A RUN LENGTH SMOOTHING-BASED ALGORITHM FOR NON-MANHATTAN DOCUMENT SEGMENTATION

Stefano Ferilli, Fabio Leuzzi, Fulvio Rotella and Floriana Esposito
Dipartimento di Informatica – Universitą di Bari
Via E. Orabona, 4 70126 Bari
{ferilli, esposito}@di.uniba.it
{fabio.leuzzi,fulvio.rotella}@uniba.it

Abstract Layout analysis is a fundamental step in automatic document processing, because its outcome affects all subsequent processing steps. Many different techniques have been proposed to perform this task. In this work, we propose a general bottom-up strategy to tackle the layout analysis of (possibly) non-Manhattan documents, and two specializations of it to handle both bitmap and PS/PDF sources. A famous approach proposed in the literature for layout analysis was the RLSA. Here we consider a variant of RLSA, called RLSO (short for "Run-Lengh Smoothing with OR"), that exploits the OR logical operator instead of the AND and is particularly indicated for the identification of frames in non-Manhattan layouts. Like RLSA, RLSO is based on thresholds, but based on different criteria than those that work in RLSA. Since setting such thresholds is a hard and unnatural task for (even expert) users, and no single threshold can fit all documents, we developed a technique to automatically define such thresholds for each specific document, based on the distribution of spacing therein. Application on selected sample documents, that cover a significant landscape of real cases, revealed that the approach is satisfactory for documents characterized by the use of a uniform text font size.

1. Introduction

Automatic document processing is a hot topic in the current computer science landscape [Nagy, 2000], since the huge and ever-increasing amount of available documents cannot be tackled by manual expert work. Document layout analysis is a fundamental step in the document processing workflow, that aims at identifying the relevant components in the document (often called *frames*, because they can be identified with rectangular regions in the page), that deserve further and specialized processing. Thus, the quality of the layout analysis task outcome can determine the quality and even the feasibility of the

whole document processing activity. Document analysis is carried out on (a representation of) the source document, that can be provided either in the form of a digitized document, i.e. as a bitmap, or in the form of a born-digital document (such as a word processor file, a PostScript or PDF file, etc.), made up of *blocks*. In the former case, the basic components are just pixels or connected components in a raster image, while in the latter they are higher level items such as words or images. The differences between the two cases are so neat that specific techniques are typically needed for each of them.

The literature usually distinguishes two kinds of arrangements of the document components: Manhattan (also called Taxi-Cab) and non-Manhattan layouts. Most documents (e.g., almost all typeset documents) fall in the former category, in which significant content blocks correspond to rectangles surrounded by perpendicular background areas. Conversely, components in non-Manhattan layouts have irregular shapes and are placed in the page in 'fancier' positions. Many algorithms assume that the document under processing has a Manhattan layout, which significantly simplifies the processing activity. Unfortunately, this is not true in general. Previous work has agreed on identifying in bottom-up techniques the best candidates for handling the non-Manhattan case, since basic components grouping can ignore high level regularities in identifying significant aggregates. This work proposes a general bottom-up strategy to tackle the layout analysis of (possibly) non-Manhattan documents, and two specializations of it to handle the different cases of bitmaps and PS/PDF sources. We propose a variant of the Run Length Smoothing Algorithm, called RLSO, that exploits the OR logical operator instead of the AND [Ferilli et al., 2009]. Advantages range from efficiency, to the ability of successfully handling non-Manhattan layouts, to the possibility of working on both scanned and born-digital documents with minor changes. As for the classical technique, however, it requires the user to set thresholds for the run length smoothing, which is a quite delicate issue. Indeed, the quality of the overall result depends on setting proper thresholds, but this is not a typical activity for humans, that are for this reason often unable to provide correct values. Additionally, there is no single threshold that can fit all documents, but each case may require a specific value depending on its peculiar layout features. Thus, we also propose a technique for automatically assessing the run length thresholds to be applied in the RLSO algorithm.

The next section introduces the proposed the Run Length Smoothing technique for non-Manhattan document segmentation, relating it to relevant past work in the literature, and the automatic assessment of document-specific thresholds for it. Then, Section 3 shows the effect of applying such a technique on real-world documents having layouts at different levels of complexity. Lastly, Section 4 concludes the paper and outlines current and future work on the subject.

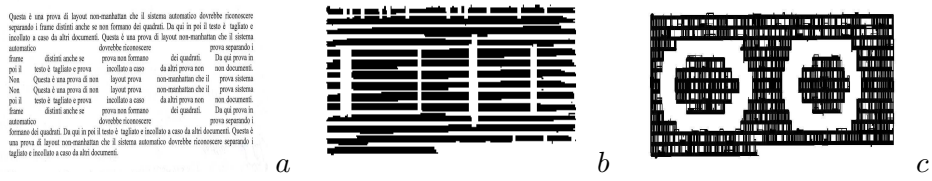

Figure 1. A non-Manhattan fragment of a scanned document (a), the corresponding RLSA output (b), and the corresponding RLSO output (c)

2. Run Length Smoothing-based Algorithms

In this section we propose a novel approach to the segmentation of both scanned and born-digital documents together with the automatic assessment of the needed thresholds.

2.1 Application to scanned images

The proposed RLSO approach includes some ideas taken from the RLSA, CLiDE and the DOCSTRUM plus other novelties. In the case of digitized documents, it works as follows:

1 horizontal smoothing, carried out row-wise on the image with threshold t_h;
2 vertical smoothing, carried out column-wise on the image with threshold t_v;
3 logical OR of the images obtained in steps 1 and 2 (the outcome has a black pixel wherever at least one of the input images has one, or a white pixel otherwise).

While the RLSA returns rectangular frames (see Figure 1-b), the RLSO algorithm identifies irregular connected components (see Figure 1-c), each of which is to be considered a frame. The actual content of a frame can be extracted by applying a logical AND of the connected component with the original image. Figure 1 refers to a non-Manhattan fragment of document layout.

RLSO improves flexibility and efficiency of the results with respect to classical RLSA. A straightforward advantage of using the OR operator is that it preserves all black pixels from the original smoothed images, and hence no second horizontal smoothing is required to restore the layout fragments dropped by the AND operation. Another improvement of efficiency is due to the fact that, since the aim is grouping small original connected components (such as characters) into larger ones (such as frames), shorter thresholds are sufficient (to fill inter-character, inter-word or inter-line spaces), and hence the algorithm is faster because it has to fill with black pixels less runs. A further improvement

may consist of applying vertical smoothing of step 2 directly on the image resulting from step 1, which allows to completely avoid step 3. This latter option should not introduce further connections among components, and could even improve the quality of the result (e.g., by handling cases in which adjacent rows have inter-word spacings vertically aligned, and would not be captured by the pure vertical smoothing).

2.2 Application to born-digital documents

In the case of scanned images processed by the RLSO algorithm, black pixels play the role of basic blocks, and two blocks are merged together whenever the horizontal or vertical distance in white pixels (i.e., the white run length) between them is below a given threshold. Thus, RLSO can be cast as a bottom-up and distance-based technique. The same approach can be, with minor changes, extended to the case of digital documents, where the basic blocks are rectangular areas of the page and the distance among adjacent blocks can be defined as the Euclidean distance between their closest edges, according to the projection-based approach exploited in CLiDE [Simon et al., 1997]. Thus, assuming initially a frame for each basic block, more complex frames can be grown bottom-up by progressively merging two frames whose distance (computed as the shortest distance of the blocks they contain) is below a given threshold. Such a technique resembles a single-link clustering procedure [Berkhin, 2002], where however the number of desired clusters is not fixed in advance, but automatically derives from the number of components whose distance falls below the given thresholds.

A problem to be solved in such an extension is how to define adjacency between blocks. Indeed, in scanned images the pixels are lined up in a grid such that each element has at most four adjacent elements in the horizontal and vertical directions. Conversely, rectangular blocks of variable size may have many adjacent blocks on the same side, according to their projections (as explained in [Simon et al., 1997]). A similar issue is considered in the DOCSTRUM, where the basic blocks correspond to text characters, and is tackled by imposing a limit on the number of neighbors to be considered rather than on the merging distance. This solution would not fit our case, that is more general, where a block can include a variable number of characters, and hence the number of Nearest Neighbors to be considered could not be easily determined as in the DOCSTRUM. The resulting algorithm works as follows, where steps 2-3 correspond to horizontal smoothing and steps 4-5 correspond to vertical smoothing:

1 build a frame for each basic block, such that the corresponding basic block is the only element of the frame;

Figure 2. (a) A non-Manhattan fragment of a (scanned/natively digital) document and the corresponding RLSO output on (b) scanned document and (c) natively digital document.

2 compute the set H of all possible triples $(d_h, b1, b2)$ where $b1$ and $b2$ are horizontally adjacent basic blocks and d_h is the horizontal distance between them;

3 while H is not empty and the element having lowest d_h value, $(\overline{d_h}, \overline{b_1}, \overline{b_2})$, is such that $\overline{d_h} < t_h$

   (a) merge in a single frame the frames to which $\overline{b_1}$ and $\overline{b_2}$ belong

   (b) remove that element from H

4 compute the list V of all possible triples $(d_v, b1, b2)$ where $b1$ and $b2$ are vertically adjacent basic blocks and d_v is the vertical distance between them;

5 while V is not empty and the element having lowest d_v value, $(\overline{d_v}, \overline{b_1}, \overline{b_2})$, is such that $\overline{d_v} < t_v$

   (a) merge in a single frame the frames to which $\overline{b_1}$ and $\overline{b_2}$ belong

   (b) remove that element from V

After obtaining the final frames, each of them can be handled separately by reconstructing the proper top-down left-to-right ordering of the basic blocks it includes. Figure 2-c shows the output, along with the discovered frames highlighted, of this version of the algorithm on the born-digital version of the document in Figure 2-a. Efficiency of the procedure can be improved by considering the H and V sets as priority queues based on the distance attribute and by preliminarily organizing and storing the basic blocks top-down in 'rows' and left-to-right inside these 'rows', so that considering in turn each of them from top to bottom and from left to right is sufficient for identifying all pairs of adjacent blocks. This method resembles that proposed in [Simon et al., 1997], with the difference that no graph is built in this case, and more efficiency is obtained by considering adjacent components only. The "list-of-rows" structure allows to efficiently find adjacent blocks by limiting the number of useless comparisons. The "nearest first" grouping is mid-way between the minimum spanning tree technique of [Simon et al., 1997] and a single-link clustering technique. Compared to the ideas in [O'Gorman, 1993], we exploit the distance between component borders rather than between component centers. Indeed, the latter option, adopted in DOCSTRUM, cannot be exploited in our case since the

starting basic components can range from single characters to (fragments of) words, and this would cause a lack in regularity that would affect the proper nearest neighbour identification.

2.3 Automatic assessment of RLSO thresholds

A very important issue for the effectiveness of Run Length Smoothing-based algorithms is the choice of proper horizontal/vertical thresholds. Indeed, too high thresholds could erroneously merge different content blocks in the document, while too low ones could return an excessively fragmented layout. Thus, a specific study of each single document would be in principle needed by the human expert to obtain effective results, which would tamper at the basis the idea of a fully automatic document processing system. The threshold assessment issue becomes even more complicated when documents including different spacings among components are considered. This typically happens in document pages that include several different kinds of components, that are independent of each other and hence exploit different spacing standards (although the spacings are homogeneous within each component). For instance, a title, usually written in larger font characters, will use an inter-character, inter-word and inter-line spacing wider than that of normal text. Obviously, the attempt to merge related wider components would also merge smaller unrelated ones as an undesirable side effect. Thus, a single threshold that is suitable for all components cannot be found in these cases, but rather the layout analysis process should be able to identify homogeneous areas of the page and to selectively define and apply different thresholds for each of them. Hence, a strong motivation for the development of methods that can automatically assess proper thresholds, possibly based on the specific document at hand. This is why some works in the literature have been devoted to trying and finding automatically values for the t_h, t_v and t_a parameters of the RLSA in order to improve its performance (e.g., [Gatos et al., 2000]). Studies on RLSA suggest setting large thresholds, in order to keep the number of black pixels in the smoothed images sufficiently large to prevent the AND operator from dropping again most of them (indeed, this is the reason why an additional horizontal smoothing is provided for).

The use of the OR operator in RLSO needs small values, that brings several advantages in terms of efficiency, but the threshold assessment must be carefully evaluated, because as soon as two components are merged there is no step that can break them again, and this can be a problem in cases in which logically different components are very close to each other and might be merged together. We carried out a study of the behavior of run lengths in several documents to search for regularities that can help in developing an automatic thresholding technique for RLSO, that is able to define proper thresholds for each

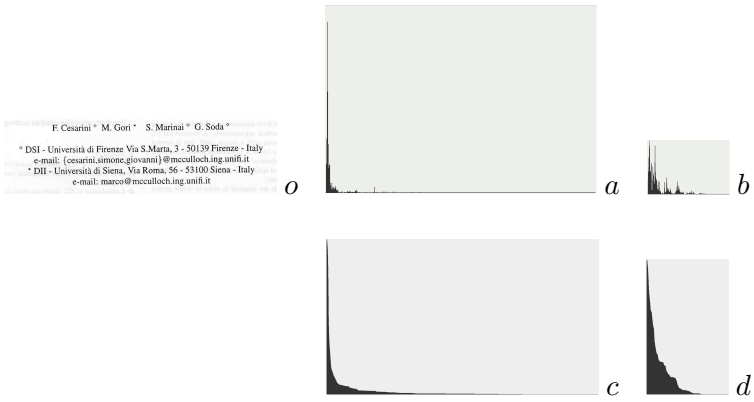

Figure 3. A sample study on run length distribution on a fragment of scientific paper. (o) The original scanned document, histograms of the (a) horizontal and (b) vertical run lengths and the corresponding (c-d) *cumulative* histograms.

document at hand based on their specific distribution of spacings. Specifically, we focus on the scanned document case, leaving an analysis of possible application to natively digital documents for future work. Consider the sample document depicted in Figure 3-o, corresponding to a fragment of the first page in reference [Cesarini et al., 1999], and representing the authors and abstract of a scientific paper. Figures 3-a and 3-b show the histograms of the white run lengths distribution, reporting the amount of runs (respectively horizontally and vertically) for each possible length (notice that the runs from the borders to the first black pixels have been ignored). Several irregular peaks are immediately apparent, that are in some sense clustered into groups separated by an absence of runs for several adjacent lengths. Indeed, the peaks correspond to the most frequent spacings in the document, and the most prominent peak clusters can be supposed to represent the homogeneous spacings corresponding to different layout components. More precisely, a significantly high and wide peak is present for very short runs, corresponding to the most frequent spaces in the page, i.e. intra-letters, inter-letters and inter-words ones in the horizontal case, and intra-letter and inter-lines ones in the vertical case. After such an initial peak, the behavior is quite different for the horizontal (Figure 3-a) and vertical (Figure 3-b) cases. In the former the histogram shows another noticeable peak immediately after the first one, corresponding to the distance between the first two and the last two author names, and then becomes quite flat, with a few small peaks in the middle that, looking at the original image, should correspond to the runs between the outlines of some letters. Conversely, the vertical histogram has several peaks at various lengths, that are due to the different inter-line spacings used in the fragment. This suggests, as a first idea,

that each peak corresponds to a group of homogeneous spacings that are likely to separate blocks in a same frame, and hence should be somehow related to the various thresholds of interest. However, the histograms are too irregular to be usefully exploited by some technique. For this reason, a more regular (and interpretable) shape would be obtained by considering the *cumulative* histograms of run lengths (see Figures 3-c and 3-d), where each bar corresponds to a possible run length and represents the amount of runs in the given image having length larger or equal than that. To introduce some tolerance, the actual histogram bars are scaled down to 10% of the original value. Now, the shape of the cumulative graphic is clearly monotonically decreasing, and flat zones identify lengths for which no runs are present in the document (or, equivalently, if several adjacent run lengths are missing, the slope is 0). Mathematically, the derivative of the shape is always less or equal than 0, and should indicate variations in the amount of runs introduced by a given length: 0 means no runs for that length, while the larger the absolute value, the more the difference in the number of runs between adjacent lengths in that point. In our implementation, the slope in a given bar b of a histogram $H(\cdot)$ is computed according to the difference between the previous bar $b-1$ and the next bar $b+1$ as

$$\frac{H(b+1)-H(b-1)}{(b+1)-(b-1)} = \frac{H(b+1)-H(b-1)}{2} \tag{1}$$

for inner bars; as

$$\frac{H(b+1)-H(b)}{(b+1)-(b)} = H(b+1)-H(b) \tag{2}$$

for the first bar, and as

$$\frac{H(b)-H(b-1)}{(b)-(b-1)} = H(b)-H(b-1) \tag{3}$$

for the last bar. A different definition could take into account only equation (2) for all bars (except the last one). [1]

Adopting as RLSO thresholds the first horizontal and vertical length where the slope becomes 0 yields the result in Figure 4-a1. The result successfully separates the abstract and the authors in different connected components (i.e., frames), as desired. Actually, one can note that two components are found for the authors, each containing a pair of author names. This happens because in the author section a different spacing is exploited, so that the space between the second and third authors is bigger than that between the first-second and

[1]Note that very short white runs, made up of a few pixels, are often unlikely, which would yield an initial flat region in the histogram, having slope 0. Of course this should be skipped for our purposes.

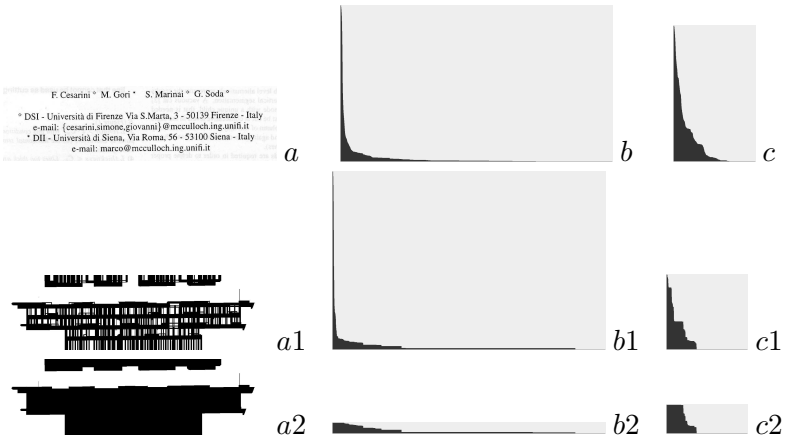

Figure 4. (a, a1, a2) Successive applications of RLSO with automatic threshold assessment on a fragment of scientific paper and the corresponding (b, b1, b2) horizontal and (c, c1, c2) vertical white run histograms used for automatic threshold assessment on the document reported in (a)

third-fourth pairs. Although it is clear that the final target layout should include all authors in a single frame, the fact that the names were not equally spaced suggests that the Authors of the paper wanted to mark some difference between the first and second pair of names, which is satisfactorily caught by the above result. Now, in case further block merge is desired, one might take for granted the connected components retrieved by the previous step, and wonder how to go on and find further mergings on this new image. By applying the same reasoning as before, the new cumulative histograms are computed, as shown in Figure 4-b1 and 4-c1. An initial peak is still present, due to the small white rectangles inside the connected components. Applying again the RLSO with thresholds equal to the first lengths where the slope becomes 0 in this new histogram yields the result in Figure 4-a2, that is just what we were looking for. Notice that, while the former round produced connected components with many inner white regions, the second round completely eliminated those small spaces, and hence computing the new cumulative histogram, reported in Figure 4-b2 and 4-c2, reveals a neat cut (wide region with slope 0) on the first bars. Thus, horizontally there is no more slope 0 region except the trailing one, while vertically there would be one more flat zone that would merge the authors and abstract frames together. The above example suggests a progressive technique that merges connected components at higher level identified according to a significant group of runs having similar length. In other words, it finds and applies first a threshold that merges smaller components, and then progressively larger thresholds for merging wider components. As to

Figure 5. Progressive application of RLSO with automatic thresholding on a non-Manhattan layout portion of the document in Figure 1.

Figure 6. Application of RLSO to a layout portion of a newspaper.

the possible noise in documents, it is worth noting that born-digital documents are noise free and only scanned documents could be affected by some noise. However, even in this case, the method can be successfully applied to a scale reduction of the image with the aim of obtaining a denoised image and, after the process execution, taking back the resulting output to the original scale.

3. Sample Applications to Complex Layouts

The proposed technique has been applied to several scanned images of documents showing complex layout. In the following, a selection of fragments belonging to such images is reported, that represent a landscape of features and problems that may expected to be found in a general set of documents.

Figure 5 shows the progressive application of the RLSO algorithm with automatic thresholding on a scanned portion of the document in Figure 1. Here, the problems are the non-Manhattan nature of the document layout, and the use of different font sizes (and spacings) in the various frames: larger in the central quotation, smaller in the surrounding running text. On the left is the original fragment, and on the right the results of two successive applications of the RLSO. It is possible to note how the thresholding technique succeeds in progressively finding different levels of significant spacings and, as a consequence, different levels of layout components. The first step already merges correctly the paragraphs in the running text, but just the words in the quotation. As expected, the merging of components written in larger size/spacing is de-

layed with respect to those using smaller size/spacing, and indeed it takes place in the second step. Actually, the second step confirms the same surrounding frames as the first one, and additionally merges all quotation rows, except the first one, into a single frame. No better result can be found, because a further iteration would start merging different frames and spoil the result. Figure 6 shows on the left two scanned portions of newspapers, characterized by many components very different for type of content (text, lines and one case even pictures), font size and spacings. In this case the first application of the automatic thresholded RLSO, shown on the right, already returns a satisfactory set of components. No different components are merged, and text paragraphs are already retrieved. As to components having larger font size, those of medium size are successfully retrieved as well, while those of significantly larger size (the titles) are merged at least at the level of whole words. However, it would be safer to preliminarily identify and remove straight lines, not to affect the histograms on which the automatic threshold computation is based.

4. Conclusion

Document layout analysis is a fundamental step in the document processing workflow. The layout analysis algorithms presented in literature are generally divided into two main categories, bottom-up and top-down, according to their approach to the problem, and have been mainly developed to deal with digitized documents. However, document analysis has to be carried out on both digitized documents and born-digital documents. Furthermore, most of them are able to process Manhattan layout documents only. Thus, existing algorithms show a limitation in dealing with some situations. Among the several layout analysis algorithms presented in literature, a very successful approach has been the application of Run Length Smoothing. We presented a variant of the classical RLSA approach, called RLSO because it is based on the application of the OR operator. RLSO is a general bottom-up strategy to tackle the layout analysis of non-Manhattan documents. Like RLSA it is based on the definition of horizontal and vertical thresholds for the smoothing operator, but requires different values than those that are useful in RLSA, due to the different approach. We also proposed a technique to automatically define RLSO thresholds for each single document, based on the distribution of spacing therein. Application on selected samples of documents, that aimed at covering a significant landscape of real cases, revealed that the approach is satisfactory for documents characterized by the use of a uniform text font size. More complex cases, in which text, graphics and pictures are present, and several different font sizes and spacings are exploited for different components (such as title, authors, body of an article), obviously require more complex approaches. However, the proposed technique candidates itself as an interest-

ing starting point, since by iterating its application the basic elements of even complex layouts can be successfully found.

Current and future work will be centered on a deeper study of the behavior of the automatic threshold technique on one side, and of the features of such complex documents on the other, in order to reach a full technique that is applicable to any kind of layout. First of all, a way to determine when the iterations are to be stopped, in order to obtain a good starting point without spoiling the result, must be found. Then, how to exploit this technique as a component of a larger algorithm that deals with complex documents must be defined. We are currently investigating the possibility of analyzing the distribution of size and spacings of basic components to obtain a clustering of components based on features such as their position and spacing. After identifying clusters characterized by uniform parameters, the technique can be applied separately to each such zone by selectively computing and applying a proper threshold for each.

References

[Berkhin, 2002] Berkhin, Pavel (2002). Survey of clustering data mining techniques. Technical report, Accrue Software, San Jose, CA.

[Cesarini et al., 1999] Cesarini, F., Marinai, S., Soda, G., and Gori, M. (1999). Structured document segmentation and representation by the modified x-y tree. In *ICDAR '99: Proceedings of the Fifth International Conference on Document Analysis and Recognition*, page 563, Washington, DC, USA. IEEE Computer Society.

[Ferilli et al., 2009] Ferilli, Stefano, Biba, Marenglen, Esposito, Floriana, and Basile, Teresa M.A. (2009). A distance-based technique for non-manhattan layout analysis. In *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR-2009)*, volume I, pages 231–235. IEEE Computer Society.

[Gatos et al., 2000] Gatos, B., Mantzaris, S. L., Perantonis, S. J., and Tsigris, A. (2000). Automatic page analysis for the creation of a digital library from newspaper archives. *International Journal on Digital Libraries (IJODL*, 3:77–84.

[Nagy, 2000] Nagy, George (2000). Twenty years of document image analysis in pami. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(1):38–62.

[O'Gorman, 1993] O'Gorman, L. (1993). The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1162–1173.

[Simon et al., 1997] Simon, Anik◆, Pret, Jean-Christophe, and Johnson, A. Peter (1997). A fast algorithm for bottom-up document layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(3):273–277.