

An automatic benchmarking scheme for page segmentation

Sabine Randriamasy^{*1 2}, Luc Vincent^{**}, Ben Wittner^{**}

* Harvard Robotics Lab., 29 Oxford street, Cambridge MA 02138

**Xerox Imaging Systems, 9 Centennial Drive - Peabody MA - 01960

ABSTRACT

An automatic bitmap-level, set-based benchmarking scheme for page segmentation, comparing results with predefined "ground truth files" containing all the possible correct solutions, is presented here. A successful page segmentation is a necessary precondition for a document recognition process to be successful. There exists currently no robust method to benchmark segmentation techniques. Moreover, there exist no standard to represent all correct segmentations of a page. The problems addressed here are: design methods to describe all possible correct segmentations for a given page and design methods to compare two segmentations.

The proposed segmentation ground truth representation scheme defines ground truth text regions as non-mergeable maximal sets of text lines, merged in a language-dependent direction. It includes the other possible correct segmentations in that authorized cuts in the region are explicitly specified. At this low-level stage, quality criteria for a page segmentation are mainly defined as providing correct input for region ordering and classification. The qualitative and quantitative evaluation method tests the overlap between the two sets of regions. In fact, the regions are defined as being the black pixels contained in the derived polygons. The algorithm is simple and fast, and provides an explicit multi-level output for each segmentation.

1 INTRODUCTION

Document Understanding is a growing field, both in terms of applications and the research efforts being carried out, see [15, 8]. The field is, however, very complex due to the extreme diversity of documents to be processed as well as their visual quality.

The process of document recognition is usually decomposed in the following main steps:

- Page segmentation: which is concerned with the decomposition of a document page into its structural and logical units, like text, halftones, graphics, rulings, tables, etc.,
- Region ordering: whereby the the reading order of the page is derived,
- Recognition: whereby the semantic contents of the significant extracted regions is recognized, e.g. Optical Character Recognition (OCR) is run on the extracted text regions.
- Format/layout analysis: where such things as margins, paragraph endings, tabulations and indentations are determined.

Over the past twenty years, research efforts on document recognition have been focused on recognition; thus improving OCR accuracy [14, 2]. These improvements were made possible by the availability of reliable benchmarking tools for OCR. In fact, estimating the quality of a recognition algorithm is rather easy: the solution is unique, and one can simply compare the text output with a pre-stored "ground truth file", containing the ideal output [13].

Page segmentation is often seen as the central piece of a document recognition system, since it is a necessary precondition for the other steps in the process: identification of region types allows the system to make the appropriate treatment for each of them. This step also provides useful information on the characteristics of these regions, such as shape, size and type, without which no serious format analysis is possible. Lastly, without correct segmentation, the *lexical order* of the page may be outputted in a wrong sequence by the ensuing region ordering step. Therefore, estimating the quality of a page segmentation is a crucial issue.

¹Supported by a grant from the french ministry of research and technology

²Current address: Inria-Rocquencourt, Domaine de Voluceau, B.P. 105, 78153 Le Chesnay

Page segmentation techniques are now of very good quality, [8, 3, 9, 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 segmentation technique is optimal, and there is now a wide agreement upon the need for a reliable method to benchmark page segmentation algorithms, [1]. Moreover, a segmentation algorithm must be tested on hundreds of input pages, which stresses the need for an automatic, reliable and explicit tool.

Unlike character recognition, designing a scheme for estimating the quality of a page segmentation algorithm is neither simple nor straightforward. The optimal segmentation is not necessarily well defined and unique for a page. Usual region characteristics, are not sufficient to compare segmentations: two sets of regions similar in shape are not necessarily of equivalent quality. Errors are not uniform across the page: the cost of merging or splitting regions is orientation-dependent: text lines may be vertical or horizontal. Moreover, zone representation schemes are not standardized, and therefore cannot be compared geometrically.

The only known automated way to measure the performance of segmentation algorithms [6], provides a quantitative evaluation by comparing OCR outputs, see § 2.1.

Yet, page decomposition must be evaluated independently of the rest of the recognition tasks, by a system providing an accurate and explicit specification of detected errors. As far as we know, the only method known to work reliably is visual inspection of the regions extracted by segmentation. This way, evaluation can only be performed on a necessarily limited range of test documents; in addition, no global nor statistical performance analysis of a page decomposition system is possible.

The following issues are addressed in this paper:

- design methods to describe all possible correct segmentations for a given page,
- design low-level methods to compare two segmentations.

An automatic image-level, region-based benchmarking scheme for page segmentation, which compares results with predefined “ground truth files” containing all the possible correct solutions is presented here. The algorithm provides an explicit multi-level output for each segmentation, from the page to the region level, where segmentation errors are explicitly detailed.

The present approach is set-based in that, regions are in fact equivalent to the set of black pixels contained in the polygons derived by segmentation. This frees one from committing to any particular zone representation scheme. Together with the comparison method, we introduce in § 3, a segmentation ground truthing scheme, which encompasses all the possible correct segmentations of a given page.

At this low-level stage, defining what a good segmentation is expected to be, depends closely on the ensuing tasks: the goal of the algorithm is defined here as providing correct input for region ordering and classification. Any region hindering this goal must be rejected.

For each page I , the set of all the correct segmentations is described in a “segmentation ground truth file” ($SGTF(I)$), containing a set of “ground truth regions” GTR . Each of these is equivalent to a *non-mergeable* and *maximal* set of lines, gathered in a language-dependent direction D^3 , while respecting region ordering. The non-uniqueness of the optimal segmentation associated with I mainly lies in the fact that basically, text columns may be split in paragraphs, sub-paragraphs and titles. So within each GTR , region cuts are defined in order to encompass all possible correct solutions. Cuts between lines are not defined, although tolerated as long as correctness is preserved. Page cuts, allowing in some exceptions merges between GTR are also defined. A set of guidelines on how to generate a $SGTF$ is provided.

Comparison of a segmentation S and its corresponding $SGTF$ G , is done by testing the region overlap between the two sets of regions, and comparing their contents in I , see § 4. A list $M_k = \{(\bigcup Rg_p, \bigcup Rs_q)\}$ of pairs of overlapping unions of regions is established. Since the GTR are maximal and not mergeable, a straightforward identification of the operations done on the GTR is possible.

The detection of segmentation errors is based on the spatial relations between the regions in each matching union of M_k , and the predefined cuts, see § 4.2. For each page, a rough quantitative measure is then derived.

The presented algorithms have been implemented in C language on a SPARC-station. They were tested on about a dozen of different machine written multicolumn mixed image/text document images.

³which is *vertical* for european languages and *horizontal* for asian languages

2 APPROACHES TO BENCHMARKING SEGMENTATION

2.1 High level approach: text-based benchmarking

The only known automated way to measure the performance of segmentation algorithms has been proposed by Kanai et al., [7] together with an *automatic zoning metric*. The principle is to compute the minimal number of text insertions, deletions, and block moves necessary to transform the OCR output into the reference text. The cost of segmentation *alone* is derived by comparing the costs corresponding to manually and automatically zoned pages.

The advantage of such a method is that it frees one from relying on any particular zone representation scheme. Moreover, there is no need for specifying any particular format for segmentation ground truth.

However, the output is a number of editing operations from which it is impossible to derive any specification on the *type* of error made by segmentation. Furthermore, OCR results also depend on region ordering, and there is, in general, no way to know whether the OCR errors are due to segmentation or to region ordering.

2.2 Lower-level approach: comparison of ordered lists of pixels

Another possibility is to adapt the technique described above to the bitmap level, by manipulating strings of pixels rather than strings of characters. The system could compare the order in which valued pixels of the image are scanned for the ground truth segmentation and the segmentation to be evaluated.

As the case in § 2.1, but without using OCR, such a method would derive a minimal number of: insertions (pixels missed by segmentation), deletions (non-significant pixels, (e.g. noise, halftone) mistaken for text), block moves (segmentation or region ordering errors). No segmentation ground truth or zone representation scheme would be necessary.

However, similarly as above, such a method would not be able to provide an explicit specification of segmentation errors. Moreover, it is still linked to region ordering. Last, the strings of pixels to compare are extremely long, and designing efficient algorithms to match them seems very difficult.

2.3 Proposed region and bitmap-based approach

We have opted for a region-based method, capable of performing the evaluation of segmentation only by using segmentation results only.

Similarly to OCR error identification, given a segmentation and the associated segmentation ground truth, evaluation first consists of determining the operations to perform on one file in order to obtain the other one. Instead of additions, deletions and block moves like in [7], the operations we have to 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 the *regions* in the automatically and manually zoned images. Compared with ASCII text or ordered lists of pixels, regions are a much richer, and more synthetic way to describe segmentations. Using them, segmentation errors will be easier to find and to characterize.

Our proposed method proceeds in two main steps:

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

2 - Matching of segmentation result against ground-truth file A region-based comparison algorithm is then used to match the ground truth file against the segmentation result to be evaluated.

A segmentation ground truthing scheme and a representation format are introduced in § 3.

Bitmap-level set-based approach The existing page decomposition systems use various region or zone representation schemes (ZRP). The main ones are: bounding rectangles, nested bounding rectangles, polygons and piecewise rectangles. ZRP are not standardized, therefore geometrical comparison of zones is not feasible [6].

Our method is independent of any ZRP: we define regions as being the set of pixels 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 The comparison of the two sets of regions $S(I)$ and $GTF(I)$ enables a detailed and accurate list of segmentation errors, together with their types and associated penalty.

The benchmarking algorithm is capable of providing, among other things: the number of regions that have been incorrectly merged or split and the type of merging or splitting (horizontal, vertical), the number of meaningless regions extracted and the type of incorrectly segmented regions.

A global rating of segmentation quality of can then be derived for the page. It is then possible to derive a global rating, together with statistics, over the whole test suite.

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

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

The region representation format Basically, $S(I)$ and $SGTF(I)$ are two sets of regions to be compared. The format we have chosen to represent them 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. Figure 1 shows the $SGTF$ built for example page 2. The region description format is exactly the same for $S(I)$ and $SGTF(I)$.

Let us recall, however, that in our system, *regions* are equivalent to *set of black pixels*. Geometrical information is only used at the beginning, to label the black pixels of I , into sets of pixels belonging to the same polygon. This is done prior to benchmarking itself.

We have opted for polygons, since it is both more realistic than rectangles, and encompasses most of the possible representations. We, however, impose that each polygon has either vertical or horizontal edges, which corresponds to a set of *merged* piecewise rectangles. The reason is that we implicitly consider a text region as a set of merged lines, each line corresponding to a rectangle.

Assumptions Note that in order to determine whether a merging or splitting is purely horizontal or vertical, we need to calculate the level of skew in the original document. Very accurate and fast skew estimation techniques have been recently developed, [5, 4] and enable to assume prior deskewing of the image.

In the present paper, we suppose that all the pages are read from top-left to bottom-right, like european languages and middle eastern documents, where text lines are horizontal. It would be easy yet, to adapt this method to some asian languages, where text lines may be read in both vertical and horizontal directions within a page.

We currently consider the following types of regions: Text (including text columns, title and drop caps), Tables, Line-Art, Rulings and Halftone.

3 SEGMENTATION GROUND TRUTHING

Given I , several different segmentations of I may be considered correct. For example, a text column may be extracted as one single region, or may be split into its different paragraphs. In other words, $GTF(I)$ is meant to encompass every possible correct segmentation of I . For this reason, creating a ground truth file $GTF(I)$ is not a trivial task.

Manual zoning is a hand-made correct segmentation. Although independent from any page segmentation technique, it must follow a set of strict guidelines.

Quality criteria for a segmentation A segmentation algorithm must be at least required to avoid endangering the ensuing tasks. At this low-level stage, the main low-level constraint is therefore to provide a usable input for region ordering. This task depends closely on the current line orientation D^\perp , which for european languages, is the horizontal direction. Therefore, no splitting or merging in the direction D^\perp should be done by segmentation.

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 zone merged with its attached adjacent text paragraph. In such cases, font uniformity will be taken into account by defining corresponding regions or page cuts.

Another criterion lies in the type of the region. Non-text zones, which are not processed by the recognition task, must be separated from text zones. Although composed of characters, tables must be separated from any other region. This also allows to benchmark region classification.

Maximal set 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, the most frequently expected case is having *one GTR* split in n regions by segmentation, $n = 1$ being the “ideal” case.

A synthetic representation scheme is to extract each ground truth region as a maximal set of minimal sets of pixels, merged in a given direction D . Our ground truth regions are thus maximal sets of vertically merged lines, in other words, text columns. Such a “maximal set” based representation scheme, in addition to be compact, enables a simple error identification strategy.

Related work In order to allow the OCR and document understanding community to test and evaluate their algorithms, an english document database has been issued on a CD-ROM by Phillips et al.,. It includes machine-printed documents of various types and visual quality. The implementation methodology is presented in [10]. It is mainly focused on OCR ground truthing.

It also includes, however, a protocol for drawing zone boxes, see [11]. The chosen zone representation scheme is the rectangle. Zones may therefore be split in several rectangles. The smallest region corresponds to a single character and the largest one to a paragraph. This implicitly allows a horizontal splitting of a region. Among the considered ground truth region types are: text, math equations, tables, captions, figures, pictures, page numbers. The main region-related zone descriptors are: label, line orientation, dominant font size and type.

3.1 Protocol for manual zoning

Our segmentation benchmarking 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. In addition, regions are fitted into polygons, allowing thus the use of “maximal sets”, which would be impossible with rectangles.

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 would still be 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 lines. White spaces must be thicker than a line space. Horizontal spaces are first located: when the number NC of columns changes, a horizontal separator is virtually drawn in the space where NC changes, expanding until columns, or any valued clusters, are met. Within the derived zones, the vertical spaces are virtually drawn, mostly to separate columns, and expanded until valued clusters are met. However, within tables the columns are not separated, in order to maintain a correct reading order.

2 - Formation of maximal ground truth regions Within each of the derived zones, specify the direction D orthogonal to the text lines. Then, gather the text lines into regions, according to the local “line merging direction” D and constraints related to region ordering such as type and rough font uniformity, by surrounding them with isothetic polygons. Note that drop caps are merged to their bottom-right adjacent text zone, in order to maintain a correct reading order.

3 - Encompass all possible correct segmentations

Define Region Cuts Allowed partitions of each ground truth region are explicitly defined by storing the locations of authorized *region cuts* in the direction D^\perp ; for example, there is no penalty attached with cutting a text column between two paragraphs, thus there should be a *region cut* between any two consecutive paragraphs in a text column. On the contrary, if a segmentation was to cut a paragraph in-between two lines, the benchmarking system should penalize it. Note that if D is vertical (resp. horizontal), we talk about “horizontal” (resp. vertical) cuts.

Define Page Cuts as places where two distinct regions may be merged, the penalty associated with it being relatively low. This is the case for example between a text region and an adjacent headline or footer. These *page*

cuts constitute an exception to the strategy exposed earlier, in that they allow for vertical merging (with a small associated penalty), whereas *region cuts* allow for vertical splitting⁴ (with no penalty).

Region cuts and page cuts 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 .

3.2 Ground truth representation format

A ground truth file $SGTF(I)$ describes a complete set of correct partitions of I into regions of different types. It is, in fact, an “ideal” segmentation composed of maximal regions, where other possible segmentations are considered by defining explicit cuts.

The difference with S is that it includes extra information, appended to the list of maximal regions. This information is the number and list of the defined *page cuts* and *region cuts*, also called “softcuts”.

The ground truth format we propose is therefore:

- simple, so that anyone can use it,
- general enough to represent any possible segmentation,
- expandable.

Examples In figure 3 , we show examples with two test documents, for which the segmentation results were rather poor. For each of them, we successively give its associated ground truth file and an example of its segmentation.

Figure 1 shows the ground truth file built for example **page 2**. Up to the definition of “softcuts”, the format is exactly the same as for a segmentation.

```

FILENAME page2.gt                               587 3256
IMAGE_WIDTH 2551 IMAGE_HEIGHT 3301             195 3256
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

----- 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 showed for the first and the last region only.

4 Identification of segmentation errors

Qualitative evaluation of segmentation is done in two steps.

⁴According 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.

1. Identification of operations done: overlapping $S(I)$ and $SGTF(I)$ allows to determine the operations to perform in one file in order to obtain the other one. The operations considered here are *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:

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

acceptable: small associated penalty. e.g., merging over a page cut, 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 of a text region, or any horizontal merging of text regions.

To output a fair and reliable estimate of zoning quality, we need not only to look at the number of ground truth regions that have been correctly segmented, but also at the number of segmentation regions that are considered correct. A comprehensive error detection strategy will be proposed in [12].

4.1 Identification of operations done

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 is computed. *one* region Rg in $SGTF(I)$ matching with P regions Rs_p in S indicates that Rg has been split by segmentation. *one* region Rs in S matching with Q regions Rg_q in $SGTF(I)$ indicates that those Rg_q have been partially or totally merged. Two match lists $\mathcal{M}(Rg_p)$ and $\mathcal{M}(Rs_q)$ are computed.

Yet, as it is illustrated in example [page2](#) (see Fig. 3), it often occurs that P regions in one image match with Q regions in the other image. In this page, the union of regions $\{0, 2, 3\}$ in the $SGTF$ matches with the union $\{25, 1\}$ in the segmentation. Likewise, the GTR 7 is both split in regions 4, 5, 6, 7, 8, 9, 10 in S , and partially merged with GTR 8, 9.

In other words, segmentation can both split a GTR and merge it partially or totally with other GTR . This cannot be directly seen by scanning the two lists $\mathcal{M}(Rg_p)$ and $\mathcal{M}(Rs_q)$ separately. On the other hand, all operations done 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, which is derived from $\mathcal{M}(Rg_p)$ and $\mathcal{M}(Rs_q)$.

$$\mathcal{M}_k = \{(g_k, \mathcal{S}_k)\} = \{(\bigcup Rg_p, \bigcup Rs_q)\}.$$

Identification and prior diagnosis A GTR is defined as a *non-mergeable, maximal* set of lines. Therefore the most frequently expected case is having *one* GTR matching with n regions in S , $n = 1$ being the “ideal” case.

The operations done on the GTR and some segmentation errors are identified in the following way:

- let $\mathcal{M}_k = \{\bigcup Rg_p, \bigcup Rs_q\}$ be a match of unions of regions,
- let $card(Ug_k)$ (resp. $card(Us_k)$) be the number of regions in $\bigcup Rg_p$ (resp. $\bigcup Rs_q$).
- $card(Ug_k) = 0$ and $card(Us_k) > 0$: the Rs_q have no corresponding $Rg_q \Rightarrow$ *noise improperly detected as a region*,
- $card(Us_k) = 0$ and $card(Ug_k) > 0$: a Rg has no corresponding Rs , \Rightarrow *undetected region*,
- $card(Ug_k) = 1$ and $card(Us_k) > 1$: a Rg has two or more corresponding $Rs \Rightarrow$ *region splitting*,
- $card(Us_k) = 1$ and $card(Ug_k) > 1$: a Rs has two or more corresponding $Rg \Rightarrow$ *region merging*.
- $card(Us_k) > 1$ and $card(Ug_k) > 1 \rightarrow$ region splitting *and* merging.

This way, a prior diagnosis is derived on the GTR . Figure 2 shows the matches of unions of regions obtained on example [page2](#).

4.2 Detection of segmentation errors

The prior diagnosis allows immediate rejection of regions in S corresponding to noise, or regions in $SGTF(I)$ having been undetected or only partially detected.

The evaluation strategy consists, in fact, in progressively rejecting regions in both files, and making additional tests on the remaining non rejected regions. For example, in an “ideal” case where *one* GTR matches with *one* region in S , we have to verify that the overlap between the two regions is total.

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, and if it is horizontal (direction D^\perp), vertical (direction D), or neither. The direction is determined by checking the spatial relationships between the regions of each union.

```

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 overlapping unions of regions, identification of operation made and prior diagnosis in page2

A union of regions can be of three kinds: a vertical alignment, a horizontal alignment, a “not aligned” set, i.e. both vertically and horizontally aligned regions, or empty.

In addition, taking into account the *page cuts* and *region cuts* contained in the ground truth file, we can associate a penalty with the detected merging or splitting.

In example *page2*, region 25 is formed by merging halftones 0 and 2, which is “acceptable”; but the merging also takes place over part of text region 3, which as a result will be improperly tagged as halftone. Moreover, region 1 of *S(I)* matches with the remaining part of region 3 of *GTF(I)*, and therefore is not valid. Thus, for this match, only region 0 of *GTF(I)* is considered to have been segmented in an “acceptable” way, but none of the matching regions in *S(I)* is acceptable.

Test over the ground truth regions Table 1 summarizes the simplified diagnosis of segmentation errors on *GTR*: For each of these operations, in either the vertical or horizontal 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, using again the example of 3cc8001, the *GTR* 7 has been vertically split into 7 regions by segmentation. Yet, two of the latter (4 and 5), are horizontally merged with parts of *GTR* 8 and 9. Therefore, *GTR* 7 (as well as *GTR* 8 and 9) cannot be considered as having been correctly segmented, because of this “bad horizontal merging”. On the other hand, on that same example, *GTR* 0, which is a halftone, has been horizontally merged with *GTR* 2, which is also a halftone. This type of “horizontal merging” does not affect OCR, so *GTR* 0 is tagged as having been “acceptably” segmented.

Following the rules illustrated in Table 1, a detailed output is compiled for each region in the *GTF*.

Operation	Decision	Horizontal	Vertical
Splitting .	Allowed:	NO	YES
	Unless: .	halftones only	- lines split - horizontal merges
Merging .	Allowed:	NO	NO
	Unless:	halftones only	- through page cuts

Table 1 : Error diagnosis rule, on operations done on ground truth regions by segmentation.

Additional tests on some of the segmentation regions As illustrated in example *3cc8001*, tagging a *GTR* as badly segmented (like regions 7, 8, or 9), 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, rather than rejecting the corresponding segmentation regions, we need to examine them one after the other and tag them as good, acceptable, or bad.

5 Results

The algorithm provides a multi-level output for a page segmentation. An image of 1275×1650 pixels, takes around 40 seconds total time on a SPARC-10. Figure 4 and 5 shows the benchmarking results obtained for **page1** and **page2**.

Quantitative evaluation over the test suite A numerical score is provided over the suite of documents on which segmentation has been tested. For the attached example, this score is 5.78/10. It simply represents the average score obtained on each page of the suite.

Qualitative and quantitative page-level evaluation Page-level output is provided both from the ground truth and the segmentation point of view. For each of the two files, regions are classified as “good”, “acceptable” and “bad”, according to the diagnosis of the operations done on the *GTR*. Errors are classified in “horizontal”, “vertical” and “junk extraction” for the group of “bad” regions. For each of the three classes and their sub-classes, the label and type of region are provided. In fact, “missed extraction” represents the number of *pixels* in the *GTF*, that have been missed by segmentation, rather than the number of missed *regions*.

The numerical score provided for the page corresponds to the the average normalized ratio of all existing regions to the “good” and “acceptable” regions in each file. For instance, the score is 3.63/10 for page **page1**, and 4.96/10 for page **page2**. Such a rating is extremely primitive! It does not take into account region size and region type and does not distinguish between good and acceptable regions. A reliable way to numerically evaluate the quality of a segmentation is being designed.

Region-level error specification Output is detailed only for *GTR* which have been incorrectly segmented. For each of these, the benchmarking algorithm provides:

- the label and type,
- the number of pixels of the *GTR* missed by segmentation (“NbNonSegmPix”),
- the number of regions it has been split in (“NbSegments”),
- the diagnosis on the splittings done on that region,
- the diagnosis on the mergings done on that region.

Yet, like for the *GTR* 1 in **page1**, the errors in a region are not always detailed, since for example, the fact that this region is a table and had horizontal mergings in *S(I)* is enough to tag it as improperly segmented and invalidate all the corresponding regions in the segmentation.

6 Conclusion

A reliable benchmarking system can be helpful for anyone wishing to evaluate standalone page decomposition systems, as well as the quality of the segmentation provided by commercial document recognition systems. An automatic

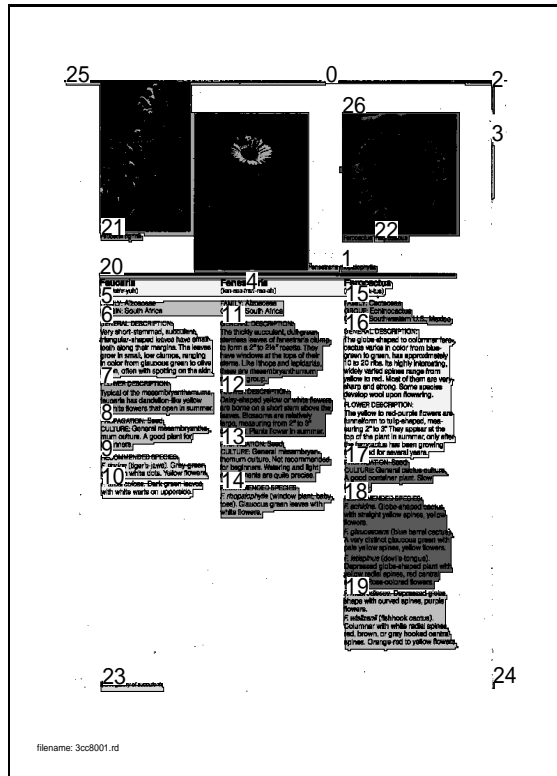
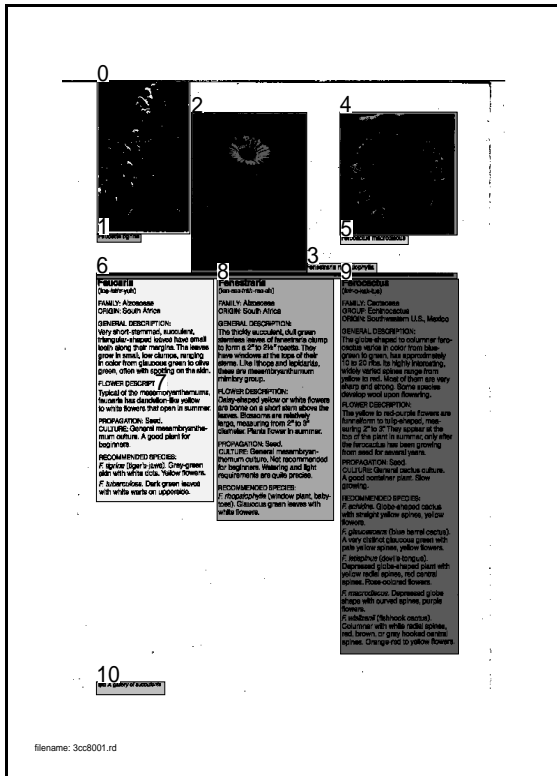
system, processing large test suites is also an invaluable development tool. The need of such a tool is now more and more stressed.

We have introduced the first automatic benchmarking scheme providing a comprehensive and explicit specification of page segmentation errors. This is the first attempt to benchmark segmentation, by using segmentation results only. Furthermore, it is independent of any zone representation scheme. A quantitative evaluation is derived from the error identification and a more reliable measure is being developed. We regard the present system as a prototype of a benchmarking system to evaluate a broad range of segmentation techniques.

A segmentation ground truth file format and creation strategy have also been proposed. To enable a broader use of the presented tool by the document recognition community would require an agreement upon a standard segmentation ground truth representation scheme together with a normalization of the segmentation output produced by devices.

7 References

1. ARPA. Dimund workshop on page decomposition, character recognition and data standards. Harpers Ferry WV, August 9–11 1993.
2. H. Baird. *Structured Document Image Analysis*, chapter Document Image Defect Models, pages 546–556. Springer Verlag, 1992.
3. H. Baird, S. Jones, and S. Fortune. Image segmentation by shape-directed covers. In *10th International Conference on Pattern Recognition (ICPR'90)*, pages 820–825, Atlantic City, NJ, June 1990.
4. D. Davies. Xerox (aods) unpublished work. 1992.
5. S. Huids, J. Fisher, and D. d'Amato. A document skew detection method using runlength encoding and the hough transform. In *Proc. of 10th International Conference on Pattern Recognition (ICPR'90)*, pages 464–468, 1990.
6. J. Kanai, T. Nartker, S. 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.
7. J. Kanai, S. Rice, and T. Nartker. A preliminary evaluation of automatic zoning. Annual research report, ISRI, University of Nevada, 4505 Maryland Parkway, BOX 454021, Las Vegas 89154-4021, 1993.
8. M. Nadler. A survey of document segmentation and coding techniques. *Computer Vision Graphics and Image Processing*, (28):240–262, 1984.
9. T. Pavlidis and J. Zhou. Page segmentation and classification. *Computer Vision Graphics and Image Processing*, 54(6):484–486, November 1992.
10. I. Phillips, S. Chen, J. Ha, and R. 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.
11. I. Phillips, S. Chen, and R. Haralick. Database zone label definition and examples. Technical report, University of Washington, Intelligent Systems Lab., Seattle WA 98195, 1993.
12. S. Randriamasy and L. Vincent. Benchmarking page segmentation algorithms. In *to appear in: IEEE conference on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle WA 98195, June 20-23 1994.
13. S. Rice, J. Kanai, and T. Nartker. An evaluation of ocr accuracy. Technical report, ISRI, University of Las Vegas, 1993.
14. S. Shrihari and J. Hull. *Encyclopedia of Artificial Intelligence*, chapter Character Recognition, pages 138–150. Wiley Interscience, NY, 1992.
15. M. Vision and Applications. Document image analysis techniques. *Special Issue*, 5(3), 1992.



0 The following table summarizes stock option activity for 1984, 1985, and 1986:

	Shares	Option Price per Share	Shares	Option Price per Share
Common Stock				
Outstanding at December 31, 1983	1,608,616	\$0.55-\$81.17	80,750	\$7.25-\$7.75
Granted	1,765,870	\$0.55-\$85.50	-	-
Exercised	(241,886)	\$0.55-\$28.50	(11,400)	\$7.25-\$7.75
Cancelled	(129,253)	\$0.55-\$81.17	(14,600)	\$7.25
Outstanding at December 29, 1984	3,002,567	\$0.55-\$88.17	82,750	\$7.25-\$7.75
Granted	3,946,736	\$0.55-\$77.86	-	-
Exercised	(275,769)	\$0.55-\$85.45	(900)	\$7.75
Cancelled	(3,137,606)	\$2.00-\$98.17	-	-
Conversion of restricted common to common stock options	33,673	\$4.85 & 5.17	(2,456)	\$7.25-\$7.75
Outstanding at December 28, 1985	3,644,797	\$0.55-\$88.17	-	-
Granted	1,284,086	\$0.55-\$12.66	-	-
Exercised	(208,545)	\$0.55-\$28.00	-	-
Cancelled	(897,883)	\$0.55-\$88.17	-	-
Outstanding at January 9, 1987	4,959,082	\$0.55-\$98.17	-	-

1 In October 1986, the Company offered such options who held outstanding unexercised options granted under the 1985 Incentive Stock Option Plan or the 1984 Non-Qualified Common Stock Option Plan ("Old Options") the opportunity to exchange such options on a one-for-one basis for new options to be granted under the 1984 Non-Qualified Common Stock Option Plan ("New Options"). The New Options have an exercise price of \$6.00 per share. The fair market value of the Company's common stock on the date of the exchange was \$9.75. Options who elected to make the exchange lost the accumulated vesting on the cancelled Old Options, and a new vesting schedule began on the New Options at the Company's standard rate of 5% per quarter from the date of grant. The effect of the exchange program described above was to increase unexercised compensation by approximately \$6,000,000 in 1986.

2 Under the 1982 Employee Stock Purchase Plan, the Company is authorized to grant options to purchase an aggregate of 780,000 shares of common stock in a series of six-month offerings through November 1987. All employees are eligible to receive options under the plan. The purchase price is 80% of the lower of the fair market value of the stock on the first or last day of the offering. To date, employees have purchased 490,068 shares in seven price offerings at prices ranging from \$2.89 to \$17.67. The eighth offering commenced on September 29, 1986. Options have been granted to 643 cancelled employees to purchase an aggregate of 82,389 shares (subject to an adjustment in the event of a decrease in fair market value as of April 4, 1987) at a price not to exceed \$18.92 per share.

0 The following table summarizes stock option activity for 1984, 1985, and 1986:

	Shares	Option Price per Share	Shares	Option Price per Share
Common Stock				
Outstanding at December 31, 1983	1,608,616	\$0.55-\$81.17	80,750	\$7.25-\$7.75
Granted	1,765,870	\$0.55-\$85.50	-	-
Exercised	(241,886)	\$0.55-\$28.50	(11,400)	\$7.25-\$7.75
Cancelled	(129,253)	\$0.55-\$81.17	(14,600)	\$7.25
Outstanding at December 29, 1984	3,002,567	\$0.55-\$88.17	82,750	\$7.25-\$7.75
Granted	3,946,736	\$0.55-\$77.86	-	-
Exercised	(275,769)	\$0.55-\$85.45	(900)	\$7.75
Cancelled	(3,137,606)	\$2.00-\$98.17	-	-
Conversion of restricted common to common stock options	33,673	\$4.85 & 5.17	(2,456)	\$7.25-\$7.75
Outstanding at December 28, 1985	3,644,797	\$0.55-\$88.17	-	-
Granted	1,284,086	\$0.55-\$12.66	-	-
Exercised	(208,545)	\$0.55-\$28.00	-	-
Cancelled	(897,883)	\$0.55-\$88.17	-	-
Outstanding at January 9, 1987	4,959,082	\$0.55-\$98.17	-	-

1 In October 1986, the Company offered such options who held outstanding unexercised options granted under the 1985 Incentive Stock Option Plan or the 1984 Non-Qualified Common Stock Option Plan ("Old Options") the opportunity to exchange such options on a one-for-one basis for new options to be granted under the 1984 Non-Qualified Common Stock Option Plan ("New Options"). The New Options have an exercise price of \$6.00 per share. The fair market value of the Company's common stock on the date of the exchange was \$9.75. Options who elected to make the exchange lost the accumulated vesting on the cancelled Old Options, and a new vesting schedule began on the New Options at the Company's standard rate of 5% per quarter from the date of grant. The effect of the exchange program described above was to increase unexercised compensation by approximately \$6,000,000 in 1986.

2 Under the 1982 Employee Stock Purchase Plan, the Company is authorized to grant options to purchase an aggregate of 780,000 shares of common stock in a series of six-month offerings through November 1987. All employees are eligible to receive options under the plan. The purchase price is 80% of the lower of the fair market value of the stock on the first or last day of the offering. To date, employees have purchased 490,068 shares in seven price offerings at prices ranging from \$2.89 to \$17.67. The eighth offering commenced on September 29, 1986. Options have been granted to 643 cancelled employees to purchase an aggregate of 82,389 shares (subject to an adjustment in the event of a decrease in fair market value as of April 4, 1987) at a price not to exceed \$18.92 per share.

Figure 3: Ground truth (left) and obtained (right) segmentations on documents "page1" (top pair) and "page2" (bottom pair).

```

*****
SEGMENTATION BENCHMARKING FOR FILE : table18
*****
=====
PAGE OUTPUT (file: table18) = 3.63 / 10
=====

Nb Regions [SEGM] = 50
Nb Good Regions [SEGM] = 2
Nb Acceptable Regions [SEGM] = 1
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:
=====
Region = 47 Type = TEXT
Region = 48 Type = TEXT

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

ERRORS IN : SEGM
=====

Horizontal Errors: 40
type TEXT : 40

(not enumerated)
Vertical Errors: 6
type TEXT : 6
(not enumerated)

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

```

Figure 4 : Output of the benchmarking for page1

```

*****
SEGMENTATION BENCHMARKING FOR FILE : 3cc8001
*****

=====
PAGE OUTPUT (file: 3cc8001) = 4.963 / 10
=====

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
=====

GOOD REGIONS:
=====
Type = TEXT : 17
(not enumerated)

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 :
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

SEGMENTATION ERRORS
=====
** SPLITTINGS :

Horizontal :
Region = 3 Type = TEXT

Vertical :
Region = 5 Type = TEXT

** MERGINGS :
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
* Mergings : H_ERROR: 1 H_CORRECT: 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 : V_OK: 6
* Mergings : H_ERROR: 1

REGION : 8 Type : TEXT
* NbNonSegmPix : 0
* NbSegments : 6
* Splittings : V_OK: 5
* Mergings : H_ERROR: 2

REGION : 9 Type : TEXT
* NbNonSegmPix : 0
* NbSegments : 6
* Splittings : V_OK: 5
* Mergings : H_ERROR: 1

VERTICAL ERRORS: 2

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

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

```

Figure 5 : Output of the benchmarking for page2