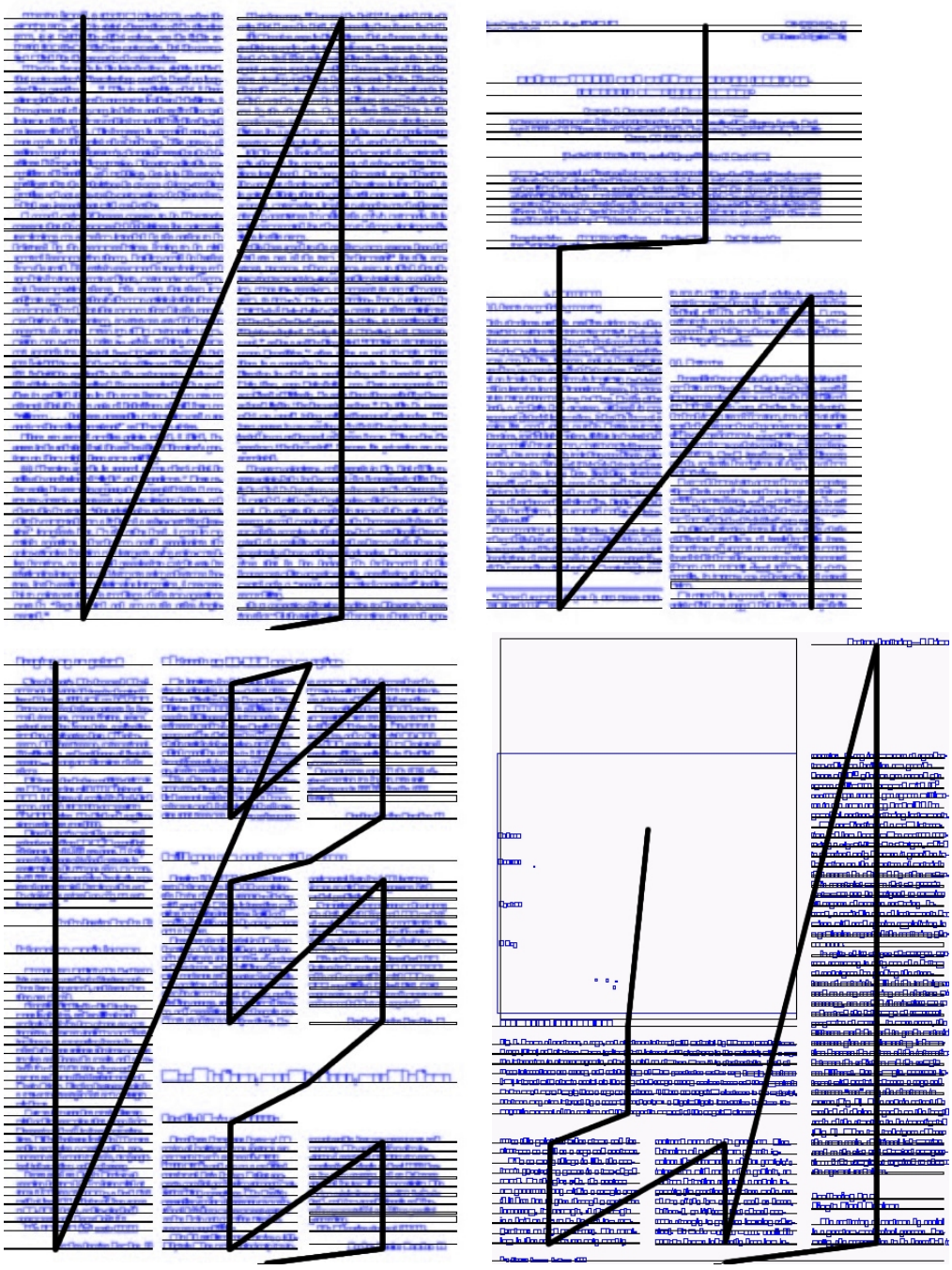
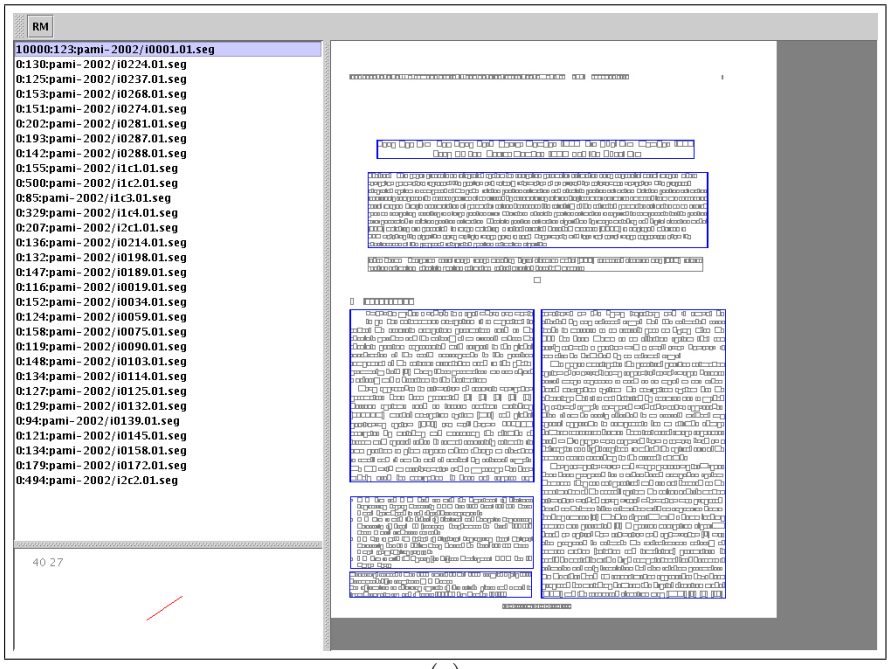
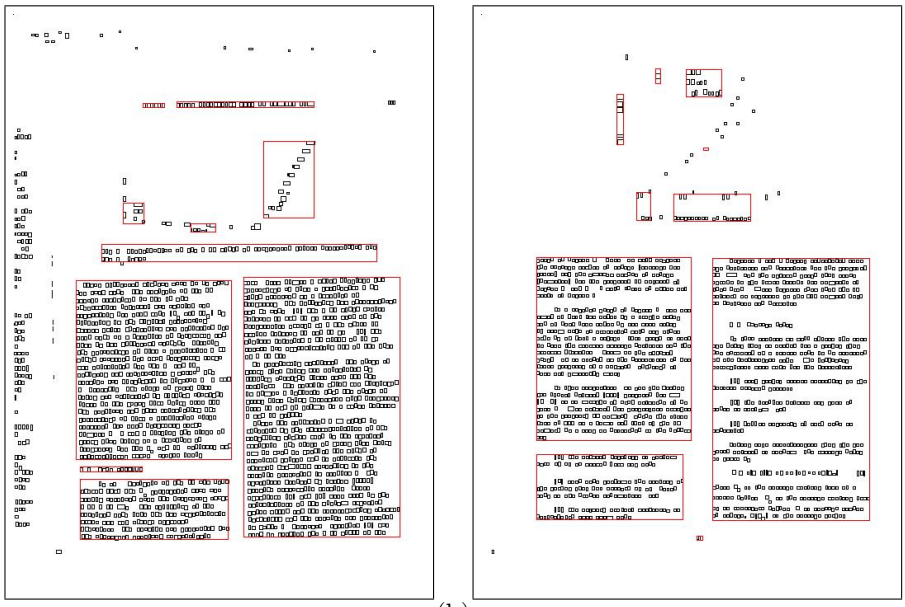


Figure 3: Recovering reading order by topological sorting; the thin black horizontal lines indicate text line segments, and the thick black lines running down and diagonally across the image indicate reading order; they connect the center of each text line with the center of the text line immediately following it in reading order. Text lines used are the lines as returned by the constrained text line finder; that is, text lines extend all the way across each column, even if actual characters only fill the line partially. Note that floating elements like headers and footers were not removed prior to determination of reading order and simply appear at some point inside the reading order (see the text for details).



(a)



(b)

Figure 4: Scale-space layout analysis and appearance based document retrieval. (a) Interactive GUI permitting fast manually supervised segmentation of document layouts using hierarchical layout analysis. By permitting the user to select interactively, with visual feedback, a scale at which the page is to be segmented, most documents can be segmented within a few seconds. This greatly speeds up manual layout analysis relative to approaches that require the user to mark each layout block individually. (b) An example of appearance-based retrieval. The database consists of 751 documents from the UW-1 collection. The query document is shown on the left and its closest match in the database is shown on the right. The appearance-based retrieval system adjusts the scale at which the document is segmented automatically while documents are being matched. This greatly improves the accuracy of appearance based retrieval and reduces the need for manual corrections to layouts.

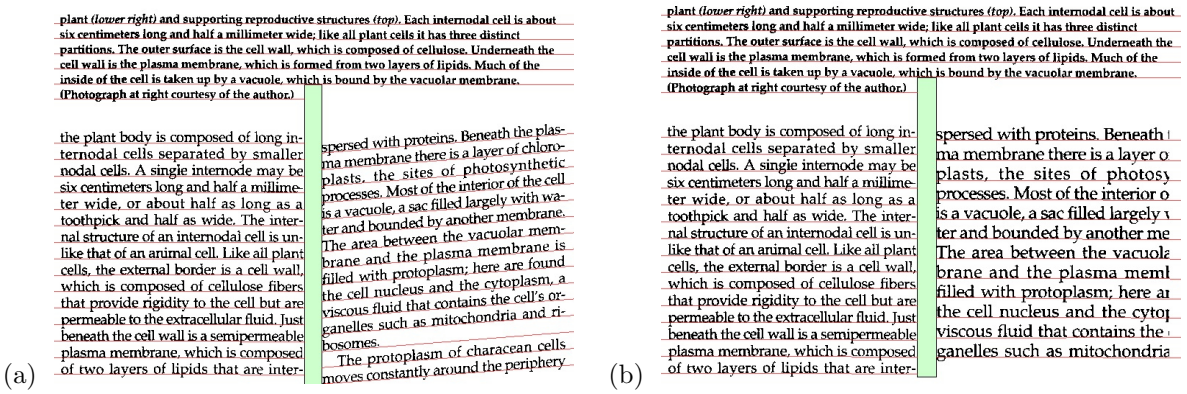


Figure 5: Application of the constrained line finding algorithm to simulated variants of a page. Gutters (obstacles) were found automatically using the algorithm described in the paper and are shown in green. Text lines were found using the constrained line finder and are shown in faint red. (a) Two neighboring columns have different orientations (this often occurs on the two sides of a spine of a scanned book). (b) Two neighboring columns have different font sizes and, as a result, the baselines do not line up.

2 Fast and Simple Maximum Empty Rectangles

A key step in document layout analysis is the determination of the structure of the page background. This structure allows us to identify major page layout features like columns and sections.

Background structure analysis as an approach to document layout analysis has been described by a number of authors [1, 12]. The work by Baird *et al.* [2] analyzes background structure in terms of rectangular covers, a computationally convenient and compact representation of the background. However, past algorithms for computing such rectangular covers have been fairly difficult to implement, requiring a number of geometric data structures and dealing with special cases that arise during the sweep (Baird, personal communication). This has probably limited the widespread adoption of such methods despite the attractive properties that rectangular covers possess. Our new algorithm requires no geometric data structures to be implemented and no special cases to be considered; it can be expressed in less than 100 lines of Java code. In contrast to previous methods, it also returns solutions in best-first order.

We define the maximal white rectangle problem as follows. Assume that we are given a collection of rectangles $C = \{r_0, \ldots, r_n\}$ in the plane, all contained within some given bounding rectangle r_b. In layout analysis, the r_i will usually correspond to the bounding boxes of connected components on the page, and the overall bounding rectangle r_b will represent the whole page. Also, assume that we are given an evaluation function for rectangles $Q : \mathbb{R}^4 \to \mathbb{R}$; in the simplest case, this will simply be the area, although we will consider more complex evaluation functions in the next section. The maximal white rectangle problem is to find a rectangle $\hat{r} \subseteq r_b$ that maximizes $Q(T)$ among all the possible rectangles $r \subseteq r_b$, where r overlaps none of the rectangles in C. The key idea behind the algorithm is similar to quicksort or branch-and-bound methods; details can be found in the references [3].

An application of this algorithm for finding a greedy covering of a document from the UW3 database with maximal empty rectangles is shown in Figure 1. Computation times for commonly occurring parameter settings using a C++ implementation of the algorithm on a 400MHz laptop are under a second. As it is, this algorithm could be used as a drop-in replacement for the whitespace cover algorithm used by [11], and it should be useful to anyone interested in implementing that kind of page segmentation system. However, below, this paper describes an alternative use of the algorithm that uses different evaluation criteria.

3 Improved Whitespace Evaluation

As we can see in Figure 6, merely computing an optimal whitespace cover does not necessarily yield a meaningful analysis of the page background. The approach by [1] addresses this problem by constructing an evaluation function Q of candidate whitespace rectangles based on their size and aspect ratio. However, that information by itself is not very reliable for distinguishing meaningful whitespace components from meaningless ones.

We have developed a simple set of evaluation criteria that identifies meaningful whitespace with an estimated error rate of less than 0.5% on the UW3 database with a single set of parameters. The idea is that for layout whitespace to be meaningful, it should separate text. Therefore, we require rectangles returned by the whitespace analysis algorithm

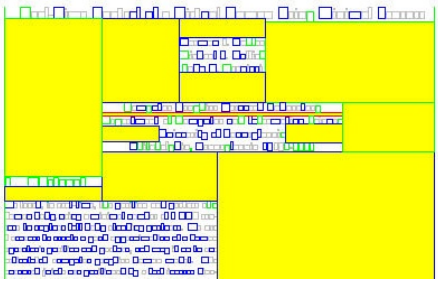


Figure 6: Partial, greedy whitespace cover for a portion of a document, computed using the algorithm described in the paper.

to be bounded by at least some minimum number of connected components on each of its major sides. This essentially eliminates false positive matches and makes the algorithm nearly independent of other parameters (such as preferred aspect ratios).

A gallery of automatically determined whitespace components is shown in Figure 1. The quality function was chosen such that only "tall" whitespace was returned. The resulting whitespace rectangles found by the algorithm correspond exactly to whitespace separating text columns.

4 Non-Axis Aligned Whitespace Rectangles

The algorithms described in the previous section for analyzing page background find axis-aligned whitespace. This section presents an algorithm that finds globally maximal whitespace rectangles on page images at arbitrary orientations. This algorithm eliminates the need for page rotation correction prior to background analysis. That has some advantages for complex documents containing text at multiple orientations in different columns, as well as for significantly degraded documents, for which determination of page rotation (skew) prior to analyzing background structure may be difficult.

The algorithm is resolution independent and takes as input a list of foreground shapes (e.g., character or word bounding boxes or polygons) and a set of parameter ranges; it outputs the N largest non-overlapping maximal whitespace rectangles whose parameters (location, width, height, orientation) fall within the required parameter ranges. It is somewhat analogous to the method described in the previous section, in that it also uses a branch-and-bound algorithm. Details of the algorithm will be described elsewhere [4].

5 Textline Finding

Another essential aspect of document layout is the lines of text. Identifying lines of text reliably is necessary in order to perform OCR. Reliable identification of lines of text also permits the detection of important page layout features such as paragraphs, section headings, etc.

Most past methods to text line finding have either been entirely global (projection methods, Hough transform methods, etc.), or very local (connected component linking, etc.), or they have required a complete page layout analysis as input prior to being able to identify text lines reliably. Global methods have serious problems with multi-column layouts or facing pages in scanned books, in which the orientation or spacing of neighboring text lines may differ significantly (Figure 5 shows two sample instances, plus a successful solution using the approach developed in this section).

We have developed a new approach to textline finding. It combines the advantages of previous approaches without their disadvantages. Like global methods, our approach can take advantage of long-range alignments of textual components, but the method is robust in the presence of multiple columns, like local methods.

The idea is to take the column boundaries identified by the background analysis described in the previous sections (shown in green in Figures 1 and 5) and introduce them as "obstacles" into a statistically robust, least square, globally optimal text line detection algorithm we have previously developed [6]. This match score corresponds to a maximum likelihood match in the presence of Gaussian error on location and in the presence of a uniform background of noise features, as shown in the literature [10].

Perhaps surprisingly, incorporating obstacles into the branch-and-bound textline finding algorithm is simple and does not noticeably increase the complexity of the algorithm on problems usually encountered in practice. This is because of a computational trick we have developed that greatly reduces the dimensionality of the parameter space that needs to be searched. Details can be found in [3]. An evaluation on the UW3 database shows nearly perfect detection of text lines. When used for page skew detection and correction, the method finds estimates of page skews that are within the variability of line orientations within a single page (less than 0.2 degrees on the UW3 database).

6 Reading Order by Topological Sort

As we noted above, recovery of reading order is a hard problem and can depend not only on the geometric layout of a document, but also on linguistic and semantic content. The key idea behind our approach is to determine all the pairwise constraints on reading order that we can, from the geometric

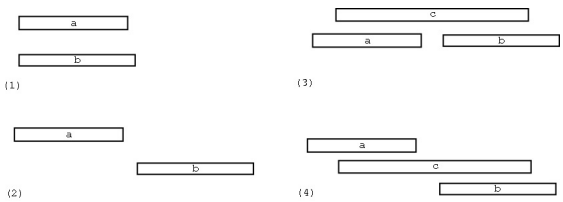


Figure 7: Figure illustrating the partial order criteria. In Example (1), segment a comes before segment b by criterion one. In Example (2), segment a does not come before b by criterion one because their x-ranges do not overlap (but criterion two may apply). In Example (3), segment a comes before segment b by criterion two: a is completely to the left of b, and there is no intervening line segment c. In Example (4), segment a does not come before b by criterion two because segment c separates them.

arrangement of text line segments on the page, as well as possibly linguistic relations. This partial order is then extended to a total order of all elements using a topological sorting algorithm [8].

Applying only two ordering criteria turns out to be sufficient to define partial orders suitable for determining reading order in a wide variety of documents:

1. Line segment a comes before line segment b if their ranges of x-coordinates overlap and if line segment a is above line segment b on the page.

2. Line segment a comes before line segment b if a is entirely to the left of b and if there does not exist a line segment c whose y-coordinates are between a and b and whose range of x-coordinates overlaps both a and b.

These criteria are illustrated in Figure 7. The first criterion basically ensures that line segments are ordered within their own column. The second criterion orders columns from left to right, but only for columns that fall under a "common heading".

Examples of recovered reading order using this approach are shown in Figure 3. Note that floating elements, like page headers, footers, and captioned images, were not removed prior to reading order determination. The output of reading order determination therefore contains these floating elements somewhere, usually close to where element is logically referenced. This is acceptable in applications like image-based document reflowing [5]. Otherwise, the floating elements can be removed prior to, or after, reading order determination.

An informal visual inspection of results on documents from the UW3 database suggest that reading order is recovered correctly using this approach in most cases; failures appeared to occur only when the source image contained severely degraded text areas (e.g., due to poor quality photocopies), or occasionally due to incorrectly determined bounding boxes for images appearing in the text.

7 A Novel Layout Analysis System

With the algorithms described in the previous sections, we can now put together a novel approach to layout analysis, consisting of the following steps:

1. Find tall whitespace rectangles and evaluate them as candidates for gutters, column separators, etc.

2. Find text lines that respect the columnar structure of the document.

3. Identify vertical layout structure (titles, headings, paragraphs) based on the relationship (indentation, size, spacing, etc.) and content (font size and style etc.) of adjacent text lines

4. Determine reading order using both geometric and linguistic information.

To evaluate the performance, the method was applied to the 221 document pages in the "A" and "C" classes of the UW3 database. Among these are 73 pages with multiple columns. The input to the method consisted of word bounding boxes corresponding to the document images. After detection of whitespace rectangles representing the gutters, lines were extracted using the constrained line finding algorithm. The results were then displayed, overlayed with the ground truth, and visually inspected. Inspection showed no segmentation errors on the dataset. That is, no whitespace rectangle returned by the method split any line belonging to the same zone (a line was considered "split" if the whitespace rectangle intersected the baseline of the line), and all lines that were part of separate zones were separated by some whitespace rectangle. Sample segmentations achieved with this method are shown in Figure 1.

8 Scale-Space Layout Analysis

The previous sections have dealt with layout analysis in terms of background whitespace structure, text lines, and reading order. In this section, I briefly describe some work in our lab on scale-space document layout analysis.

Generally, layout analysis methods depend on a number of segmentation parameters, such as the width and height at which whitespace is considered sufficiently salient in order to be considered a layout component. The core idea behind scale space layout analysis is to find a representation of the document layout that makes it possible to perform operations

like matching or retrieval across all scales simultaneously, and to generate segmentations with specific segmentation parameters very quickly. For details, the reader is referrered to the references [3]. Here, we will limit ourselves briefly to the applications of such methods. And example of this is shown in Figure 4(a).

Fast Interactive Segmentations Even with the best automatic layout analysis methods, occasionally, layouts need to be created and/or corrected manually. Scale-space layout analysis permits document layout analysis at interactive rates; that is, a user can use GUI controls to modify layout parameters and instantly watch the response. Many documents can be segmented in this way with a simple sweep of the mouse, selecting the horizontal and vertical segmentation thresholds. This permits interactive layout analysis within a few seconds per page.

Layout Based Retrieval Retrieval of documents from document databases based on their physical or logical layout has been described, for example, by Doermann *et al.* [9]. The idea is to first perform a layout analysis of the documents in the database and the query document and then to compare the layouts for the purposes of retrieval. A common problem in layout-based document retrieval occurs when similar documents are segmented slightly differently– the separation of text blocks in one document may fall slightly below or above the segmentation thresholds. Such a document may match a query document poorly, even though a slightly different choice of segmentation parameters might produce a nearly perfect match.

Scale-space layout analysis addresses this problem by not representing documents at a single scale. Instead, when a query document is compared against a document in the database, they are matched with all possible segmentation parameters, and the optimal set of segmentation parameters is selected as part of the matching process. Thereby, problems where slight changes in segmentation parameters cause the quality of match to change significantly are eliminated. And example of this is shown in Figure 4(b).

Segmentation by Example. Many tasks that use document layout analysis are performed on large databases containing pages with similar layouts. Examples are legacy conversions of company memos, patent documents, scientific journals, and medical data sheets. The ability to match page layouts more reliably using scale space segmentation makes it possible to perform segmentation by example. That is, a user adjusts the segmentation parameters of a sample document as needed, and the system adjusts the segmentation parameters on novel documents to match those on the sample document.

9 Conclusions

This summary paper has described a number of novel algorithms and statistical methods for layout analysis. A combination of globally optimal geometric algorithms with sound statistical models and careful engineering has permitted us to create a new generation of flexible page layout analysis algorithms. This combination yields demonstrably highly robust and versatile layout analysis on documents from the University of Washington Database 3 (UW3). It also holds promise for robust layout analysis in lantuages using non-Latin writing systems.

The system has been used for building electronic book (e-book) and document conversion applications. We are interested in finding commercial or government partners willing to sponsor the evaluation and adaptation of our software and methods to non-Latin writing systems (Arabic, Chinese, minority languages), and information retrieval and intelligence applications.

References

[1] H. S. Baird. Background structure in document images. In *H. Bunke, P. S. P. Wang, & H. S. Baird (Eds.), Document Image Analysis, World Scientific, Singapore*, pages 17–34, 1994.

[2] H. S. Baird, S. E. Jones, and S. J. Fortune. Image segmentation by shape-directed covers. In *Proceedings of the Tenth International Conference on Pattern Recognition, Atlantic City, New Jersey*, pages 820–825, 1990.

[3] T. M. Breuel. Two algorithms for geometric layout analysis. In *Proceedings of the Workshop on Document Analysis Systems, Princeton, NJ, USA*, 2002.

[4] T. M. Breuel. An algorithm for finding maximal whitespace rectangles at arbitrary orientations. (under review), 2003.

[5] T. M. Breuel, W. C. Janssen, K. Popat, and H. S. Baird. Paper-to-pda. In *Proceedings of the International Conference on Pattern Recognition (ICPR'02), Quebec City, Quebec, Canada*, 2002.

[6] T.M. Breuel. Robust least square baseline finding using a branch and bound algorithm. In *Proceedings of the SPIE - The International Society for Optical Engineering*, page (in press), 2002.

[7] R. Cattoni, T. Coianiz, S. Messelodi, and C. M. Modena. Geometric layout analysis techniques for document image understanding: a review. Technical report, IRST, Trento, Italy, 1998.

[8] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. MIT Press, Cambridge, MA, 1990.

[9] D. Doermann, C. Shin, A. Rosenfeld, H. Kauniskangas, J. Sauvola, and M. Pietikainen. The development of a general framework for intelligent document image retrieval. In *Document Analysis Systems*, pages 605–632, 1996.

[10] William Wells III. Statistical approaches to feature-based object recognition. *International Journal of Computer Vision*, 21(1/2):63–98, 1997.

[11] D. Ittner and H. Baird. Language-free layout analysis, 1993.

[12] K. Kise, A. Sato, and M. Iwata. Segmentation of page images using the area voronoi diagram. *Computer Vision and Image Understanding*, 70(3):370–82, June 1998.