Document Structure and Layout Analysis

Chapter · March 2007

DOI: 10.1007/978-1-84628-726-8_2

2 authors, including:

Anoop M. Namboodiri
International Institute of Information Technology, Hyderabad
108 PUBLICATIONS **1,560** CITATIONS

SEE PROFILE

Document Structure and Layout Analysis

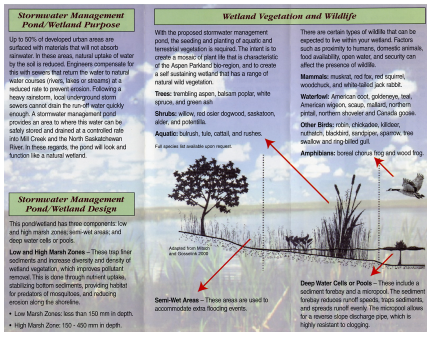
Anoop M. Namboodiri[1] and Anil K. Jain[2]

[1] International Institute of Information Technology, Hyderabad, 500 019, India,
[2] Michigan State University, East Lansing, MI - 48824, USA,
anoop@iiit.ac.in jain@cse.msu.edu

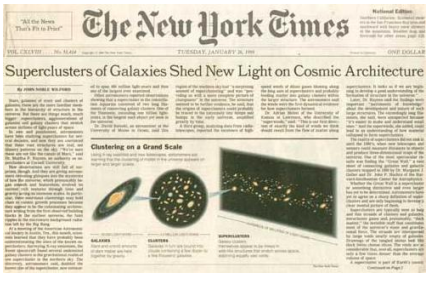
1 Introduction

A document image is composed of a variety of physical entities or regions such as text blocks, lines, words, figures, tables, and background. We could also assign functional or logical labels such as sentences, titles, captions, author names, and addresses to some of these regions. The process of *document structure and layout analysis* tries to decompose a given document image into its component regions and understand their functional roles and relationships. The processing is carried out in multiple steps, such as pre-processing, page decomposition, structure understanding, etc. We will look into each of these steps in detail in the following sections.

Document images are often generated from physical documents by digitization using scanners or digital cameras. Many documents, such as newspapers, magazines and brochures, contain very complex layout due to the placement of figures, titles, and captions, complex backgrounds, artistic text formatting, etc. (see Figure 1). A human reader uses a variety of additional cues such as context, conventions and information about language/script, along with a complex reasoning process to decipher the contents of a document. Automatic analysis of an arbitrary document with complex layout is an extremely difficult task and is beyond the capabilities of the state-of-the-art document structure and layout analysis systems. This is interesting since documents are designed to be effective and clear to human interpretation unlike natural images.



(a)



(b)

Fig. 1. Examples of document images with complex layouts.

As mentioned before, we distinguish between the physical layout of a document and its logical structure [4]. One could also divide the document analysis process into two parts accordingly.

1.1 Physical Layout and Logical Structure

The *physical layout* of a document refers to the physical location and boundaries of various regions in the document image. The process of *Document Layout Analysis* aims to decompose a document image into a hierarchy of homogenous regions, such as figures, background, text blocks, text lines, words, characters, etc. The algorithms for layout analysis could be classified primarily into two groups depending on their approach. Bottom-up algorithms start with the smallest components of a document (pixels or connected components) and repeatedly group them to form larger, homogenous, regions. In contrast, top-down algorithms start with the complete document image and divide it repeatedly to form smaller and smaller regions. Each approach has its own advantage and they work well in specific situations. In addition, one could also employ a hybrid approach that uses a combination of top-down and bottom-up strategies.

In addition to the physical layout, documents contain additional information about its contents, such as titles, paragraphs, captions, etc. Such labels are logical or functional in nature as opposed to the structural labels of regions assigned by layout analysis. Most documents also contain the notion of *reading order*, which is a sequencing of the textual contents that makes comprehension of the document, easier. Languages such as Arabic, Chinese, etc. can have different reading directions as well (right-to-left, top-to-bottom). The set of logical or functional entities in a document, along with their inter-relationships is referred to as the *Logical Structure* of the document. The analysis of logical structure of a document is usually performed on the results of the layout analysis stage. However, in many complex documents, layout analysis would require some of the logical information about the regions to perform correct segmentation.

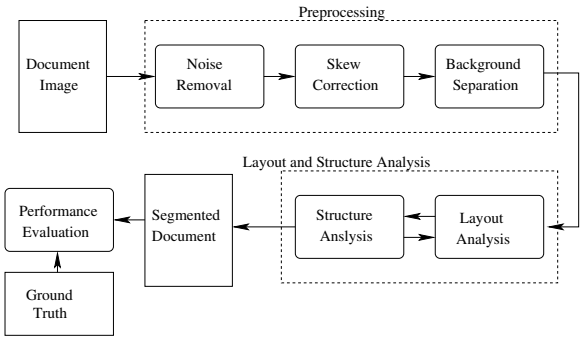


Fig. 2. Schematic diagram of a document layout and structure analysis system.

Most document images contain noises and artifacts that are introduced during the document generation or scanning phase. In order to make the analysis algorithms more

robust to this noise, the layout and structure analysis is usually preceded by a pre-processing stage. The pre-processing stage consists of tasks such as noise removal, background separation, skew detection and correction, etc. Figure 2 shows a schematic diagram of a document layout and structure analysis system. We will discuss each of the modules in detail in the remainder of this chapter.

2 Preprocessing

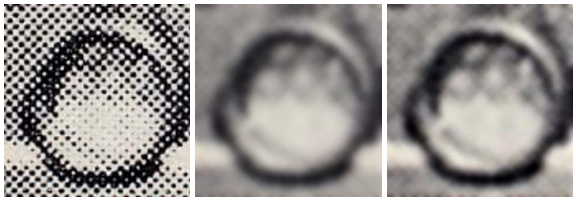
The preprocessing stage in document understanding primarily involves the following processes: i) removal of noise and other artifacts that are introduced during the scanning phase and during the production of the document, ii) separation of background regions of the document from the foreground, and iii) correction of skew that is introduced in the document image during scanning/acquisition. We will look into each of these problems and present commonly used techniques in this section. Some of these pre-processing algorithms could also be implemented as part of other modules during layout and structure analysis. However, pre-processing algorithms that are more specific to other modules of document understanding, such as character recognition are not studied here. Such methods might include binarization of a gray-level image and scaling of characters to a standard size.

2.1 Noise Removal

Noise is a common problem in most of the image understanding problems. These involve white noise that is introduced by interferences in the sensors and amplification circuits of the digitization mechanism (scanner, camera). The common sources of noise include white noise, salt and pepper noise, quantization artifacts, etc. These are well known noise sources that are compensated for by using techniques such as median filtering, dithering, low pass filtering, etc. In this section, we will look at one of the artifacts that is specific to document images, called halftones.

Dealing with Halftones: Halftoning is the process by which documents with smooth gray level or color changes are printed using fewer gray levels or colors. This is achieved by placing dots of various sizes, depending on the desired tone, at different parts of the image (see figure 3(a)). The method is also used in color documents when one would like to create continuously varying colors with few-color printing. This artifact becomes apparent especially in high resolution scans, which are very common with modern day sensors. There exist a variety of methods for halftoning depending on the resolution and gray levels available in the printing process [2]. Many algorithms that are designed for processing gray level images breaks down in such situations.

One of the most commonly used solutions to overcome this problem is to develop algorithms to detect half toned areas from a model of the halftoning process. The detected regions are either processed separately or converted to continuous gray scale using an appropriate low-pass filtering, which is often followed by application of a sharpening filter to reduce the blur (see figures 3(b) and 3(c)).



(a) (b) (c)

Fig. 3. Halftoning: (a) an image printed using halftoning, (b) result of applying gaussian blur to (a), and (c) result of sharpening (b).

2.2 Foreground Detection

One of the important problems in document structure understanding is that of separating the foreground from the background image. The problem is relatively simple in case of many documents that have a white or plain background. Even in such documents, determining the exact pixels that belong to the foreground is a challenging problem due to sampling and quantization of slant edges and lines. One could take two different approaches to solve this problem. We could assign every pixel to either foreground or background based on a specific criterion, such as a threshold, and assume that the later processing algorithms will accommodate the errors made during foreground detection. A second approach is to assign a probability measure or a fuzzy membership to each pixel for the foreground and background classes. This uncertainty is propagated to the following stages, which will utilize the context to arrive at a final decision. In most applications, the first approach yields useful results and hence a hard classification is used due to its simplicity.

The problem of foreground detection is much more challenging in the case of documents with complex backgrounds (see figure 1(a)). A variety of approaches have been used to decide whether a particular pixel belongs to the foreground or background. A common approach is to compute statistical or spectral properties of image patches and assign them as foreground or background using a classifier trained on labeled samples. Statistical features used for this purpose include the variance and moment features computed in a local neighborhood. Spectral features such as the responses from a bank of Gabor filters have been used with good success for this purpose [9]. A third approach is to train a classifier such as a neural network that directly accepts the image values in a neighborhood of a pixel region as input for its classification. Figure 4 shows a sample document with complex background and color reversed text, along with the foreground and background regions.

The problem of foreground detection could be posed as that of detecting text, since many documents contain only text in the foreground. One of the problems that many text detection algorithms find difficult to handle is that of reverse color text, i.e., text of light color in a dark background. One of the approaches to overcome this difficulty is to run an edge detection algorithm on the document image and use the properties of the edges to determine whether a pixel is part of text (foreground) or not (background).

(a) (b)

Fig. 4. (a) a document image with complex background, (b) foreground and background separated from (a).

These algorithms use properties of character edges such as the presence of parallel lines and uniformity of color between them for classification.

2.3 Skew Correction

Skew is introduced in a document image when a document is scanned or imaged at an angle with respect to the reference axes (see figure 5). The problem of skew correction plays an important role in the effectiveness of many document analysis algorithms, such as text line estimation, region boundary detection, etc. For example, algorithms based on projection profiles assume an axis-aligned scan. The primary challenge in skew correction is the estimation of the exact skew angle of a document image.

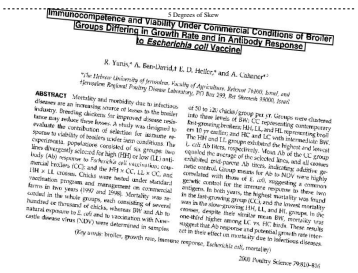


Fig. 5. Example of a document image with a skew of 5 degrees.

A variety of techniques are used for the detection of skew. Most of them assume the presence of some text component in the document and estimate the orientation of text lines using different methods. The commonly used techniques are projection profiles and line fitting. In the projection profile based approach, we compute the histogram of text pixels on the vertical axis (vertical projection) and try to maximize the distance between consecutive line profiles, called *line spacing*, while rotating the document image.

We will examine the projection profile based method in the next section. One could also estimate the skew of a document by fitting a line to the centroids of connected components within a line and measuring the angle with respect to the horizontal axis, which can then be corrected.

3 Representing Document Structure and Layout

A critical component in facilitating the understanding of document structure and layout is its representation in memory. An appropriate representation can encapsulate the prior knowledge about the documents being processed. In addition, the representation can also help to organize the extracted information and guide the analysis algorithm.

A document contains multiple regions that are spatially and logically related to each other. Such a model is captured well in the formalism of a *graph*. A graph $\mathcal{G}$ is defined as a set of *vertices*, $\mathcal{V}$, and a set of *edges*, $\mathcal{E}$. Each edge, $e \in \mathcal{E}$, is an ordered pair of vertices, (v_1, v_2), where $v_1, v_2 \in \mathcal{V}$. The regions in a document image correspond to the vertices or nodes in a graph, while a relationship between two regions is denoted by an edge between the corresponding nodes. For a document we need to specify the relationship between two regions, rather than just stating that there exists one. For this purpose, we will associate an attribute with each edge that specifies the nature of the relationship. The vertices also have attributes that specify the properties of a region, such as the type, size, location, etc. Such graphs, where the vertices and edges have associated attributes are called *attributed graphs*.

The graph representation can encapsulate the relationship between regions and their properties. However, in many documents, the attributes are not conclusive during the analysis stage. One needs to handle the ambiguity and estimate the most likely set of values for the attributes. To facilitate this processing and to enable the graphs to model the uncertainties in the knowledge itself, one could describe the attributes in terms of probability distributions. Such graphs are referred to as *attributed random graphs*. In attributed random graphs, each attribute is defined by the parameters of a probability distribution (say Gaussian with specified mean and variance values). Special cases of attributed random graphs, such as First order Gaussian Graphs have been successfully used in modeling and classifying document images [7].

The graph representation, however, fails to capture the hierarchical nature of a document structure and layout. To accomplish this, one could use a rooted tree for representing document layout and logical structure [1]. The root of the tree will represent the complete document, while each node of the tree will represent a specific region within the document. The set of children of a node will form subregions that compose the parent region. For example, consider the business card recognition application. A child node of the root could represent the address field, and its children in turn could represent fields such as street, city, zip code, etc. A node representing the name field could be another child of the root, and hence a sibling of the address node. Such a representation incorporates the prior information about the logical structure of a business card. Similar to attributed graphs, a tree-based representation, where each node and edge is associated with a set of properties or attributes, is known as an attributed tree. The trees could also be implicitly represented in terms of a set of rules [11]. Probabilistic versions of

tree representations are quite useful to handle the uncertainties in real-life documents in structure and layout.

One of the most powerful ways to express hierarchical structures is to use formal grammars. The class of *regular* and *context free* grammars are extremely useful in describing the structure of most documents. A document is represented as a sequence of *terminal symbols*, T, which correspond to smallest units that we would like to represent in a document (pixels, character, regions, etc.). A formal grammar is defined as a set of rules that transform a starting symbol, corresponding to the complete document, into a set of terminal symbols, each representing a specific primitive. The primitives are the smallest units that one needs to address during the document structure and layout analysis. These could be units such as words, characters, pixels or connected components. Note that unlike the generic rule-based representation of trees that we mentioned before, formal grammars restrict the type of rules that can be used, which make their analysis and estimation more tractable. We also define a set of intermediate symbols, called non terminals, which correspond to the different regions in a document layout. The non-terminals could correspond to physical regions such as text regions, columns, paragraphs, lines, etc. or structural units such as headings, captions, footnotes, etc. The grammar transforms the start symbol into a string of non-terminals and terminals. Each non-terminal is in turn replaced with other non-terminals or terminal symbols, according to the rules defined by the grammar. This is repeated until all symbols in the string consists only of terminals.

$$
\begin{array}{l}
[0.5] < START > \rightarrow < TITLE > < COLUMN > < COLUMN > \\
[0.5] < START > \rightarrow < TITLE > < COLUMN > \\
[1.0] < TITLE > \rightarrow < text\_line > \\
[1.0] < COLUMN > \rightarrow < TEXT\_BLOCKS > \\
[0.8] < TEXT\_BLOCKS > \rightarrow < TEXT\_BLOCK > < space > < TEXT\_BLOCKS > \\
[0.2] < TEXT\_BLOCKS > \rightarrow < TEXT\_BLOCK > \\
[1.0] < TEXT\_BLOCK > \rightarrow < TEXT\_LINES > \\
[0.9] < TEXT\_LINES > \rightarrow < text\_line > < newline > < TEXT\_LINES > \\
[0.1] < TEXT\_LINES > \rightarrow < text\_line >
\end{array}
$$

(a)

$$
\begin{array}{l}
< START > \rightarrow < TITLE > < COLUMN > \\
\rightarrow < text\_line > < COLUMN > \\
\rightarrow < text\_line > < TEXT\_BLOCKS > \\
\rightarrow < text\_line > < TEXT\_BLOCK > < space > < TEXT\_BLOCKS > \\
\rightarrow < text\_line > < text\_line > < newline > < TEXT\_BLOCK > < space > < TEXT\_BLOCKS > \\
\rightarrow < text\_line > < text\_line > < newline > < text\_line > < space > < TEXT\_BLOCKS > \\
\rightarrow < text\_line > < text\_line > < newline > < text\_line > < space > < TEXT\_BLOCK > \\
\rightarrow < text\_line > < text\_line > < newline > < text\_line > < space > < text\_line >
\end{array}
$$

(b)

Fig. 6. Representing a document in terms of formal grammars: (a) example of a stochastic context free grammar that derives a text document with a title. Upper case symbols refer to non-terminal symbols, while lower case symbols show terminal symbols. (b) a sample document with a title and three lines, derived using the grammar in (a)

A specific interpretation of the layout or structure of a document will correspond to a specific sequence of grammar rules (productions) that generate the terminal symbols corresponding to the document. Such a sequence of productions is referred to as a derivation. However, there could be multiple derivations corresponding to a particular sequence of terminals. This would mean multiple interpretations of the structure or layout. The process of document analysis is hence to determine the correct derivation for a particular document. See figure 6 for an example of a grammar and the sequence of derivations that describe a specific document.

One could associate a probability measure to each of the grammar rules, which represent the probability that the rule would form a part of the interpretation of a document. Such a grammar, where there are probabilities associated with each production is referred to as a *stochastic grammar*. Stochastic grammars are especially useful to integrate multiple evidences and estimate the most probable parse or interpretation of a given document. One could also attach attributes to each of the nodes in the grammar, such as properties of a region. Such grammars are called *attributed grammars* and are commonly used for layout and structure analysis purposes.

4 Document Layout Analysis

The primary purpose of a document layout analysis module is to identify the various physical regions in a document and their characteristics. A *maximal region* in a document image is the maximal homogenous area of a document image. The property of homogeneity in the case of layout refers to the type of region, such as text block, image, graphic, text line, word, etc. These regions are not mutually exclusive and one region could contain other region types within it. Before proceeding further, we define a segmentation of an image as follows:

Definition 1. *A segmentation of an image is a set of mutually exclusive and collectively exhaustive subregions of the image. Given an image, I, a segmentation is defined as $S = \{R_1, R_2, \cdots, R_n\}$, such that, $R_1 \cup R_2 \cup \cdots \cup R_n = I$, and $R_i \cap R_j = \phi \; \forall i \neq j$.*

Based on the above observations one could define layout analysis as follows:

Definition 2. *The process of document layout analysis decomposes a document image into a hierarchy of maximally homogeneous regions, where each region is repeatedly segmented into maximal sub-regions of a specific type.*

For a document image, D, the layout is a sequence of segmentations, $S_1, S_2, ..S_n$, where S_1 is a segmentation of the complete document image, and S_i is a segmentation of one of the regions in $S_j; 1 \leq j < i$.

Note that the regions of each segmentation are also defined to be homogeneous in addition to being maximal. In fact, there exists a unique segmentation of a document image, given a specific definition of homogeneity of the regions. However, in practice, one could get multiple interpretations due to an ill defined homogeneity criterion. For example, two text blocks that are close to each other may be considered as a single block if the homogeneity measure used is not able to differentiate between them based on the spacing and alignment between the two blocks.

4.1 Approaches to Layout Analysis

Document layout analysis algorithms are primarily divided based on their order of processing into *top-down* approaches and *bottom-up* approaches. Top-down approaches start with the complete document image and repeatedly split them into smaller homogeneous regions. The splitting is stopped when all the resulting regions correspond to the primitives, based on which the document is to be described. The primitives could be pixels, connected components, words, etc., depending on the application. Conversely, bottom-up approaches start with the primitives and repeatedly group them into larger regions such as words, lines, text-blocks, columns, etc. There are also *hybrid* algorithms that combine the above two approaches. The primary challenges in any of the above approaches are:

- Region Segmentation: To split a document image into component regions, we need to decide where the homogeneity criterion is not satisfied. The homogeneity is often defined based on a parameter vector, $\boldsymbol{\theta}$, whose value plays a critical role in the success of the algorithm. This is also true for bottom-up approaches, when we try to group smaller components into regions. Computing a parameter vector, $\boldsymbol{\theta}$, that works well for all possible documents is often not possible. Current approaches try to tune the parameters based on feedback from later processing stages such as character recognition. The problem is also present at lower levels of segmentation such as line and word segmentation.
- Region Classification: The second challenge is to decide the type of a particular region, once it is segmented. Typical problems are in text versus image classification and identification of graphics and tables. Region classification is a prerequisite for further processing of many region types.

There are a variety of algorithms that has been proposed to perform layout analysis of documents. We now will look into three of the popular algorithms to illustrate the three approaches mentioned above.

Examples of Algorithms Typical top-down approaches proceed by dividing a document image into smaller regions using the horizontal and vertical projection profiles. The *X-Y Cut* algorithm [13], starts dividing a document image into sections based on valleys in their projection profiles (see figure 7). The algorithm repeatedly partitions the document by alternately projecting the regions of the current segmentation on the horizontal and vertical axes. The splitting is stopped when a particular criterion that decides the atomicity of a region is met. Projection profile based techniques are extremely sensitive to the skew of the document. A small amount of skew could completely change the nature of the projection, resulting in no clear valleys. Hence extreme care has to be taken while scanning of documents, or a reliable skew correction algorithm has to be applied before the segmentation process.

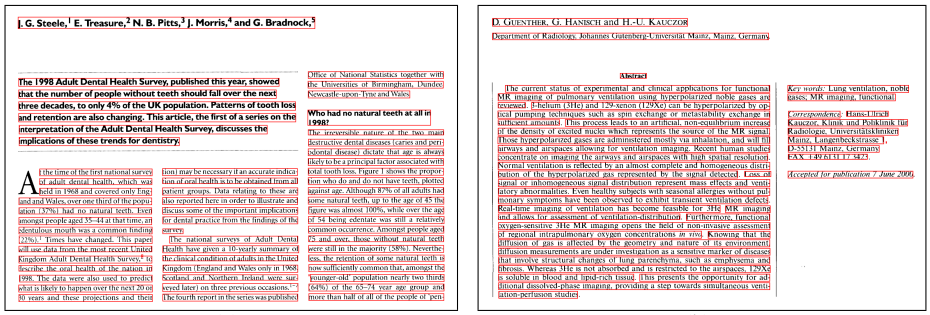
Efficient and accurate methods have been devised for documents with white background and a manhattan (rectangular) layout of the regions, based on geometric approaches that cover the background. Examples include the shape-directed cover algorithm by Baird et al. [17] and the white streams based segmentation by Pavlidis and

Fig. 7. Horizontal and vertical projection profiles of a document image.

Zhou [20]. These are top-down approaches that divide a document image into smaller regions based on the structure of the background. The primary idea is to detect a set of maximal rectangles that do not contain any foreground pixels. These rectangles divide the document image into regions that may contain text, images or graphics. A region classification algorithm could be applied to them before further processing. The algorithms mentioned above are very complex to implement due to the representation of intermediate results and many special cases that need to be handled. The whitespace cover algorithm by Breuel [19] is an efficient variation of the same idea using geometric principles. The algorithm is simpler due to the lack of special cases and performs very well for this class of documents.

Bottom-up approaches need to define primitive components to start the grouping process. The *Docstrum* algorithm by O'Gorman [15], proceeds by identifying a set of connected components in the document as the primitives. Large noise or non-text components are then removed from this set. K nearest neighbors of each component are identified and text lines are formed based on a threshold of the angle between the component centroids. Histograms of component spacings are used to identify inter-character distance within a word and between words. Transitive closures of neighboring components using the within-line-distance and between-line-distance are computed to form text lines and text blocks. The algorithm has a set of threshold parameters, which are set by experiments on a set of documents. The algorithm performs well for English language documents with a wide variety in layout. Figure 4.1 shows two examples of document images segmented using Docstrum. As we can see, the algorithm handles fonts of various sizes and styles well, even with relatively complex layouts.

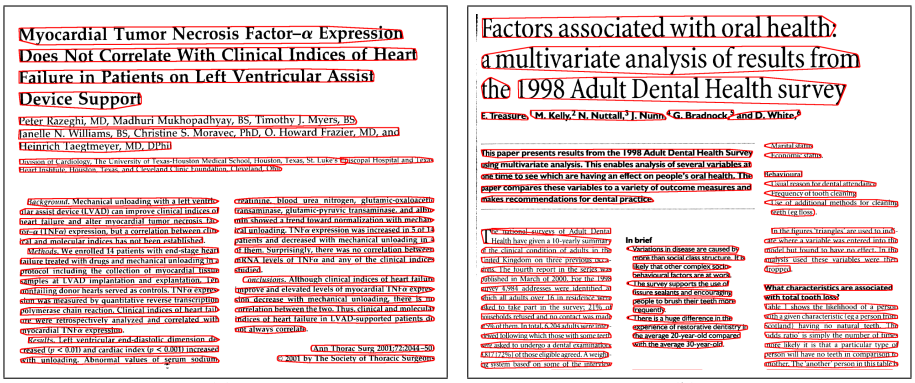
The Voronoi diagram based algorithm by Kise et al. [14] is another effective method that functions by grouping connected components in a page. We start with computing a Voronoi tessellation of the document image. The neighbor graph of connected components is computer from the Voronoi diagram. The algorithm then uses a threshold based on the area and distance between adjacent components in the Voronoi diagram to decide region boundaries. Text lines are identified based on a threshold on the inter-component



(a) (b)

Fig. 8. Typical examples of document images segmented using the *Docstrum* algorithm. Detected lines of text are shown in boxes.

distance computed from its distribution in the neighbor graph. The results are very reliable and highly accurate on documents with white background, where the connected components can be identified reliably. Figure 4.1 shows two examples of document images segmented using the Voronoi-based algorithm. As in *Docstrum*, the Voronoi-based algorithm is able to handle very complex layouts and performs well on most pages in English.



(a) (b)

Fig. 9. Typical examples of document images segmented using the *Voronoi* algorithm. Detected lines of text are shown with the boundaries of the voronoi cells.

The run-length smearing algorithm of Wahl et al. [21] is another example of bottom-up approach. Kopec and Chou [16] describe an algorithm for segmenting a column of text that is modeled using a generative Markov model. However, this model makes the assumption that the connected components are recognizable and belong to a specific vocabulary.

Pavlidis and Zhou [23] proposed a hybrid algorithm using a split-and-merge strategy. Here a top-down approach is used to arrive at an over-segmented document image. Regions that are similar and nearby are then merged to form homogeneous regions.

5 Understanding Document Structure

Structure understanding deals with logical descriptions of regions rather than their physical attributes.

Definition 3. *The logical structure of a document image is a mapping from the physical regions in the documents to their logical labels. Document structure analysis is the process of assigning the logical labels to physical regions identified during layout analysis.*

The logical labels include title, abstract, sub-title, paragraph, sentence, words, header, footer, caption, page number, etc. Even though the logical layout analysis process is defined to follow the layout analysis, in practice, the two processes could be combined into a single document understanding framework.

One of the popular approaches to define logical structure of a document is to treat the set of regions in a document as a string of symbols. A grammar is defined over these symbols that describes the logical structure of an arbitrary document under consideration. The process of structure analysis then computes the most likely parse (set of grammar rules) that generates the observed string of symbols. The problem arises when there are multiple parses corresponding to a given document and we have to decide which interpretation to choose. The grammar rules are augmented with attributes of regions and their relationships to accommodate this situation. The most likely parse could be defined using a cost function that takes into account, the attribute values. One could also use a stochastic grammar, that could give a likelihood value for a parse, depending on the rules used and the terminal attributes. The stochastic grammar rules also contain a probability associated with it, which can be used to determine the most likely parse, along with the region attributes.

Related methods use other representations of the logical structure, such as tree or graph representations that were discussed in section 3. The approach is similar to what we described for the grammar-based analysis. The domain knowledge is stored as a set of constraints on the structure of the tree or graph, instead of grammar rules. The problem is to identify the variation of the tree, according to the definition, that best matches the observed document [24].

Rule-based systems have also been proposed to determine the logical structure of a document. Note that these are different from the grammar rules that describe, formally, a set of derivations that are permissible. Rules in a rule-based system could denote actions to be taken in a particular situation or encode domain knowledge. The *DeLoS* system proposed by Niyogi and Srihari [26] uses three levels of rules that encode domain knowledge, control information, and strategies. The approach works well even for complex documents such as newspaper pages. However, the rules themselves are hard coded and rule based systems do not have the ability to learn from a set of sample documents.

A second approach is to apply domain knowledge about the layout and structure on the result of OCR and graphics recognition on a given document [25]. This approach assumes that the layout identification stage correctly identifies the text regions, which are then recognized by an OCR. The approach is very powerful, since it can make use of the contents of the text to decide its function. Note that this was not available in the previous methods. For example, the OCR-based approach can use the words in a text line near an image to decide whether it is a caption or not.

6 Performance Evaluation

There are many factors that affect the results of evaluation of a document segmentation algorithm. To perform evaluation, one needs to decide on the following points: i) A set of test documents, ii) ground truth of segmentation of the chosen documents, iii) an evaluation or performance metric that is unambiguous and meaningful, and iv) a set of statistics over the performance metric to summarize the results over the complete document set.

Algorithms could be compared against each other only when they are tested on a common dataset that has sufficient number and variety of documents. In addition, one should also agree on a common ground truth. This is often difficult as even human experts could assign different ground truths to a specific document. One needs to develop standardized datasets with ground truth to overcome these problems. The University of Washington datasets, the *NIST* document image dataset [30], etc. are among the most commonly used standards of datasets with ground truth that are used for performance evaluation. The data sets need to be updated and expanded or new ones created according to the future requirements. Details about some of the current datasets could be found at [31].

To evaluate the performance of a segmentation algorithm, we need to compare the result of segmentation against a ground truth. However, this is not a trivial task, since the result of a page segmentation algorithm is not a single mathematical quantity or model that can be compared with the ground truth to compute the error. General image segmentation algorithms output a set of regions for an image, which could be compared against the ground truth for overlap. However, document segmentation poses additional problems due to the hierarchical nature of segmentation. The approaches to define a performance metric could be primarily classified into two categories:

– Overlap-based performance metrics: This is similar to image segmentation evaluation measures. The metric is defined based on the overlap area of labelled regions againt regions of the same label in the ground truth. We could compute the overlap at the lowest level of the hierarchy or report results at multiple levels. The performance metric could be any function of the overlap area. For example, the accuracy, $\rho(G, S)$, could be measured as:

$$\rho(G, S) = \frac{O_I(G, S) - (M_I(G, S) + S_I(G, S))}{Area(G)}, \tag{1}$$

where I is the document image, G is the ground truth and S is the result of layout and structure analysis. The function O_I measures the overlap area of regions

between G and S, where the labels agree, $M_I()$ and $S_I()$ measure the areas of missing and spurious regions in S, compared to G, and $Area(G)$ is the total area of the regions in the image (or G).

– Split and Merge based metrics: Such metrics are defined based on the observation that most errors that are made by a document segmentation algorithms stem from splitting of uniform regions and merging of two different neighbouring regions. We measure the accuracy of a segmentation based on the number of split and merge operations that are required to transform the result of segmentation to the ground truth. We could also include missing and spurious regions into the metric. For example, Mao and Kanungo [27] defines the accuracy of text line detection, $\rho(G, S)$, as:

$$\rho(G, S) = \frac{\#L - \#\{C_L \cup S_L \cup M_L\}}{\#L}, \tag{2}$$

where $\#$ stand for the cardinality operator, L is the set of all text lines in the document, and C_L, S_L, and M_L represents sets of spurious, split and merged lines respectively.

Common statistical measures that are used for summarization of error results include the mean and variance of the accuracy over the complete test set. Cross validation accuracy is reported for algorithms that estimate parameters or learn from a dataset.

7 Handwritten Document Analysis

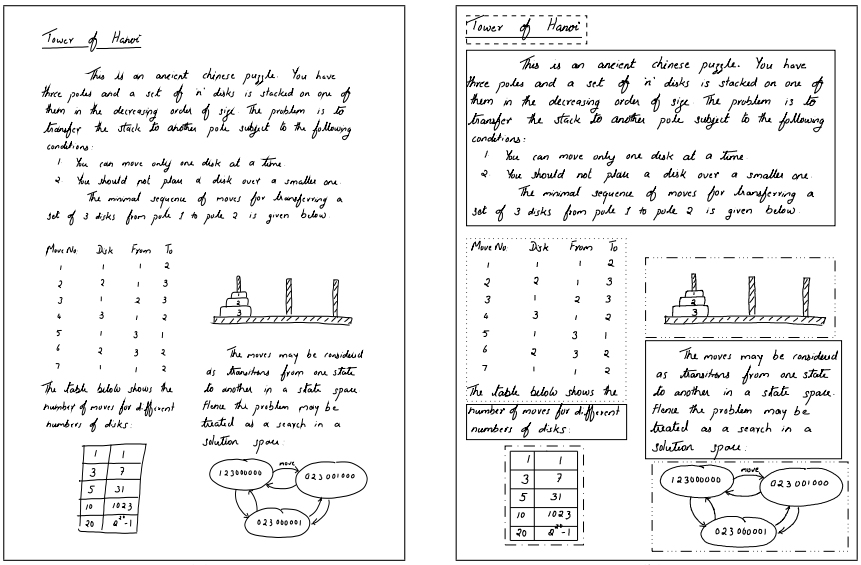
The problem of document understanding can be extended to the domain of handwritten documents as well. The problem is to decipher the physical layout and the logical structure of a handwritten page. The problem is much more complicated due to the lack of physical organization in most handwritten documents. Hence the problem has not been studied in a generic setting. However, handwritten document analysis has been extensively used in specific areas such as postal address recognition and recognition of hand filled forms. In the case of hand filled forms, the format of the form is assumed to be known. This helps the algorithms to remove the form background from the document image. In addition, the recognition is also facilitated in many form filling applications, since the characters are often well separated and printed.

One of the most impressive success stories of handwritten document understanding has been in the field of postal automation. The problem is to recognize the destination of an envelope for automatic routing of mails. The problem of postal address recognition will be discussed in detail, elsewhere in this book. A third successful application area of handwritten document analysis is that of recognition of mathematical equations.

Most of the successful systems that carry out handwritten document analysis have been domain specific with clear and simple document structures. A generic handwritten document analyzer is still beyond the reach of the state of the art in the field. We now look into a special case of handwritten documents that have received attention, namely online handwritten documents.

7.1 On-line Documents

Online documents are captured during the writing process using a digitizer such as special pens and pads, touch sensitive screens or vision-based pen tip trackers. They encode the dynamic information of the writing process in addition to the static physical layout. The data in online documents is stored as a sequence of strokes [3]. The temporal information regarding the stroke order helps us to reduce the two-dimensional segmentation problem into that of segmenting a linear sequence of strokes. This is true in most documents, where one does not move back and forth between multiple regions. Jain et al. [12] proposed a clustering based scheme, where the individual strokes are initially classified as belonging to text or non-text regions. The strokes are then clustered based on their spatial and temporal proximities onto various regions. Figure 10 shows an online document and the result of layout analysis from [12].



(a) (b)

Fig. 10. Segmentation of online documents:(a) an online document and (b) result of segmentation [12].

Online document understanding and handwriting recognition are popular due to the large number of handheld devices that accept pen input. Such data require segmentation algorithms without assuming strict models about the layout. Grammar-based techniques are very promising for this problem, especially those using stochastic formal grammars for modeling document regions [28, 29].

[3] A stroke is the trace of the pen between a pen-down and the following pen-up

8 Summary

Document layout and structure understanding often forms the basis on which many document understanding systems are built. One can reliably segment and decipher the structure of many document images with a regular structure and plain white background, such as technical journals, telephone books, text books, etc. However, a generic document layout and structure understanding system that can deal with complex backgrounds and layout is still an elusive goal. One such application that deals with a large variety of layouts is the document understanding/search module in a digital library. Learning-based approaches have the potential to adapt to such large variations in layout and structure.

A complete understanding of the structure of a document is possible only with information about the contents of the document. Hence it is important to develop an integrated approach to document layout and structure analysis and various recognition modules. Robust OCR techniques that will work on noisy and degraded documents will go a long way towards structure and layout understanding.

References

1. Nagy, G. and Seth, S.C.: Hierarchical Representation of Optically Scanned Documents. Proceedings of the $7th$ International Conference on Pattern Recognition, Montreal (1984) 347–349.
2. Ulichney, R.: Digital Halftoning. The MIT Press, Cambridge, Mass., (1987).
3. Fujisawa, H. and Nakano, Y. and Kurino, K.: Segmentation Methods for Character Recognition: From Segmentation to Document Structure Analysis. in Proceedings of the IEEE **80**, (1992) 1079–1092.
4. Haralick, R.M.: Document Image Understanding: Geometric and Logical Layout. in Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, Seattle, WA (1994) 385–390.
5. Jain, A.K. and Yu, B.: Document Representation and its Application to Page Decomposition. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**, (1998) 294–308.
6. Nagy G.: Twenty Years of Document Image Analysis in PAMI. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**, (2000) 38–62.
7. Bagdanov, A.D. and Worring, M.: First order Gaussian Graphs for Efficient Structure Classification. Pattern Recognition **36**, (2003) 1311–1324.
8. Etemad, K. and Doermann, D.S. and Chellappa, R.: Multiscale Document Page Segmentation using Soft Decision Integration. IEEE Transactions on Pattern Analysis and Machine Intelligence **19**, (1997) 92–96.
9. Jain, A.K. and Bhattacharjee, S.: Text Segmentation using Gabor Filters for Automatic Document Processing. Machine Vision and Applications **5**, (1992) 169–184.
10. Jain, A.K. and Zhong, Y.: Page Segmentation using Texture Analysis: Pattern Recognition **29**, (1996) 743–770.
11. Fisher, J.L.: Logical Structure Descriptions of Segmented Document Images. in Proceedings of International Conference on Document Analysis and Recognition, Saint-Malo, France (1991), 302-310.
12. Jain, A.K. and Namboodiri, A.M. and Subrahmonia, J.: Structure in Online Documents. in Proceedings of International Conference on Document Analysis and Recognition, Seattle, WA (2001), 844-848.

13. Nagy, G. and Seth, S. and Viswanathan, M.: A Prototype Document Image-Analysis System for Technical Journals. Computer **25**, (1992), 10–22.
14. Kise, K. and Sato, A. and Iwata, M.: Segmentation of Page Images using the Area Voronoi Diagram. Computer Vision and Image Understanding **70** (1998), 370–382.
15. O'Gorman, L.: The Document Spectrum for Page Layout Analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **15** (1993), 1162–1173.
16. Kopec G.E. and Chou, P.A.: Document Image Decoding using Markov Source Models. IEEE Transactions on Pattern Analysis and Machine Intelligence **16** (1994), 602–617.
17. Baird, H.S. and Jones, S.E. and Fortune, S.J.: Image Segmentation by Shape-Directed Covers. in Proceedings of International Conference on Pattern Recognition, Atlantic City, NJ (1990), 820–825.
18. Baird, H.S.: Background Structure in Document Images. Document Image Analysis, Word Scientific, Singapore (1994), 17–34.
19. Breuel, T.M.: Two Geometric Algorithms for Layout Analysis. in Proceedings of the Fifth International Workshop on Document Analysis Systems, Princeton, NY (2002), LNCS 2423, 188–199.
20. Pavlidis, T. and Zhou, J.: Page Segmentation by White Streams. Proceedings of International Conference on Document Analysis and Recognition, Saint-Malo, France (1991), 945-953.
21. Wahl, F. and Wong, K. and Casey, R.: Block Segmentation and Text Extraction in Mixed Text/Image Documents. Graphical Models and Image Processing **20** (1982), 375–390.
22. Wu, V. and Manmatha, R. and Riseman, E.M.: Finding Text in Images. ACM DL (1997), 3–12.
23. Pavlidis, T. and Zhou, J. Page Segmentation and Classifcation. Graphical Models and Image Processing **54** (1992), 484–496.
24. Yamashita, A. and Amano, T. and Takahashi, I. and Toyokawa, K.: A Model-based Layout Understanding Method for the Document Recognition System. in Proceedings of the International Conference on Document Analysis and Recognition, Saint-Malo, France (1991), 130–138.
25. Kreich, J. and Luhn, A. and Maderlechner, G.: An Experimental Environment for Model-based Document Analysis. in Proceedings of the International Conference on Document Analysis and Recognition, Saint-Malo, France (1991), 50–58.
26. Niyogi, D. and Srihari, S.N.: Knowledge-based Derivation of Document Logical Structure. in Proceedings of the International Conference on Document Analysis and Recognition, Montreal, Canada (1995), 472–475.
27. Mao, S. and Kanungo, T.: Empirical Performance Evaluation Methodology and its Application to Page Segmentation Algorithms. IEEE Transactions on Pattern Analysis and Machine Intelligence **23** (2001), 242–256.
28. Artires, T.: Poorly Structured Handwritten Documents Segmentation using Continuous Pobabilistic Feature Grammars. in Workshop on Document Layout Interpretation and its Applications (DLIA2003).
29. Namboodiri, A.M. and Jain, A.K.: Robust Segmentation of Unconstrained On-line Handwritten Documents. in Proceedings of the Fourth Indian Conference on Computer Vision, Graphics and Image Processing, Calcutta, India (2004), 165–170.
30. NIST : NIST Scientific and Technical Databases, http://www.nist.gov/srd/.
31. LAMP: Documents and Standards Information, http://documents.cfar.umd.edu/resources/database/