Proc. DAS'94, Document Image Analysis Systems, Kaiserslautern, Germany, October 1994

A REGION-BASED SYSTEM FOR THE AUTOMATIC EVALUATION OF PAGE SEGMENTATION ALGORITHMS

Sabine Randriamasy
S.E.I./IRESTE
La Chantrerie C.P. 3003
44087 Nantes, France
e-mail: srandria@lati.ireste.fr

Luc Vincent
Xerox Imaging Systems
9 Centennial Drive
Peabody, MA 01960, USA

ABSTRACT

A method for automatically evaluating the quality of document page segmentation algorithms is described. Page segmentation involves decomposing a page into its structural and logical units such as paragraphs, halftones, captions and tables. These units are then ordered and logically associated. These two steps are very important in a document recognition strategy. Many different techniques have been proposed in literature to segment document pages, but since no robust method to benchmark them is available, it is impossible to evaluate them reliably. Our proposed page segmentation benchmarking strategy is a region-based approach, in which segmentation results are compared with manually generated "ground truth files", containing a synthetic description of all possible correct segmentations. It is described in detail and we give a set of guidelines to generate ground truth files and propose a general format to represent them. The algorithm is simple and fast, and provides a multi-level output for each segmentation, from the page level to the region level.

1 Introduction

The document recognition of commercially available OCR packages usually proceeds in the following steps:

- Page segmentation: decomposes a document page into units such as text, halftones, graphics, rulings, tables, etc.,

- Region ordering: derives the reading order of the page,

- Recognition: retrieves the semantic contents of the significant extracted regions, e.g. Optical Character Recognition (OCR) is performed on the text regions.

- Format/layout analysis: determines such things as margins, paragraph endings, tabulations and indentations.

Over the past twenty years, research efforts on document analysis have been focused on recognition [15, 7]. Significant improvements of OCR accuracy were enabled by the availability of reliable benchmarking tools for OCR [14, 2]. Estimating the quality of an OCR algorithm is moderately difficult: the solution is unique, and one can simply compare the text output with a pre-stored "ground truth file", containing the ideal output [13].

A successful page segmentation is a necessary precondition for the other steps in the document recognition process: identification of region types allows the system to make the appropriate treatment for each of them. Information provided on the regions, such as shape, size and type, is useful for format analysis. An incorrect segmentation, may cause region ordering to output the lexical order of the page in a wrong sequence. Lastly, segmentation "decolumnizes" multi-column documents, in order for OCR to read the text lines in the correct order. Estimating the quality of a page segmentation is a therefore a crucial issue.

Page segmentation techniques are now numerous and of very good quality, [7, 3, 8, 15]. Improvements, however, are needed, especially in decomposing poorly printed documents, tables, schematic drawings and low contrast zones. Due to the extreme diversity of documents to process, no technique is optimal, and there is now a wide agreement upon the need for a reliable method to benchmark such techniques [1]. Moreover, a segmentation algorithm must be tested on hundreds of input pages, which stresses the need for an automatic tool. A tool capable of providing an explicit error specification can help anyone wishing to improve a currently developed segmentation technique or to choose one.

Unlike for OCR, benchmarking page segmentation algorithms is not straightforward. The optimal segmentation is not necessarily well defined and unique for a page. Two sets of regions with similar characteristics are not necessarily of equivalent quality. Zone representation schemes are not standardized, and therefore regions cannot be compared geometrically [5]. Last, errors are not uniform across the page: they may depend on the orientation of the text lines or on the type of region.

1.1 Related work: text-based approach

The only known automated way to measure the performance of segmentation algorithms proposed by Kanai et al., [6], is part of a system which evaluates the quality of OCR output by comparing it with the OCR ground truth. The system computes the minimal number of text insertions, deletions, and block moves necessary to correct the OCR output. The derived cost includes OCR errors, so the cost of segmentation *alone*, called *"automatic zoning metric"*, is obtained by comparing the costs corresponding to manually and automatically zoned pages.

The advantage of such a method is that, being purely text-based, it does not require to specify any particular format for segmentation ground truth. Moreover, any zone representation scheme can be used.

However, the output is a number of editing operations from which it is difficult to derive much information on segmentation mistakes. Furthermore, there is, in general, no way to know whether OCR errors are due to segmentation or to region ordering.

1.2 Proposed image-based approach

Page decomposition should be evaluated independently of the rest of the recognition tasks, by a system providing an accurate and explicit error specification. As far as we know, the only method known to work reliably is visual inspection of the extracted regions. The subsequently limited range of test documents makes it then impossible to gather reliable statistics on the performances of a page decomposition system.

An automatic image-level, region-based benchmarking scheme for page segmentation, which compares results with predefined "ground truth files" describing all the possible correct solutions, is presented here. The algorithm provides an explicit diagnosis, from the page to the region level, where segmentation errors are detailed. This approach is set-based in that, regions are equivalent to the set of "black" pixels contained in the zones extracted by segmentation. It is thus independent of zone representation schemes.

In § 3, we present our segmentation ground truthing scheme. At this low-level stage, the goal of a good segmentation algorithm is defined as providing correct input for region ordering and classification. The "segmentation ground truth file" (*SGTF*(*I*)), associated with a page *I*, contains a set of "ground truth regions" (*GTR*). Each of them is equivalent to a *non-mergeable* set of text-lines gathered in a language-dependent direction D^1, while respecting reading order. The non-uniqueness of the optimal segmentation of *I* mainly lies in the fact that text columns may be split into paragraphs, subparagraphs and titles. So, in order to consider all the correct solutions, authorized "regions cuts" and tolerated "region merges" are defined in *SGTF*(*I*).

Comparison of a segmentation and its corresponding *SGTF*, is described § 4. It proceeds by testing the overlap between the two sets of regions, and comparing their contents in *I*. This way, the operations done on the *GTR* are straightforwardly identified. Segmentation errors are detected by testing the spatial layout of each union of regions in one image overlapping a union of regions in the other one, against the predefined cuts and merges. A numerical performance measure is then derived from the error diagnosis.

The segmentation ground truthing scheme has been introduced in [12] and the error detection strategy in [11]. The presented algorithms have been implemented in *C* language on a SPARC-station. They were tested on about a dozen of machine written multicolumn mixed image/text binary document images.

2 Benchmarking approach

The present region and bitmap-based method evaluates segmentation only by using segmentation results. Similarly to OCR benchmarking, given a segmentation and the associated ground truth, evaluation first consists in determining the operations to perform on one file in order to obtain the other one. Instead of additions, deletions and block moves like in [6], the operations we consider here are **splitting** and **merging,** in either the **vertical** or the **horizontal** direction.

The error analysis is done at the bitmap level, by comparing sets of *regions* in the automatically and manually zoned images. Compared with ASCII text, regions are a richer and more synthetic way to describe segmentations. Using them, segmentation errors are easier to find and to characterize. Our method proceeds in two main steps:

1 - Creation of ground-truth file *SGTF*(*I*) for each document *I* on which segmentation is to be benchmarked. This file contains a synthetic description of every possible correct segmentation of *I*.

2 - Matching of segmentation result against ground-truth file by a region-based comparison algorithm testing the overlap between the two sets of regions.

Bitmap-level and set-based approach The mostly used zone representation schemes (ZRP) are: bounding rectangles, nested bounding rectangles, polygons and piecewise rectangles. Our method is not committed to any particular ZRP: regions are defined as being the set of "black" pixels

[1] which is *vertical* for european languages and often *horizontal* for Asian languages

contained in the zones derived by segmentation, rather than the zones themselves. A region may therefore have any shape, have holes, be disconnected, etc., as long as its black pixel content is correct.

Qualitative and quantitative evaluation After the comparison of the two sets of regions $S(I)$ and $SGTF(I)$, the benchmarking algorithm provides, among other things: the number of incorrectly merged or split regions with the type of merging or splitting (horizontal, vertical), the number of meaningless regions extracted, the type of incorrectly segmented regions and the penalty associated to errors. A global rating of segmentation quality can then be derived for the page, and statistics can be gathered over the whole test.

Input to the system Our page segmentation benchmarking scheme requires three inputs per document P:

- the binary image I of the original document P,
- the predefined ground truth segmentation file $GTF(I)$,
- the segmentation result $S(I)$ to be evaluated.

The region representation format Geometrical information on the zones is used *only prior to benchmarking itself*, to build the regions by labelling the black pixels of I, into sets of pixels belonging to the same zone derived by segmentation.

To represent the zones, we have opted for polygons, which are more realistic than rectangles and encompass many existing ZRS. Each polygon has either vertical or horizontal edges. The reason is that we implicitly consider a text region as a set of merged lines, each line corresponding to a rectangle.

The format to represent a segmentation is simply a list of regions together with their type (text, halftone, ruling, etc), label, and geometrical representation, including their bounding box and list of polygonal vertices. The zone description format is the same for $S(I)$ and $SGTF(I)$, see Fig. 1.

Assumptions To determine whether a merging or splitting is horizontal or vertical, we assume that the level of skew in I is known, since very accurate and fast skew estimation techniques [4] are now available. We suppose here that the pages are read from top-left to bottom-right, and that the text lines are horizontal. The currently considered types of regions are: Text (including text columns, title and drop caps), Tables, Line-Art, Rulings and Halftone.

Examples Figure 5 shows examples of two documents, for which the segmentation results used to be rather poor, and provided interesting test cases for the benchmarking. For each one, we successively show an example of its associated ground truth and actual segmentation. Note that the displayed *GTF* is *one of the* multiple correct segmentations, namely the one with the biggest possible regions. For better visibility, we display here the *zones*, with different shades of gray, corresponding to their label L. The *regions* themselves correspond to the "black" pixels of the zones, labelled with L.

3 Ground truthing segmentation

Since, $SGTF(I)$ is meant to describe every possible correct segmentation of I, creating it is not a trivial task. For example, a text column may be extracted as one single region, or be split into its different paragraphs. Although independent from any segmentation technique, manual zoning must follow a set of strict guidelines.

Quality criteria for a segmentation A segmentation algorithm must be at least required to avoid endangering the next tasks.

At this low-level stage, the main constraint is therefore to provide a usable input for region ordering, which is closely related to the current line orientation $D^{\perp}$: splittings and mergings in the direction $D^{\perp}$ should be prohibited.

Maintaining font uniformity within a zone seems more logical though. However, some merges involving text zones with different fonts neither endanger region ordering nor OCR. This is the case, for example, for a drop cap, a math equation, or a subtitle merged with its attached adjacent text paragraph. In such cases, font uniformity is taken into account by defining corresponding regions cuts or merges.

One must also consider the type of region. Non-text zones, which are not processed by recognition, must be isolated. Tables, although composed of characters, must be isolated too, since their reading-order is different from that of a text paragraph. Isolating regions according to their use allows also to benchmark region classification.

Maximal set-based representation scheme Our ground truthing scheme relies on two main assumptions:

- It is preferable to have fewer regions in a segmentation; therefore, the *GTR* should be as large as possible.

- In this case, one can expect to have *one GTR* split in n regions, $n = 1$ being the "ideal" case.

We consider each *GTR* as a maximal set of elementary sets of pixels, merged in a given direction D. A *GTR* is thus a maximal set of vertically merged text lines. Such a "maximal set" based representation, in addition to be compact, enables a simple error identification strategy.

Related work An English machine-printed document database has been issued on a CD-ROM by Phillips et al., [9]. They designed a protocol for OCR ground truthing and also for drawing zone boxes, [10]. The chosen ZRS is the rectangle. Zones may be split in several rectangles. The smallest region corresponds to a single character, the largest one to a paragraph. This implicitly allows for horizontal splittings within a region. Among the considered GTR types are: text, math equations, tables, captions, figures, pictures, page numbers. The main low-level zone descriptors are: label, type, line orientation and dominant font size.

3.1 Protocol for manual zoning

Our system does not need any high-level information, and the only content-related attribute it uses is the number of black pixels in the zone. Fitting regions into polygons allows to build "maximal sets", which is usually impossible with rectangles without overlapping adjacent zones.

To build the ground truth files, we proceed as follows:

1. Divide the page into its largest structural units

2. Specify how these maximal units can be divided in such a way that the resulting segmentation remains correct.

The three steps of the construction of *GTF*(I) are then as follows:

1 - Locate "hard separators" Region ordering and classification related constraints allow the operator to specify non-crossable vertical or horizontal white spaces or black rulings. White spaces must be thicker than a line space. When the number NC of columns changes, a horizontal separator is virtually drawn in the space where NC changes and expanded until any black clusters are met. Within the derived zones, the vertical spaces are virtually drawn, mostly to decolumnize the text, and expanded until valued clusters are met. Columns within tables are however not separated.

2 - Formation of maximal GTR Within each of the derived zones, specify the local "line merging direction" D, orthogonal to the text lines. Then, merge the latter into a region, in the direction D. Do this until adding another line disturbs the reading order of the document. Surround the region with an isothetic polygon. Note that drop caps are merged to their bottom-right adjacent text zone.

3 - Encompass all correct segmentations

Define Region Cuts Allowed partitions of each ground truth region are explicitly defined by locating the authorized *region cuts* in the direction $D^{\perp}$. Note that if D is vertical (resp. horizontal), we talk about "horizontal" (resp. "vertical") cuts. For example, a *region cut* can be defined between any two consecutive paragraphs in a text column. If segmentation cuts a paragraph between two lines, the reading order is still correct, like it is the case for page1 (fig. 5), where *GTR* 1 has been split into regions 47 and 49 in $S(I)$. Such cuts would be too cumbersome to store. So they are implicitely tolerated and only slightly penalized, as long as no text-line is cut.

Define Region Merges as places where two distinct regions can be merged, with a slight associated penalty. This is the case for example between a text region and an adjacent headline or footer. *Region merges* are an exception to the strategy exposed earlier. They allow for vertical merging (with a small associated penalty), whereas *region cuts* allow for vertical splitting[2] (with no penalty). They are also referred to as *Page Cuts* in the *GTF* in Fig. 1.

3.2 Ground truth representation format

A ground truth file *SGTF*(I) is, in fact, an "ideal" segmentation composed of maximal regions, and where other possible segmentations are considered by defining explicit cuts and merges. The difference with $S(I)$ is that it includes extra information, appended to the list of maximal regions. This information is the number and list of the defined *region merges* and *region cuts*, both also called "softcuts". They are represented by a triplet (y, x_d, x_f), where y is the coordinate in the direction D, x_d and x_f the coordinates of its starting and end points respectively, in the direction $D^{\perp}$. For *region merges* (page cuts), the label of the two mergeable regions is specified.

The ground truth format we propose is therefore: simple, so that anyone can use it, general enough to represent any possible segmentation and expandable.

Figure 1 shows the ground truth file built for page 2. The "maximal" ground truth segmentations are shown in Fig. 5.

4 Identification of segmentation errors

Qualitative evaluation of segmentation is done in two steps.

1. Identification of operations done: overlapping *S(I)* and *SGTF*(I) allows to determine the operations to perform in one image in order to obtain the other one. We considered here *splitting* and *merging* in either the *vertical* or the *horizontal* direction.

2. Detection of segmentation errors: once the operations are identified, a set of rules allows to decide the level of correctness of each operation.

With each of these operations, a cost or penalty is associated:

[2] According to our terminology, a *vertical merging* of two regions correspond to the case where two regions that are on top of each other are merged. Similarly, splitting a region vertically means dividing it into two regions that are on top of each other.

□

```
FILENAME page2.gt
IMAGE_WIDTH 2551 IMAGE_HEIGHT 3301
IMAGE_XRES 300 IMAGE_YRES 300

TOTAL_REGIONS 11
TEXT_REGIONS 7
IMAGE_REGIONS 3
VRULE_REGIONS 0
HRULE_REGIONS 1

----- START_REGIONS -----

TYPE IMAGE
LABEL 0
BOUNDING BOX 0 812 199 731
NB POLYGON VERTICES 4
199 0
731 0
731 812
199 812
     ...............

TYPE TEXT
LABEL 10
BOUNDING BOX 3188 3256 195 587
NB POLYGON VERTICES 4
195 3188
587 3188
587 3256
195 3256

----- START_SOFTCUTS -----

NB PAGE_SOFTCUTS 1
PAGE_CUT BETWEEN REGIONS 9 10
3100 195 2271

NB REGIONS_WITH_SOFTCUTS 3
LABEL 7
NB_CUTS 6
1156 195 2263
1264 195 2263
1584 195 2263
1784 195 2263
1980 195 2263
2028 195 2263
     ...............

LABEL 9
NB_CUTS 5
1156 1595 2271
1304 1595 2271
1704 1595 2271
2012 1595 2271
2196 1595 2271
```

Figure 1: **Segmentation ground truth for** page2. **Until the definition of softcuts, the format is the same as for segmentation. Details are shown for the first and the last region only.**

good: no associated penalty. e.g., splitting over a region cut.

acceptable: small associated penalty. e.g., merging over a region merge, or vertical splitting of a text region outside of a region cut (but not across text lines)

bad: strongly penalized. e.g., any horizontal splitting or merging of a text region.

4.1 Identification of operations done

4.1.1 Overlap matching between S(I) and SGTF(I)

When a region in *S(I)* and a region in *SGTF(I)* have a non-empty intersection, we say they *match*. For each region in each image, the number and identity of the regions in the other image it matches with are computed. Having *one* region Rg in *SGTF(I)* matching with P regions Rs_p in *S(I)* indicates that Rg has been split by segmentation. Having *one* region Rs in *S(I)* matching with Q regions Rg_q in *SGTF(I)* indicates that those Rg_q have been partially or totally merged. Two match lists $\mathcal{M}(R_{gq})$ and $\mathcal{M}(R_{sp})$ are computed.

Yet, segmentation can both split a *GTR* and merge it partially or totally with other *GTR*. In page2, the *GTR* 7 is both split into regions $4, 5, 6, 7, 8, 9, 10$ in *S(I)*, and partially merged with *GTR* 8 and 9. This cannot be straightforwardly seen by scanning the two lists $\mathcal{M}(Rg_p)$ and $\mathcal{M}(Rs_q)$ separately. On the other hand, all operations can be straightforwardly identified by testing the overlap between *unions* of regions. Therefore, the algorithm tests a list $\mathcal{M}_k$ of pairs of overlapping unions of regions, derived from $\mathcal{M}(Rg_p)$ and $\mathcal{M}(Rs_q)$, with $\mathcal{M}_k = \{(\mathcal{G}_k, \mathcal{S}_k)\} = \{(\bigcup Rg_p, \bigcup Rs_q)\}$.

4.1.2 Identification and prior diagnosis

A *GTR* is defined as a *non-mergeable*, *maximal* set. Therefore, the most frequently expected case is to have *one GTR* matching with $n \geq 1$ regions in S. The operations done on the *GTR* and some segmentation errors are identified as follows: let

- $M_k = (\bigcup Rg_p, \bigcup Rs_q)$ be a match of unions of regions,

- $card(Ugtruth_k)$ (resp. $card(Usegm_k)$) be the number of regions in $\bigcup Rg_p$ (resp. $\bigcup Rs_q$).

Applying the rules illustrated in table 1 allows to immediately identify: noise improperly detected as a region, undetected regions, region splittings, region mergings, region splittings and mergings. Once all region matches have been found, each of them is considered individually. If it is not a one-to-one match, we verify if it corresponds to a splitting or a merging. Whether it is horizontal, vertical, or neither depends on the spatial layout of the regions of each union.

This way, a prior diagnosis is derived on the *GTR*. Figure 2, shows for page2 the matches of unions extracted by the algorithm, together with the number of regions in each union, its spatial structure and the prior diagnosis derived.

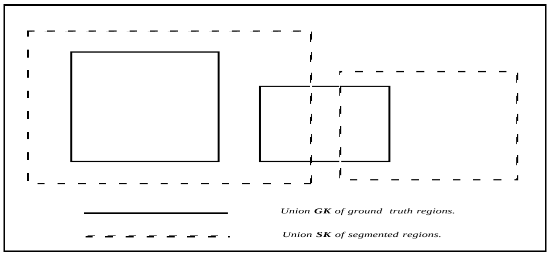
4.1.3 Spatial layout of a union

A union of regions can be of four kinds: a vertical alignment, a horizontal alignment, a mixed alignment (called "T-alignment") i.e. both vertically and horizontally aligned regions, or not aligned (0 or 1 region).

The possible adjacency relation between two regions R_A and R_B is 4-connected : top, down, right, left. It is determined by using their bounding boxes bA and bB. The horizontal overlap *h-over(A,B)* and the vertical overlap *v-over(A,B)* are computed. The direction of the adjacency relation minimizes *v-over* and *h-over*.

8-connected adjacency is not considered here. If *v-over* = *h-over*, the default direction is vertical. Indeed, the algorithm performs progressive rejection and late acceptance; therefore, an erroneously authorized relationship will be eventually rejected, where as an erroneous rejection is irreversible.

Fig. 3 shows a matching pair of unions of horizontally split and merged regions. In page2 (match # 0 in Fig. 2), region 25 in *S(I)* is formed by merging halftone *GTR* 0 and 2. But the merging also involves a part of text *GTR* 3. Region 1 of *S(I)* matches with the remaining part of *GTR* 3.


Union **GK** of ground truth regions.
Union **SK** of segmented regions.

- $\mathcal{G}_k$ has $N > 1$ regions ⇒ *GTR* merging.
- $\mathcal{S}_k$ has $P > 1$ regions ⇒ *GTR* splitting.

Figure 3: **Overlap matching of 2 unions of horizontally aligned regions, and identification of detected errors.**

4.2 Error detection strategy

The prior diagnosis allows an immediate rejection of regions in *S(I)* corresponding to noise, or regions in *SGTF(I)* which were not or only partially detected. The algorithm progressively rejects regions in both files, and makes additional tests on the remaining regions. For instance, in the "ideal" case where *one* *GTR* matches with *one* region in *S(I)*, it verifies that the overlap between the two regions is total.

To output a fair and reliable estimate of zoning quality, we need to consider both the number of correctly segmented *GTR*, and the number of correct regions in *S(I)*. For example, on page2, if we only looked at *SGTF(I)*, we would find that only three *GTR* are correctly segmented. Our estimate of segmentation quality would thus be unfairly low. On the other hand, looking at the segmentation regions, we would output

Table 1: **Straightforward identification of operations done by segmentation, by comparing the cardinal of the matching unions of regions.**

GTF/S(I)	$Card(U_{segm}) = 0$	$Card(U_{segm}) = 1$	$Card(U_{segm}) > 1$
$Card(U_{gtruth}) = 0$	No occurence	Junk Extraction	Junk extraction
$Card(U_{gtruth}) = 1$	Undetected region	Admissible region	Splitting
$Card(U_{gtruth}) > 1$	Undetected region	Merging	Merging/Splitting

a significantly higher rating. The error detection scheme thus proceeds in two steps:

1. The *GTR* are tested in order to know how they were processed by segmentation. If errors are found, the matching regions in *S(I)* are subject to rejection.

2. The regions in *S(I)* corresponding to *GTR* subject to given types of errors are tested in order to possibly cancel their rejection.

4.2.1 Test over the ground truth regions

Table 2 summarizes the simplified diagnosis of segmentation errors on *GTR*: "splitting" and "merging" refer to the operations that have been done on the *GTR*. For each operation, in either direction, the first line ("*Allowed*") indicates whether the operation is acceptable or not. The second line ("*Unless*") lists the cases where the opposite diagnosis is given.

For instance on page2, *GTR* 7 has been vertically split into 7 regions. Two of them (4 and 5), are horizontally merged with parts of *GTR* 8 and 9. Therefore, *GTR* 7, 8 and 9 are subject to rejection. On the other hand, *GTR* 0 has been horizontally merged with *GTR* 2. Both of them are halftones, therefore not processed by OCR; so the operation is acceptable. Yet, *GTR* 2 is also merged with a part of text-*GTR* 3, which is mistaken for halftone. Thus, for this match, only *GTR* 0 is considered "acceptable", but none of the matching regions in *S(I)*.

Errors in direction $D^{\perp}$ (horizontal)

Horizontal splitting: A horizontal alignment $\bigcup R_s$ of regions in *S(I)* indicates a horizontal splitting of *GTR*.

The regions are scanned pairwise and tagged as "bad", if any text is involved.

Exception: if the left region is a drop cap, then both regions are "acceptable".

Horizontal merging: A horizontal alignment $\bigcup R_g$ of *GTR* indicates a horizontal merge of these.

The regions of $\bigcup R_g$ are scanned pairwise and rejected. So is the common matching region in *S(I)*.

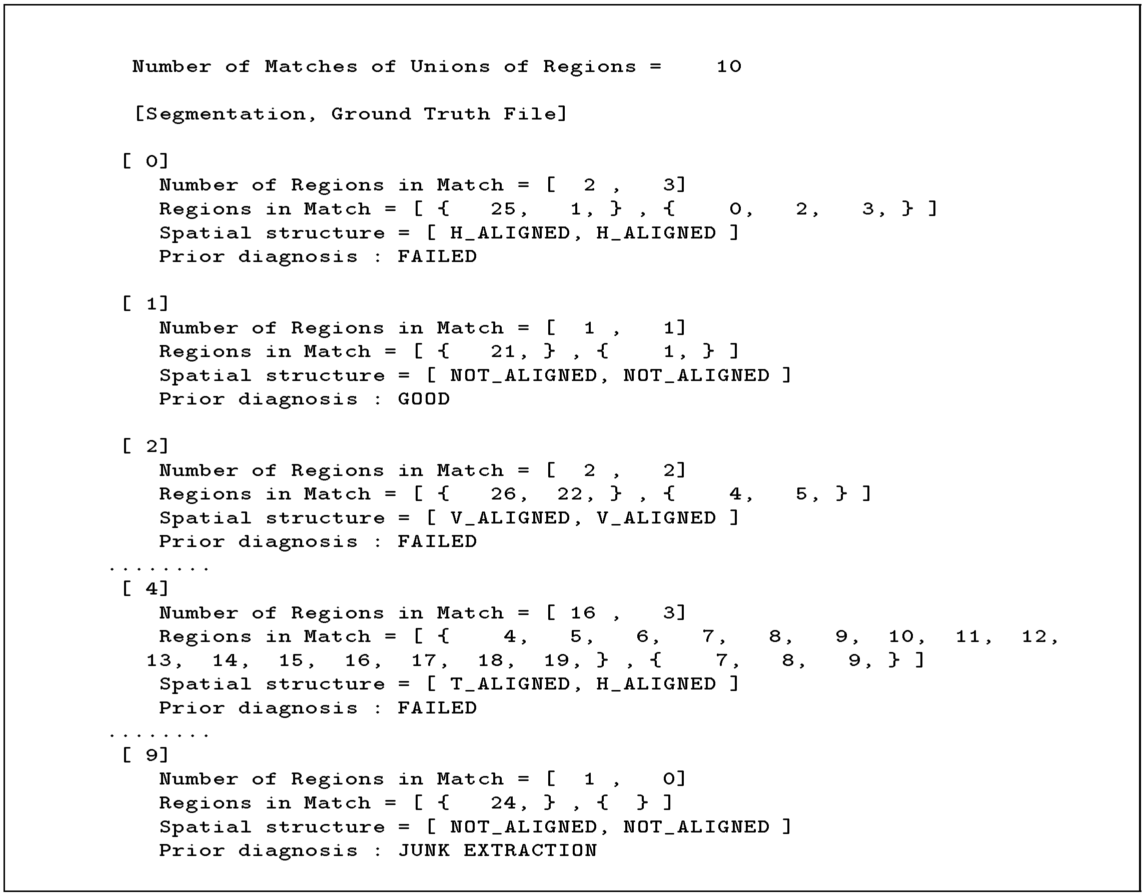
Exception: if both *GTR* are non-text regions, they are tagged as "acceptable", *unless* merged with text. So is the common match in *S(I)*, *unless* it also matches with a text-region in *SGTF(I)*.

In page2, the *GTR* 7, 8 and 9 have been partially merged. Their common matches in *S(I)*, regions 4 and 5 are rejected.

Errors in the direction D (vertical)

Vertical splitting: A vertical alignment in *S(I)* indicates a vertical splitting of a *GTR G*. In page1, *GTR* 1 has been correctly cut between lines, yielding regions 47 and 49 in *S(I)*. To be tagged as good, two vertically aligned regions A and B, with for example A being upon B, in *S(I)* must respect the following rules, (see figure 4):

- Their bounding boxes bA and bB must not intersect, otherwise it indicates a horizontal cut of a line, except for drop caps.



```
  Number of Matches of Unions of Regions =    10

  [Segmentation, Ground Truth File]

 [ 0]
    Number of Regions in Match = [  2 ,   3]
    Regions in Match = [ {   25,   1, } , {    0,   2,   3, } ]
    Spatial structure = [ H_ALIGNED, H_ALIGNED ]
    Prior diagnosis : FAILED

 [ 1]
    Number of Regions in Match = [  1 ,   1]
    Regions in Match = [ {   21, } , {    1, } ]
    Spatial structure = [ NOT_ALIGNED, NOT_ALIGNED ]
    Prior diagnosis : GOOD

 [ 2]
    Number of Regions in Match = [  2 ,   2]
    Regions in Match = [ {   26,  22, } , {    4,   5, } ]
    Spatial structure = [ V_ALIGNED, V_ALIGNED ]
    Prior diagnosis : FAILED
........
 [ 4]
    Number of Regions in Match = [ 16 ,   3]
    Regions in Match = [ {    4,   5,   6,   7,   8,   9,  10,  11,  12,
   13,  14,  15,  16,  17,  18,  19, } , {    7,   8,   9, } ]
    Spatial structure = [ T_ALIGNED, H_ALIGNED ]
    Prior diagnosis : FAILED
........
 [ 9]
    Number of Regions in Match = [  1 ,   0]
    Regions in Match = [ {   24, } , {  } ]
    Spatial structure = [ NOT_ALIGNED, NOT_ALIGNED ]
    Prior diagnosis : JUNK EXTRACTION
```

Figure 2: **Extracted matches of unions of regions in** page2, **identification of operation made and prior diagnosis.**

- The space within the rectangle $[y_{top}(bB), y_{bottom}(bA), x_{left}(bG), x_{right}(bG)]$, must contain a neglectible number of valued pixels.

- A specified region cut should lie between A and B. If not, they are tagged as "acceptable" instead of "good".

If one of the three rules is not respected, A, B and G are tagged as "bad".

Vertical merging: A vertical alignment of GTR indicates a vertical merging of them. The involved regions are tagged as "acceptable", if this occurs through a predefined *Region Merge* and as "bad" otherwise.

Errors in directions D and $D^{\perp}$

A T-aligned union U_s of regions in *S(I)*, matching with one single GTR, indicates both horizontal and vertical splittings of this GTR. When U_s matches with a union U_g of GTR, it both indicates mergings in the direction $D(U)$ of the alignment of U_g, and splittings in at least the direction $D(U)^{\perp}$, as it is the case in page2 (match # 4 in Fig. 2). This often occurs in non homogeneous zones, like graphics, halftones and text. Handling such errors depends on the type of region. Tables which are not cell-tables should be read line by line, and be contained in one single region. Therefore, any horizontal splitting in a GTR which is a table yields a rejection of all correspond-

Table 2: **Error diagnosis rule, on operations done on ground truth regions.**

Operation	**Decision**	*Horizontal*	*Vertical*
Splitting	*Allowed:*	NO	YES
	Unless:	halftones only text and drop cap	lines split horizontal merges
Merging	*Allowed:*	NO	NO
	Unless:	halftones only	-through page cuts

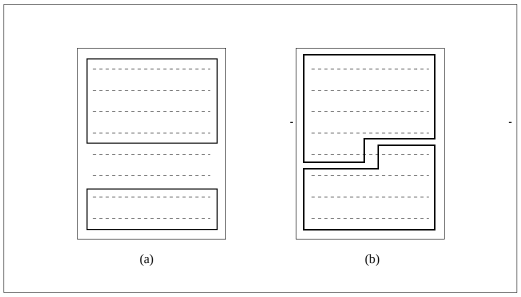


Figure 4: **Errors detected in a vertical alignment: (a) by detecting a significant number of valued pixels between 2 regions, (b) by detecting a bounding box overlap.**

ing regions in *S(I)*. For text paragraphs, a more detailed diagnosis is necessary.

4.2.2 Additional tests on some of the segmentation regions

As illustrated in example page2, tagging a *GTR* as badly segmented should not lead to a systematic rejection of *all* the corresponding regions in *S(I)*. For instance, among the regions in *S(I)* corresponding to *GTR* 7, regions 6, 7, 8, 9 and 10 are correct.

In such cases we need to examine the corresponding segmentation regions one after the other and re-tag them. A typical case is: if U_g is a union of *horizontally* aligned regions, and if each region of U_g is *vertically* split by segmentation. The algorithm rejects the regions in *S(I)* matching with several regions in U_g (horizontal merging of *GTR*), and accepts the others if they are correctly vertically split.

4.3 Quantitative performance evaluation

Designing a fitness measure between a segmentation and the corresponding ground truth file is equivalent to defining a "distance" between two sets of non ordered regions. The score must enable to compare different segmentations, as well as to measure the quality of the segmentation. The evaluation is currently provided by two quantities:

PageScore a quality ratio reflecting the effects of segmentation on the regions and their contents.

PageCost reporting the number of operations (split or merge in either directions, re-classification) needed to correct the segmentation, in order to get to the "nearest" acceptable solution.

Like for qualitative evaluation, both *GTF*(I) and $S(I)$ are considered. For each of the two sets of regions, we currently compute:

- the ratio $S_{reg}()$ of the number of "good" and "acceptable" regions to the total number of regions in the page.

- the ratio $S_{pix}()$ of the cumulated size of "good" and "acceptable" regions to the total number of labeled pixels.

On the ground truth regions, we calculate:

- The number O_{err} of "bad" operations.

- The number O_{acc} of "acceptable" operations.

PageScore is then equal to:

$w_{set}[w_{reg} * (S_{reg}(GTF(I)) + S_{reg}(S(I))) + w_{pix} * (S_{pix}(GTF(I)) + S_{pix}(S(I)))]$

PageCost is then equal to:

$$w_{err} * O_{err} + w_{acc} * O_{acc}$$

PageScore must be maximized up to 1, where as *PageCost* must be minimized down to 0. Currently the the algorithm uses $w_{set} = 0.5$, $w_{reg} = 0.5$, $w_{pix} = 0.5$, $w_{err} = 1$ and $w_{acc} = 0.5$.

For instance, for page1, *PageScore* is equal to 0.498 and *PageCost* to 32.5. For page2, *PageScore* is equal to 0.707 and *PageCost* to 10.5.

4.4 Output of the benchmarking

Our benchmarking approach provides a multi-level output for a given document test suite. Benchmarking results for page1 and page2 are provided in Fig. 6 and 7. The algorithm is simple and fast. An image of 1275 × 1650 pixels, takes around 40 seconds total time on a SPARC-10.

Overall output of the test suite: a numerical score is provided over the test suite. For the attached example, this score is 6.02/10. It simply represents the average *PageScore* obtained on the suite. Likewise, the overall cost is the average *PageCost* over the suite, in this case, 21.5.

Page-level output: is provided both for *S(I)* and *SGTF(I)*. For each of the two files, regions are classified as "good", "acceptable" and "bad", according to the error diagnosis. Errors are classified as "horizontal", "vertical" and "junk extraction". "Missed extraction" represents the number of *pixels* in the *GTF*, that have been missed by segmentation, regions being rarely missed.

Region-level output: is only detailed for the incorrectly segmented *GTR*. For each of them, the benchmarking algorithm provides: the label and type, the number of pixels missed by segmentation ("NbNonSegmPix") if significant, the number of regions it has been split in ("NbSegments"), the diagnosis on the splittings and mergings done on that region. Yet, like for the *GTR* 1 in page1, the errors in a region are not always detailed. For a table, horizontal splittings are enough to reject it, as well as the corresponding regions in $S(I)$.

5 Conclusion

We have presented a bitmap-level automatic scheme to benchmark page segmentation algorithms on machine-written mixed text/halftone binary document images. It provides an accurate qualitative diagnosis of segmentation errors, from which, a quantitative evaluation is derived.

Segmentation here is benchmarked, only by using segmentation results. By comparing sets of pixels rather than their surrounding polygons, this method is independent of any zone representation scheme. Contrary to previously proposed methods, our approach is region-based, allowing to categorize and specify the error types and to be independent of OCR errors.

Some categories of errors are highly correlated with a given type of region or context. For example, tables are typically split into many incorrect sub-regions; errors often occur in zones where the contrast in spatial density between two very close adjacent regions is low. Knowing the strengths and weaknesses of a segmentation algorithm on given types of regions can help one to choose a technique according to the type of documents to process.

Future work will aim at designing a more reliable numerical measure of segmentation quality, essential to anyone wishing to fine tune segmentation parameters or test available techniques on various kinds of input documents.

A ground truthing scheme is also proposed. An agreement upon a standard ground truth representation scheme is now necessary to enable a broad use of our system. Our results are very promising, and our system now needs to be tested and refined on a broader range of test suites, produced by different algorithms.

References

[1] ARPA. Dimund workshop on page decomposition, character recognition and data standards. Harpers Ferry WV, August 9–11 1993.

[2] H.S. Baird. *Structured Document Image Analysis*, chapter Document Image Defect Models, pages 546–556. Springer Verlag, 1992.

[3] H.S. Baird, S.E. Jones, and S.J. Fortune. Image segmentation by shape-directed covers. In *10th Int. Conf. on Pattern Recognition (ICPR'90)*, pages 820–825, Atlantic City, NJ, June 1990.

[4] S.C. Huids, J.L. Fisher, and D.P. d'Amato. A document skew detection method using runlength encoding and the hough transform. In *Proc. of 10th Int. Conf. on Pattern Recognition (ICPR'90)*, pages 464–468, 1990.

[5] J. Kanai, T.A. Nartker, S.V. Rice, and G. Nagy. Performance metrics for document understanding systems. In *2nd International Conference on Document Analysis and Recognition (ICDAR'93)*, pages 424–427, 1993.

[6] J. Kanai, S.V. Rice, and T.A. Nartker. A preliminary evaluation of automatic zoning. Annual research report, ISRI, University of Nevada, 4505 Maryland Parkway, BOX 454021, Las Vegas 89154-4021, 1993.

[7] M. Nadler. A survey of document segmentation and coding techniques. *Computer Vision Graphics and Image Processing*, (28):240–262, 1984.

[8] T. Pavlidis and J. Zhou. Page segmentation and classification. *Computer Vision Graphics and Image Processing*, 54(6):484–486, November 1992.

[9] I.T. Phillips, S. Chen, J. Ha, and R.M. Haralick. English document database design and implementation methodology. In *2nd Annual Symposium on Document Analysis and Information Retrieval*, pages 65–104, Caesars Palace Hotel, Las Vegas, NA, USA, April 26-28 1993.

[10] I.T. Phillips, S. Chen, and R.M. Haralick. Database zone label definition and examples. Technical report, University of Washington, Intelligent Systems Lab., Seattle, WA 98195, 1993.

[11] S. Randriamasy and L. Vincent. Benchmarking page segmentation algorithms. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'94)*, pages 411–416, Seattle WA, June 20-23 1994.

[12] S. Randriamasy, L. Vincent, and B. Wittner. An automatic benchmarking scheme for page segmentation. In *IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, San Jose, CA, USA, February 6-10 1994.

[13] S.V. Rice, J. Kanai, and T.A. Nartker. An evaluation of ocr accuracy. Technical report, ISRI, University of Las Vegas, 1993.

[14] S.N. Shrihari and J.J. Hull. *Encyclopedia of Artificial Intelligence*, chapter Character Recognition, pages 138–150. Wiley Interscience, NY, 1992.

[15] Machine Vision and Applications. Document image analysis techniques. *Special Issue*, 5(3), 1992.

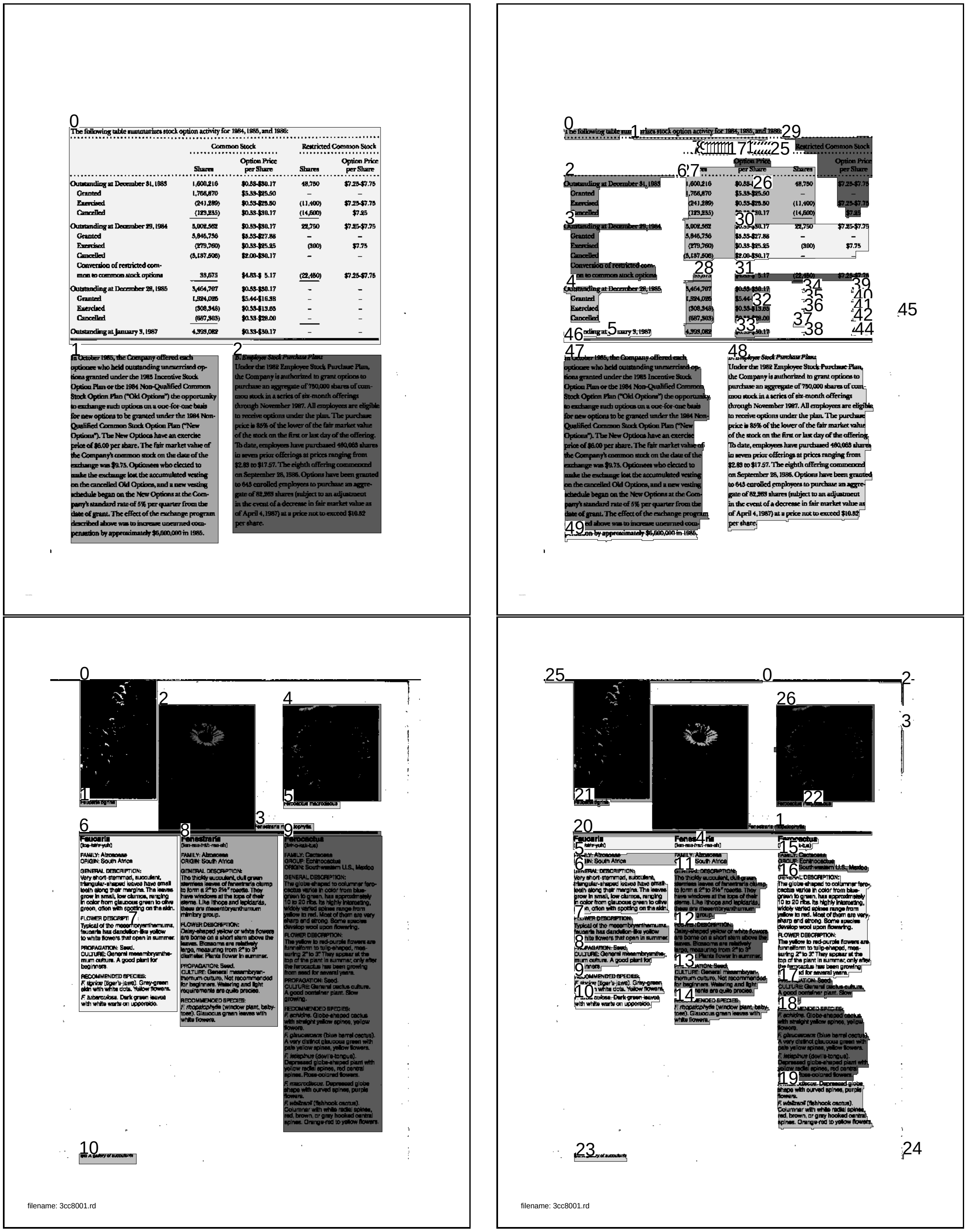


Figure 5: *Ground truth (left) and obtained (right) segmentations on documents "*`page1`*" (top pair) and "*`page2`*" (bottom pair).*

```
++++++++++++++++++++++++++++++++++++++++++++
 SEGMENTATION BENCHMARKING of PAGE : table18
++++++++++++++++++++++++++++++++++++++++++++

 =============================================
      PAGE  OUTPUT (file: table18)
 =============================================

  **** PAGE SCORE (over 1) = 0.498 ****
  **** PAGE COST  =   32.5  ****

  Region score = 0.363

  Pixel score of segm = 0.632

  Bad Operations: 32

  Acceptable Operations: 1

  Nb Regions [SEGM] =   50
  Nb Good Regions [SEGM] =   1
  Nb Acceptable Regions [SEGM] =   2
  Nb Bad Regions [SEGM] =  47

  Nb Regions [GR_TRUTH] =   3
  Nb Good Regions [GR_TRUTH] =   2
  Nb Acceptable Regions [GR_TRUTH] =   0
  Nb Bad Regions [GR_TRUTH] =   1

 ========================================
     SEGMENTATION
 ========================================

 GOOD REGIONS: 2
 ============
  Region =    48   Type = TEXT

 ACCEPTABLE REGIONS: 1
 ==================
  Region =    47   Type = TEXT
  Region =    49   Type = TEXT

 ERRORS IN : SEGM
 ================
 HORIZONTAL ERRORS:   40
 type TEXT : 40
 (not enumerated due to lack of space)

 VERTICAL ERRORS:    6
 type TEXT : 6
 (not enumerated due to lack of space)

 JUNK EXTRACTION :
  Region =    45   Type = TEXT

 ========================================
      GROUND TRUTH
 ========================================

 GOOD REGIONS:
 ============
```

```
  Region =     1   Type = TEXT
  Region =     2   Type = TEXT

 SEGMENTATION ERRORS
 ===================
 ** SPLITTINGS :    1
 ------------
 HORIZONTAL :
  Region =     0   Type = TABLE


 SEGMENTATION ERRORS - DETAILED :
 ===============================

 HORIZONTAL ERRORS:    1

 REGION :     0   Type : TABLE
 * NbNonSegmPix : 496
 * NbSegments   :  46
 * Splittings :   H_ERROR:  32
 * Mergings   :

 VERTICAL ERRORS:    0

 MISSED PIXELS : NONE


++++++++++++++++++++++++++++++++++++++++++++
 SEGMENTATION BENCHMARKING of PAGE : 3cc8001
++++++++++++++++++++++++++++++++++++++++++++

 ==========================================
      PAGE  OUTPUT (file: 3cc8001)
 ==========================================

  ****  PAGE SCORE (over 1) = 0.707 ****
  ****  PAGE COST =   10.5  ****

  Region score = 0.497

  Pixel score of segm = 0.917

  Bad Operations: 10

  Acceptable Operations: 1


  Nb Regions [SEGM] =  27
  Nb Good Regions [SEGM] =  17
  Nb Acceptable Regions [SEGM] =   0
  Nb Bad Regions [SEGM] =  10

  Nb Regions [GR_TRUTH] =  11
  Nb Good Regions [GR_TRUTH] =   3
  Nb Acceptable Regions [GR_TRUTH] =   1
  Nb Bad Regions [GR_TRUTH] =   7

 ========================================
    SEGMENTATION
 ========================================
```

Figure 6: *Output of the benchmaking for* page1 *(table18) and* page2 *(3cc8001)*

```
GOOD REGIONS: 17
============
type TEXT : 17
(not enumerated due to lack of space)

ERRORS IN : SEGM
================
HORIZONTAL ERRORS:    4
 Region =     1   Type = TEXT
 Region =     4   Type = TEXT
 Region =     5   Type = TEXT
 Region =    25   Type = HALFTONE

VERTICAL ERRORS:    2
 Region =    22   Type = TEXT
 Region =    26   Type = HALFTONE

JUNK EXTRACTION :    4
 Region =     0   Type = TEXT
 Region =     2   Type = TEXT
 Region =     3   Type = TEXT
 Region =    24   Type = TEXT

==========================================
   GROUND TRUTH
==========================================

GOOD REGIONS:
============
 Region =     1   Type = TEXT
 Region =     6   Type = H_RULE
 Region =    10   Type = TEXT

ACCEPTABLE REGIONS:
==================
REGION :     0   Type : HALFTONE

* NbNonSegmPix : 684
* NbSegments   :   1
* Splittings :
* Mergings   :   H_CORRECT:   1

SEGMENTATION ERRORS
===================
** SPLITTINGS :    2
------------
HORIZONTAL :
 Region =     3   Type = TEXT

VERTICAL :
 Region =     5   Type = TEXT

** MERGINGS :    5
----------
HORIZONTAL :
 Region =     2   Type = HALFTONE
 Region =     3   Type = TEXT
 Region =     7   Type = TEXT
 Region =     8   Type = TEXT
 Region =     9   Type = TEXT
```

```
VERTICAL :
 Region =     4   Type = HALFTONE
 Region =     5   Type = TEXT


SEGMENTATION ERRORS - DETAILED :
===============================

HORIZONTAL ERRORS:    5

REGION :     2   Type : HALFTONE
* NbNonSegmPix :   0
* NbSegments   :   1
* Splittings :
* Mergings   :   H_ERROR:   1

REGION :     3   Type : TEXT
* NbNonSegmPix :   0
* NbSegments   :   2
* Splittings :   H_ERROR:   1
* Mergings   :   H_ERROR:   1

REGION :     7   Type : TEXT
* NbNonSegmPix :   0
* NbSegments   :   7
* Splittings :
* Mergings   :   H_ERROR:   1

REGION :     8   Type : TEXT
* NbNonSegmPix :   0
* NbSegments   :   6
* Splittings :
* Mergings   :   H_ERROR:   2

REGION :     9   Type : TEXT
* NbNonSegmPix :   0
* NbSegments   :   6
* Splittings :
* Mergings   :   H_ERROR:   1


VERTICAL ERRORS:    2

REGION :     4   Type : HALFTONE
* NbNonSegmPix :   0
* NbSegments   :   1
* Splittings :
* Mergings   :   V_ERROR:   1

REGION :     5   Type : TEXT
* NbNonSegmPix :   0
* NbSegments   :   2
* Splittings :   V_ERROR:   1
* Mergings   :   V_ERROR:   1

MISSED PIXELS : NONE
```

Figure 7: *Output of the benchmaking for* page1 *(table18) and* page2 *(3cc8001)*