Structured document segmentation and representation by the modified X-Y tree

Structured Document Segmentation and Representation by the Modified X-Y tree

F. Cesarini ◇ M. Gori * S. Marinai ◇ G. Soda ◇

◇ DSI - Università di Firenze Via S.Marta, 3 - 50139 Firenze - Italy
e-mail: {cesarini,simone,giovanni}@mcculloch.ing.unifi.it
* DII - Università di Siena, Via Roma, 56 - 53100 Siena - Italy
e-mail: marco@mcculloch.ing.unifi.it

Abstract

In this paper we describe a top-down approach to the segmentation and representation of documents containing tabular structures. Examples of these documents are invoices and technical papers with tables. The segmentation is based on an extension of X-Y trees, where the regions are split by means of cuts along separators (e.g. lines), in addition to cuts along white spaces. The leaves describe regions containing homogeneous information and cutting separators. Adjacency links among leaves of the tree describe local relationships between corresponding regions.

1. Introduction

Document segmentation is a basic task in storage and retrieval systems, and is either attacked by bottom-up or top-down methods [1]. Basically, in bottom-up analysis connected components are extracted from the image and are subsequently aggregated in higher level structures, creating words, text lines, paragraphs, and so on (e.g. RLSA: Run Length Smoothing Algorithm). In top-down analysis a page is segmented from larger components to smaller subcomponents. For instance in the X-Y tree decomposition, the page is recursively split in sub-parts by alternating horizontal and vertical cuts along spaces [2].

Top-down methods are generally faster and have the advantage of directly extracting a geometric hierarchy among regions, that is frequently tightly linked with the logical structure of the document. Bottom-up methods appear to be more appropriate for "non-Manhattan" layouts (e.g. pages where figures are intermixed both in and around text).

In this paper we describe a document layout analysis method for working with documents containing tabular structures. In particular we are interested in the analysis of invoices [3], a key application of the STRETCH project[1]. The X-Y tree approach appears appropriate for the segmentation of invoices that are generally based on a regular layout. However, invoices are generally strongly based on tables, and the X-Y decomposition cannot proceed inside these regions, for the presence of lines. In order to face this problem, we modify the cutting modality of the X-Y approach, by introducing a more general concept of separators, and considering also cuts corresponding to horizontal and vertical lines.

The second property of invoices that must be taken into account is that logical relationships between objects are mostly local. For instance, the field containing the total amount is usually printed close to the keyword "Total". In order to have an efficient description of adjacency relationships among objects, we propose to add adjacency links between pairs of leaves of the tree.

In Section 2 we describe the segmentation approach, whereas in Section 3 we analyze the use of M-X-Y tree for the representation of the page layout. In Section 4 we report some examples of the segmentation obtained by the proposed approach.

2. X-Y tree segmentation

The X-Y tree ([2], [4]) is a well-known top-down method for page layout analysis. The basic assumption that is behind this approach is the fact that structured elements of the page are generally laid out in rectangular blocks. Furthermore the blocks can almost always be divided into groups in such a way that blocks that are adjacent to one another within a group have one dimension in common [2]. The document is split into successively smaller rectangu-

[1] *STRETCH: STorage and RETrieval by Contents of imaged documents* is an ESPRIT project (Contract N. 24977) `http://www.aetnet.it/Stretch`

lar blocks by alternately making horizontal and vertical "cuts" along white spaces. The spaces are found by using a thresholded projection profile (a projection profile is the histogram of the number of black pixels along parallel lines through the document). The result of such segmentation can be represented in a X-Y tree, where the root corresponds to the whole page, the leaves are for blocks of the page, whereas each level alternatively represents the results of horizontal or vertical segmentation. A vacuous cut [2] corresponds to a node with a unique child, that is needed when a region must be cut at consecutive levels in the same direction (e.g. a column of text may be first cut horizontally into paragraphs, and again cut in the same direction to break paragraphs into lines).

Some thresholds are required in order to define proper cutting elements, reducing the risk of "bad" cuttings (e.g. cuts along lines corresponding to parts of characters). In our implementation, all these thresholds are calculated on the basis of the average height and width of characters in the page to be processed (C_h, C_w).

Definition 1 Cutting space. *The rectangular region $\overline{R}$ is a* horizontal cutting space *for region R if the following conditions hold (Fig. 1):*
1) $\overline{R}.width = R.width$ *($\overline{R}$ and R have the same width).*
2) *The region $\overline{R}$ is* white[2].
3) $\overline{R}.height > T_h$. *By setting $T_h = C_h$, we allow the segmentation into separated lines of text. If $T_h = 2 \cdot C_h$, then we stop the process at the paragraph level.*
Vertical cutting spaces can be similarly defined.

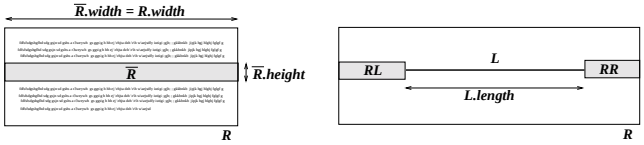


Figure 1. Left: notations for cutting space. Right: regions RL and RR for cutting lines.

Definition 2 Cutting line. L *is a* horizontal cutting line *for region R if the following conditions hold:*
1) $\frac{L.lenght}{R.width} > T_L$. *Where $L.lenght$ is the line length, and T_L is a threshold allowing to cut only along "long" lines (typically $0.25 \leq T_L \leq 0.5$).*
2) *The two rectangles RL and RR are* white, *where RL (RR) is the rectangle of height $2 \cdot C_h$ covering the area from the left (right) extrema of L to the left (right) border of R (Figure 1).*
3) $L.length > 2 \cdot C_w$. *With this condition we avoid using parts of characters (as the horizontal stroke of a "L") as cutting lines.*
4) $L.thickness < C_h$. *Lines too thick are not considered in order to avoid splitting using parts of logos or graphical objects.*
Vertical cutting lines can be defined in a similar way. Figure 2 contains an example of cutting lines.

[2]In order to tolerate small amounts of noise, a region R is said to be *white* if the percentage of black pixels in R is below a small threshold value (e.g. 2%).

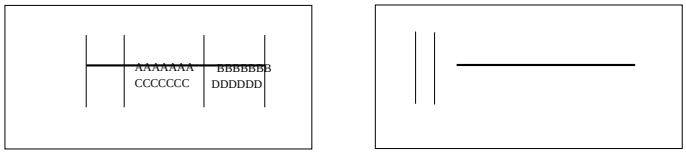


Figure 2. Example of cutting line (left) and a line that cannot be used as cutting line (right).

In the proposed approach a page is recursively segmented by means of separators introduced in Def. 1, and Def. 2. In each subregion, the algorithm first looks for cutting lines; cuts are made in correspondence of all cutting lines such that $L.lenght > T_{RL} \cdot L_{max}$ (L_{max} is the length of the longest cutting line in the subregion, and $T_{RL} = 0.7$). If cutting lines are not found, then cutting spaces are searched, and cuts are made in correspondence of all cutting spaces such that $A_S > T_{RA} \cdot A_{max}$ (where A_S is the area of cutting space S, A_{max} is the area of the largest cutting space, and $T_{RA} = 0.7$).

The algorithm just described is essentially data-driven, since the splitting strategy uses few high-level information on the document class. This approach provides worst segmentation than a model-driven one tuned on a specific problem, however it allows us to obtain good results for a broad range of documents without changing the approach (see Section 4).

It should be observed that some documents contain other kinds of separators. For instance we found documents where different items are separated by abrupt discontinuities in the background colors, or by dashed lines. In these cases, the segmentation capabilities can be further expanded by adding these separators to the set of cutting elements.

3. Modified X-Y tree

In this section we describe the M-X-Y tree as a data structure for representing documents segmented with the algorithm presented in Section 2. The main features of M-X-Y tree are the storage of cutting lines as leaves of the tree and the description of inter-leaves adjacency relations by means of appropriate links.

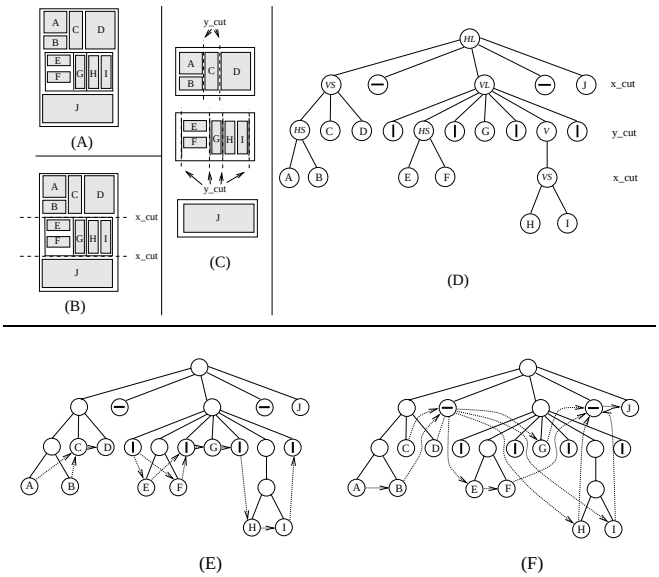


Figure 3. Example of M-X-Y tree. (A): a sample page. (B) and (C): the first and the second levels of segmentation. (D): the M-X-Y tree. (E), (F): Right of and Below arcs.

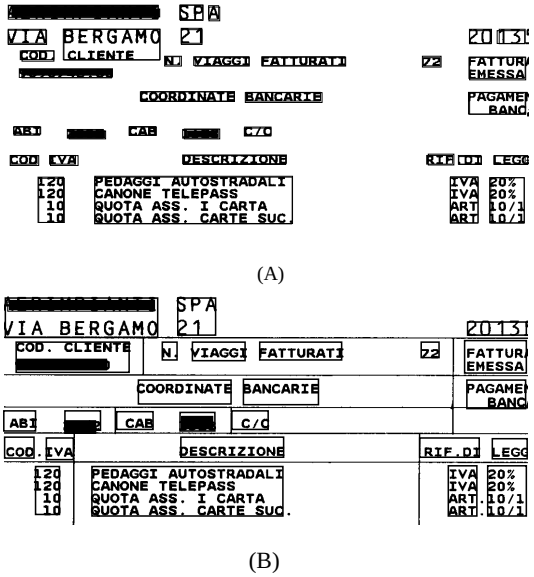


Figure 4. (A): segmentation with a RLSA-based approach. (B): segmentation with M-X-Y tree.

Definition 3 *A* **Modified X-Y Tree** $\mathcal{MXY} \doteq \{N, A, \mathcal{A}_N, \mathcal{A}_A\}$ *is a tree with nodes* N*, arcs* A *connecting leaves, node attributes in* $\mathcal{A}_N$*, arc attributes in* $\mathcal{A}_A$*.*

An example of M-X-Y tree is shown in Figure 3, node and arc attributes are defined as follows.

Definition 4 Node attributes *are 3-tuples:* $\mathcal{A}_{N_i} \doteq \{T_i, Pos_i, Child_i\}$*, where* T_i *is the* node type. Pos_i *locates the rectangle corresponding to node* N_i *(described by its upper-left* (xu_i, yu_i) *and lower-right* (xb_i, yb_i) *corners).* $Child_i$ *is the list of children of* N_i*.*

If N_i is an internal node, then T_i defines the cutting way that generates its children (HL, VL for horizontal and vertical lines, HS, VS for spaces, and V for a vacuous cut). If N_i is a leaf, then T_i defines the kind of region corresponding to the node: $T_i = T$ for text, $T_i = I$ for images, and $T_i = L$ for a line separator (represented as a small line in Figure 3).

Adjacency links among leaves of the tree can be seen as an adjacency graph, where nodes of the graph correspond to leaves of the tree. An adjacency graph (see e.g. [5]), describes the structure of a document by giving, for each region, the position of nearest regions in the horizontal and vertical directions.

Definition 5 Arc attributes *(for arc connecting* N_i *to* N_j*) are* $\mathcal{A}_{A_{i,j}} \doteq \{P_{i,j}, d_{i,j}\}$*. Where* $P_{i,j} = B$ *if* N_j *is* Below N_i *(see Def. 6).* $P_{i,j} = R$ *if* N_j *is* Right of N_i*.* $d_{i,j}$ *is the distance between the nearest borders of the rectangles corresponding to the connected nodes.*

Definition 6 *Given two regions* A *and* B*, described by nodes* N_i *and* N_j*, we say that* B *is* **Below** A *if the following three conditions are satisfied.*
1) $yu_i < yu_j$
2) $[xu_i, xb_i] \cap [xu_j, xb_j] \neq 0$
3) $\exists x \in ([xu_i, xb_i] \cap [xu_j, xb_j]) \mid$ *the segment joining points* (x, yb_i) *and* (x, yu_j) *do not crosses any other region.*

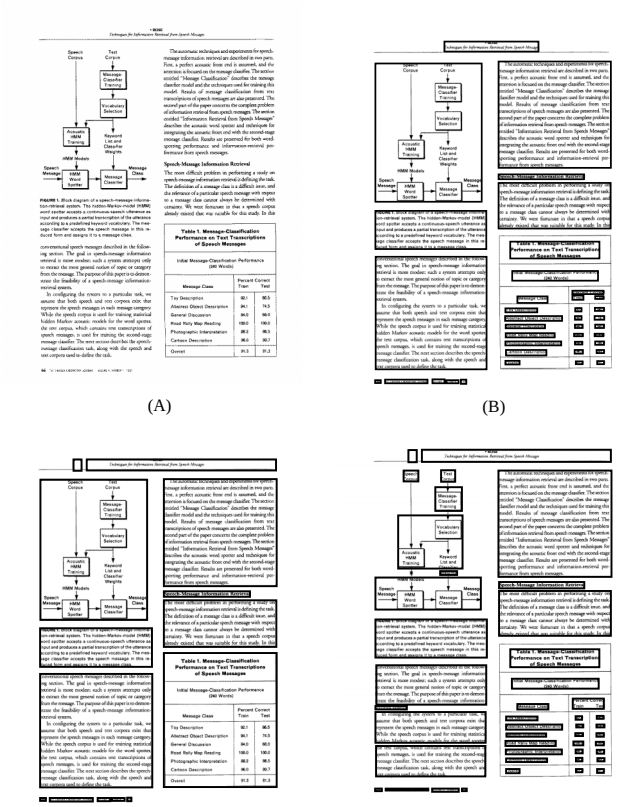
The **Right of** relation can be similarly defined.

4. Experimental results

In this section we analyze examples of the segmentation produced by the proposed approach on two classes of documents: invoices and technical documents.

An example of the segmentation of a portion of invoice obtained with M-X-Y tree and with a bottom-up method (RLSA) is shown in Figure 4. The RLSA implemented is based on two steps: First, the horizontal and vertical white runs with length below appropriate thresholds (RL_h and RL_v) are filled with black pixels. Second, regions are found by locating connected components in the filled image.

By using the RLSA on invoices containing many lines, the filling step has the effect of merging some textual parts



(A) (B) (C) (D)

Figure 5. Comparison of segmentations of a page of a technical paper (see text).

together with lines, thus locating only few regions. Consequently, a RLSA based approach can be used only after removing lines from the image, as shown in Figure 4A. After some tests, we observed that a unique value for RL_v cannot be adopted for all the document, since small values separate all the text lines, whereas higher values merge lines with different meaning. Figure 4A is in a border situation: two words with different meaning ("BERGAMO" and "CLIENTE") are put together since the vertical distance between them (after removing layout lines) is lower that the distance between lines of the same box (e.g. "BERGAMO" and the erased word that is the name of the company). Figure 4B contains the segmentation obtained by using the proposed segmentation. In this case we have a segmentation more similar to the "human meaning" of the document.

An example of the segmentation of a technical document is shown in Figure 5. **(A)** contains the original image. **(B)** is the segmentation obtained with M-X-Y tree, and the regions appear homogeneous. For instance blocks in the textual part describe paragraphs of text, whereas blocks inside the table are separated. Moreover the drawing on the upper-left part of the page is a unique region. **(C)** reports the segmentation obtained by cutting only along spaces, and considering a low threshold for the location of cutting spaces from the projection profile. Most part of regions are the same as those obtained in the first part, but the table is considered as a single region. If we desire to have separated regions for each block inside the table, we need to increase the threshold in order to make cuts also in correspondence of lines of the table (Figure 5 **(D)**). In this case the regions inside the table are well segmented, but some strange segmentations occur in the paragraphs of text, and the drawing is split in smaller sub-blocks.

Two main limits of the segmentation algorithm should be mentioned. First, as the other X-Y based methods, the algorithm requires a well aligned image. Second the document can be cut only along separators covering the whole width (or height) of the image corresponding to the node to be split. Consequently tables nested in the text cannot be segmented.

5. Conclusions

We described a top-down data-driven approach to the segmentation and representation of tabular documents, in particular invoices. Two main contributions of the proposed method can be highlighted: 1) the integration of more cutting strategies in the framework of the "classical" X-Y tree segmentation. 2) the introduction of a data structure (the Modified X-Y tree), for representing the segmented document. We would like to thank P. Tanganelli for the implementation of the early version of the algorithm. This work is partially supported by the ESPRIT project STRETCH (STorage and RETrieval by Content of imaged documents).

References

[1] A. Jain and B. Yu, "Document representation and its application to page decomposition," *IEEE TPAMI*, vol. 20, pp. 294–307, March 1998.

[2] G. Nagy and S. Seth, "Hierarchical representation of optically scanned documents," in *Proc. of ICPR*, pp. 347–349, 1984.

[3] F. Cesarini, M. Gori, S. Marinai, and G. Soda, "INFORMys: A flexible invoice-like form reader system," *IEEE TPAMI*, vol. 20, pp. 730–745, July 1998.

[4] M. Viswanathan, "Analysis of scanned documents - a syntactic approach," in *Structured Document Image Analysis*, pp. 115–136, Springer-Verlag, 1992.

[5] J. Yuan, Y. Y. Tang, and C. Y. Suen, "Four directional adjacency graphs (fdag) ant their application in locating fields in forms," in *Proc. ICDAR'95*, pp. 752–755, 1995.