Proc. IEEE Conf. on Computer Vision and Pattern Recognition, Seattle WA, June 1994

Benchmarking Page Segmentation Algorithms

S. Randriamasy*

Harvard Robotics Lab.
Harvard University
Cambridge, MA 02138

L. Vincent

Technology Development Group
Xerox Imaging Systems
Peabody, MA 01960

Abstract

A method for automatically evaluating the quality of document page segmentation algorithms is introduced. Many different zoning techniques are now available, but there exists no robust method to benchmark and evaluate them reliably. Our proposed strategy is a region-based approach, in which segmentation results are compared with manually generated "ground truth files", describing all possible correct segmentations. A segmentation ground truthing scheme was already proposed. The evaluation of segmentation quality is achieved by testing the overlap between the two sets of regions. In fact, the regions are defined as being the valued pixels contained in the extracted polygons. An explicit specification of segmentation errors and a numerical evaluation are derived. The algorithm is simple and fast, and provides a multi-level output for each segmentation.

1 Introduction

The process of document recognition usually involves:

1-Page segmentation: (also called "page decomposition" or "zoning"), where the page is decomposed into its structural and logical units, such as text, halftones, graphics, tables. It is typically followed by a *region-ordering* step, in which the natural reading order of the page is derived. This step is sometimes considered as part of segmentation itself.

2-Recognition: whereby Optical Character Recognition (OCR) is ran on the text regions extracted in step 1.

3-Format/layout analysis: where such things as margins, paragraph endings, tabulations and indentations are determined.

Over the past decades, OCR accuracy has been highly investigated and improved [2, 11, 10], thanks to the availability of reliable benchmarking tools [4, 8]. Evaluating OCR accuracy is rather easy: the solution is usually unique and one can simply compare two flows of characters, corresponding to the actual and pre-stored "ideal" output.

However, a successful page segmentation is a necessary precondition for the ensuing steps: reliable classification and ordering of the extracted regions allow the OCR system to make appropriate treatment for each of them. Information such as size, type and shape is needed for format analysis. Last segmentation decolumnizes multi-column documents, in order for OCR to read the text lines in the correct order.

Page segmentation algorithms are now numerous and improving [6, 3, 7, 12]. Many lately proposed techniques use the white spaces of the page, as the basis for segmentation. Hybrid methods then merge connected components of black pixels into logical units. None of the existing techniques is optimal, and there is now a wide agreement upon the need for an automatic tool to benchmark them [1]. Such a system, allowing to process very large document suites, would be an invaluable development tool and could help one to choose a suitable page decomposition technique.

Yet, as far as we know, there does not presently exist any technique able to automatically evaluate page decomposition *independently* of the rest of the system and provide an accurate specification of the detected errors. The only method known to work reliably is visual inspection of the regions extracted by the zoning system, making any global analysis impossible.

Unlike for OCR, benchmarking page segmentation is a complex problem: the optimal segmentation for a given page is not necessarily unique; errors are not uniform accross the page: the effect of splitting or merging regions depends on the orientation of the text lines; two sets of regions similar in shape are not necessarily similar in quality. Last, region representation schemes are not standardized, so regions cannot be compared geometrically.

We propose a bitmap-level, set and region-based approach, where results are compared with predefined "ground truth files" describing all the possible correct segmentations.

At this low-level stage, the goal of a good segmentation is to provide correct input for region ordering and layout analysis. Currently, ground truth representation schemes are not standardized. A simple, general, and expandable format, where a ground thruth file *GTF(I)* associated with an original image I, contains a set of non-mergeable ground truth regions (GTR) and their associated allowed partitions has been proposed in [9].

Rather than the whole zones in *GTF(I)* and the obtained segmentation *S(I)*, the regions are equivalent to the *set of valued pixels* they contain. This way, they may have any shape, as long as their content in I is correct. The algorithm compares $S(I)$ and *GTF(I)* by testing the overlap between the two sets of regions. Overlapping *unions* of regions are compared, as well as their cardinal[1] and spatial structure. This allows for a straightforward detection of certain types of errors. Incorrectly segmented regions are then progressively rejected while the detected errors are accurately identified. Error detection mostly relies on bounding box overlapping. Comparison of their *contents* is done on the non-rejected regions, before validating them. Therefore, the

[1] which is the number of regions in each of the two unions

process is fast and simple.

The benchmarking algorithm provides a multi-level output for each segmentation, from the page level to the region level. Quantitative evaluation of a segmentation is derived from the cost of incorrect splittings and mergings done, as well as the ratio of the correct regions to the bad ones in each of the two images. The method has been tested on about a dozen different mixed image/text machine-written document images, choosen to span a wide spectrum of typical errors and has shown very promising results.

1.1 Previous Work

High level approach: text-based benchmarking The only currently known automated way to evaluate zoning performance was proposed by Kanai et al.[5] It computes an "edit distance", which is the minimal number of text insertions, deletions, and block moves, needed to transform the OCR output into the OCR ground truth. The derived cost includes OCR errors, so the cost of segmentation *itself*[2] is derived by comparing the costs corresponding to manually and automatically zoned pages.

Being purely text-based, this method is independent of any zone representation scheme (ZRS) and does not require any prior segmentation ground truthing. However, segmentation which is mostly a bitmap-level process, is evaluated by using the output produced at a higher level, and there is in general no way to know whether the OCR errors are due to region ordering or segmentation. Moreover, the system provides a numerical score from which it is impossible to derive any understanding of what kind of segmentation errors actually occured.

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

To avoid these drawbacks, one could rather use strings of pixels and compare the order in which black pixels are scanned for $S(I)$ and *GTF(I)*. A cost of correcting operations, associating insertions to pixels missed by the segmentation, deletions to regions irrelevantly extracted as text and block moves to segmentation or region ordering errors would be derived.

Such a method could benchmark segmentation with low-level output only. Like in § 1.1, it would be independent of any ZRS and segmentation ground truthing. However, the derived score would still be linked to region ordering. Similarly as in § 1.1, the evaluation is still numerical and provides no explicit error specification. In addition, the strings of pixels to be matched here would be extremely long, making their comparison cumbersome.

1.3 Proposed region and bitmap based approach

We have opted for a region-based method, capable of evaluating segmentation *with* or *without* region ordering, and by using segmentation results only. The algorithm compares lists of *regions* in the automatically and manually zoned images. Regions are a much richer, easier to handle, and more synthetic way to describe segmentations. Using them, segmentation errors are easier to find and to characterize.

In the present system, *regions* are equivalent to the *set of "black" pixels* contained in the polygons derived by segmentation, rather than to the polygons themselves. This way, they may have any shape as long as their content in I is correct. Therefore, although region-based, our approach is totally ZRS-independent.

Similarly to OCR error identification, evaluating a segmentation first consists in determining the operations needed to correct it. The operations considered here are **splitting** and **merging,** in either the **vertical** or the **horizontal** direction.

The benchmarking system provides an explicit specification of the number and type of incorrect operations done by segmentation, as well as the number, label and type of regions affected. From all this, a global rating of segmentation is derived. The non-unicity of the optimal segmentation is considered. Our method proceeds in two main steps:

1- Creation of ground-truth file: *GTF(I)* for each document I on which segmentation is to be benchmarked. It contains, explicitely or implicitely, a synthetic description of every possible correct segmentation of I.

2- Matching of segmentation result against ground-truth file: by testing the overlap between the two sets of regions.

Assumptions To determine whether a merging or splitting is horizontal or vertical, we assume prior deskewing of I. We suppose here that the pages are read from top-left to bottom-right, like European documents. It would be easy yet, to adapt this method to Middle Eastern or to some Asian documents, where text lines may be both vertical and horizontal. The currently considered types of regions are: *Text* (text columns, title and drop caps), *Tables, Line-Art, Halftone and Rulings.*

Examples Due to lack of space, we show only one example of a test document called page2, a multi-column document with text, halftones and noise, where the segmentation exhibits both vertical and horizontal splitting and mergings of regions. Note that the displayed *GTF* is actually *one of the* multiple correct segmentations, namely the one with the biggest possible regions.

2 Automatic benchmarking scheme

2.1 Input and region representation

For each document image $\mathcal{D}$, the required inputs are: the binary image I of $\mathcal{D}$, *GTF*(I) and $S(I)$.

In order to speed up the process and without any loss of accuracy, the resolution of I is first reduced by two. Prior to benchmarking, the black pixels of I are labelled into sets belonging to the same zone. Basically, $S(I)$ and *GTF*(I) are two sets of regions to be compared. They are represented by a list of regions.

The main existing ZRS are: bounding rectangles, nested bounding rectangles, polygons and piecewise rectangles. We have opted for polygons, with either vertical or horizontal edges. This is more realistic than a rectangle and encompasses most of the ZRS. It is equivalent to *merged* piecewise rectangles, each one corresponding to a text-line. Actually, each region is described as a list of polygons, usually one. It may therefore have any shape, have holes, or be disconnected. The algorithm also processes overlapping regions, whose content is then disambiguated.

2.2 Creating Ground Truth Files

The utilized segmentation ground truthing scheme is detailed in [9], together with a set of guidelines to build a GTF and a simple and expandable representation format. A GTF contains a minimal number of non-mergeable regions. The ones not processed by recognition are first isolated. Maximal text-GTR are

[2] called "automatic zoning metric"

then formed by merging text-lines in the *reference direction* D, orthogonal to the current line orientation $D^{\perp}$.

The non-unicity of the correct segmentation lies in accepting vertical partitions of GTR, by specifying a list of authorized *Region Cuts* $(y_C, x_{Cbeg}, x_{Cend})$ in spaces wider than one line; cuts between lines are only implicitely tolerated. Mergings between regions such as titles and paragraphs, or a page number with the last read paragraph, are tolerated and specified by *Page Cuts*. Until the definition of *Cuts*, the representation format is identical for $S(I)$ and *GTF(I)*. Fig. 1 shows the GTF generated for `page2`.

```
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`.

3 Error detection and identification

To qualitatively compare $S(I)$ and *GTF(I)*:

- the operations done on the GTR are first identified by testing the overlap between the two sets of regions.

- the correctness of these operations is then tested according to predefined rules and to the *cuts* defined in the GTF.

With each operation, a cost or penalty is associated:

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

- **acceptable:** small 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.

3.1 Comparing the two sets of regions

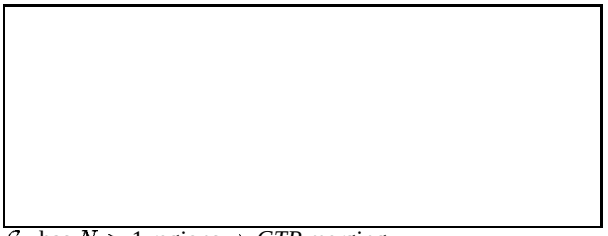
By a simple overlap matching technique, we associate each region of one of the two sets, to the region(s) in the other set it has a non-empty intersection in I with. A list $\mathcal{M}$, containing pairs of unions of matching regions is obtained. $\mathcal{M} = \{(\mathcal{G}_k, \mathcal{S}_k)\} = \{\bigcup R_g, \bigcup R_s\}$. Comparing the cardinal of each union of each pair, allows to immediately detect:

- *noise mistaken for a region*: a R_s has no corresponding R_g,
- *undetected region*: a R_g has no corresponding R_s,
- *region splitting*: a R_g has several corresponding R_s,
- *region merging*: a R_s has several corresponding R_g.
- *region splitting and merging*: several R_s have several corresponding R_g.

Spatial structure of a union The spatial relationships between the regions of each union U determine the direction in which the operation was done on the GTR. We consider four types of alignement for U: horizontal (Fig. 2), vertical (Fig. 4) , mixed, called "T-alignment", none (0 or 1 region). The possible adjacency relation between two regions is 4-connected : top, down, right, bottom. It is determined by using their bounding boxes, caracterized by the quadruplet: $(y_{top},\ y_{bottom},\ x_{left},\ x_{right})$.

The horizontal and vertical bounding box overlap are computed, and the direction of the adjacency corresponds to the smallest of these two quantities. 8-connected adjacency relations are equivalent to none. If both overlaps are equal, the alignment is set by default to the direction D (vertical), in order to avoid irreversible erroneous rejections of regions.

Fig. 2 shows a matching pair of unions of horizontally split and merged regions. Such a case occurs in `page2`: the union $\{0, 2, 3\}$ in *GTF(I)* matches with the union $\{25, 1\}$ in *S(I)*. Fig. 3, shows the



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

Figure 2: Overlap matching of 2 unions of horizontally aligned regions, and identification of operations done.

matches of unions extracted, together with the number of regions and spatial structure of each union, and the prior diagnosis derived.

```
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 3: Matches of unions of regions extracted in `page2`.

3.2 Error detection

To ouput a fair and reliable estimate of zoning quality, we consider both perspectives of *GTF(I)* and *S(I)*. For example, on `page2`, (see § 3.4), only four GTR are tagged as correctly segmented, as opposed to seventeen regions in *S(I)*. Therefrore:

- The *GTR* are first tested in order to identify possible errors. The corresponding regions in *S(I)* are subject to rejection.

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

3.2.1 Test over the ground truth regions

The incorrectly segmented regions are progressively rejected by using their bounding boxes. Their contents in I is only checked for the non-rejected ones. For instance on `page2`, the *GTR* 7 has been vertically split into 7 regions. Yet, two of them (4 and 5), are horizontally merged with parts of *GTR* 8 and 9, perturbing the reading order of the page. Therefore, *GTR* 7, 8 and 9 are tagged as "bad". Likewise, *GTR* 0 has been horizontally merged with *GTR* 2. Both of them are halftones, therefore not processed by OCR. But *GTR* 2 is also merged with a part of text-region 3, which is mistaken for halftone. Moreover, region 1 of $S(I)$ matches with the remaining part of *GTR* 3, and therefore is not valid. Thus, for this match, only region 0 of $GTF(I)$ 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 segmented regions indicates a horizontal splitting of *GTR*. The regions are scanned two by two and tagged as "bad", if any text is involved.

Exception: if the left region is a dropcap, then both regions are tagged as "acceptable".

Horizontal merging: A horizontal alignment $\bigcup R_g$ of GTR indicates a horizontal merging of these. The regions of $\bigcup R_g$ are scanned two by two, and rejected. Any common matching region in *S(I)* is tagged as "bad".

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 the *GTF*.

Errors in the direction D (vertical)

Vertical splitting: A vertical alignment in *S(I)* indicates a vertical splitting of a *GTR* G and is a-priori correct. It usually corresponds to cutting a text column into paragraphs.

To be tagged as good, two vertically adjacent regions A and B, with for example A being upon B in *S(I)*, must respect the following rules, (see Fig. 4):

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

- The space between $y_{top}(bB)$ and $y_{bottom}(bA)$ must contain a neglectible number of valued pixels.

- A specified region cut should lie between $y_{bottom}(bA)$ and $y_{top}(bB)$. If not, A and B are tagged as "acceptable" only.

If one of the three first rules is not respected, A, B and G are tagged as "bad", because of a vertical splitting error.

Vertical merging: A vertical alignement of *GTR* indicates a vertical merging of these. The GTR are defined as maximal sets, so vertical mergings are only authorized if through an explicitly defined *Page Cut*. The involved regions are then tagged as "acceptable". Any other vertical merging yields a "bad" tag.

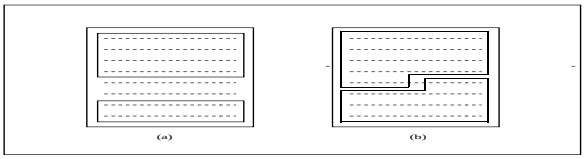


Figure 4: Errors in a vertical alignment: (a) a significant number of valued pixels between 2 regions, (b) a bounding box overlap.

Errors in directions D and $D^{\perp}$ Unions of segmented regions in both directions indicate splitting in both D and $D^{\perp}$. It often occurs in non homogeneous zones, like graphics, halftones and tables, but also in `page2`. Handling such errors depends of the type of region. Tables, (except cell tables), should be read line by line, and contained in one single region. Therefore, any horizontal splitting in a table yields a rejection of all corresponding regions in *S(I)*. For standard text, additional tests on $S(I)$ are necessary.

3.2.2 Additional tests on $S(I)$

As illustrated in `page2`, tagging a *GTR* as badly segmented should not lead to a systematic rejection of *all* the corresponding regions. 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 being rejected, the corresponding segmentation regions, are examined one after the other and re-tagged. A typical case is: if a union U_G of horizontally aligned GTR is found and if each region of U_G is vertically split, then the algorithm rejects the regions in $S(I)$ matching with several regions in U_G, but accepts the others if their horizontal cuts are correct.

3.3 Quantitative 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 also enable to compare different segmentations. The evaluation is provided by two quantities:

- *PageScore*: 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 $S(I)$.

Like for qualitative evaluation the rate is computed both in the perspective of $GTF(I)$ and $S(I)$.

Currently, for each of the two sets of regions, we 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 labelled pixels.

Similarly, on the ground truth regions, we calculate:

- the number O_{err} of "bad" operations done,

- the number O_{acc} of "acceptable" operations done.

The *PageScore* over the page is then equal to:

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

The *PageCost* over the page 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 $w_{acc} = 0.5$.

3.4 Output of the benchmarking

A multi-level output is produced for each page segmentation. An image of 1275 $\times$ 1650 pixels, takes around 40 seconds total time on a SPARC-10. Results for `page2` are provided here.

Overall output of the test suite A numerical score is provided. It simply represents the average *PageScore* obtained on the suite. Likewise, the overall cost is the average *PageCost* over the suite.

Page-level output is provided both from the perspective of *S(I)* and *GTF(I)*. For the two files, regions are classified as "good", "acceptable" and "bad". Errors are classified in "horizontal", "vertical" and "junk extraction" for the "bad" regions.

Region-level output is only detailed for *GTR* having been incorrectly or acceptably segmented. The benchmarking algorithm then provides: the label and type, the number of pixels of the *GTR* missed by segmentation ("NbNonSegmPix") if significant, the number of regions it has been split in ("NbSegments") and the type and number of errors done. Yet, the errors in a region are not always detailed. For example, if a GTR is a table and has been horizontally split, it is enough to tag it as improperly segmented and invalidate all the corresponding regions in *S(I)*.

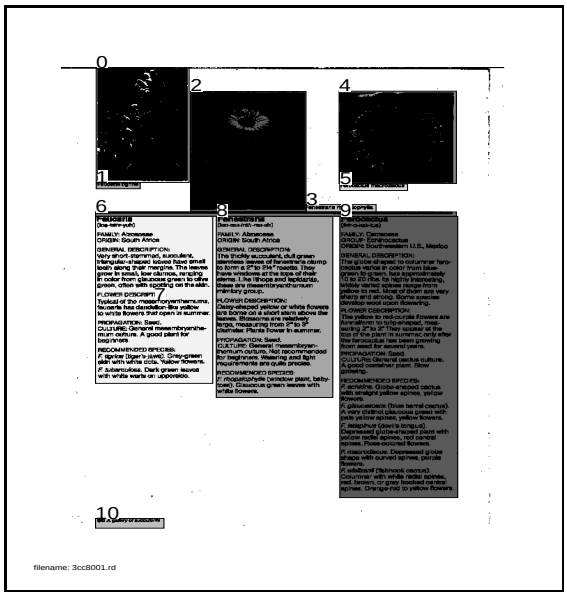


Figure 5: "Maximal" ground truth segmentation on `page2`.

```
++++++++++++++++++++++++++++++++++++++++++++
 SEGMENTATION BENCHMARKING of PAGE : page2
++++++++++++++++++++++++++++++++++++++++++++
  ****  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
```

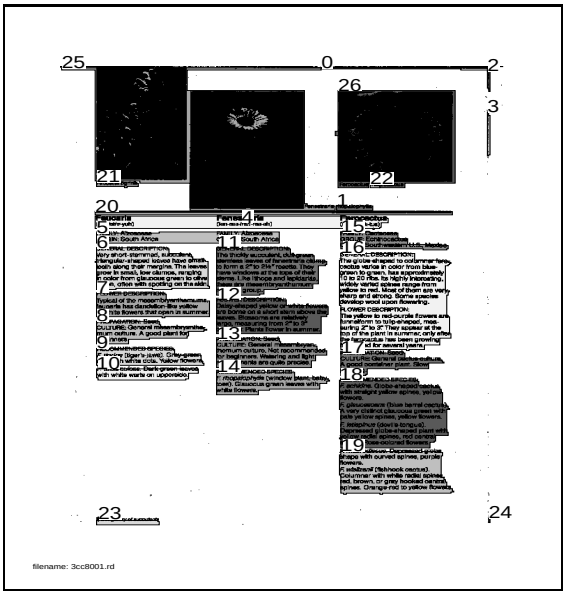


Figure 6: Segmentation obtained on `page2`.

```
 Nb Acceptable Regions [GR_TRUTH] =   1
 Nb Bad Regions [GR_TRUTH] =   7
========================================
   SEGMENTATION
========================================
GOOD REGIONS: 17
============
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 :    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
* Mergings   :   H_ACCEPT:   1

SEGMENTATION ERRORS
===================
** SPLITTINGS :  2
```

```
HORIZONTAL :
 Region =     3   Type = TEXT
VERTICAL :
 Region =     5   Type = TEXT

** MERGINGS :    7
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
* NbSegments   :   1
* Mergings   :   H_ACCEPT:   1   H_CORRECT:   1
REGION :     3   Type : TEXT
* NbSegments   :   2
* Splittings :   H_ERROR:   1
* Mergings   :   H_ERROR:   1
REGION :     7   Type : TEXT
* NbSegments   :   7
* Mergings   :   H_ERROR:   1
REGION :     8   Type : TEXT
* NbSegments   :   6
* Mergings   :   H_ERROR:   2
REGION :     9   Type : TEXT
* NbSegments   :   6
* Mergings   :   H_ERROR:   1

VERTICAL ERRORS:    2
REGION :     4   Type : HALFTONE
* NbSegments   :   1
* Mergings   :   V_ERROR:   1
REGION :     5   Type : TEXT
* NbSegments   :   2
* Splittings :   V_ERROR:   1
* Mergings   :   V_ERROR:   1

MISSED PIXELS : NONE
```

4 Conclusion

We have proposed a bitmap-level automatic scheme to benchmark page segmentation algorithms on mixed text/halftone documents. It provides an accurate qualitative diagnosis of segmentation techniques, from which, a quantitative evaluation is derived.

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 small incorrect sub-regions; errors often occur in zones of I where the density of the valued pixels changes between two very close adjacent regions, see regions 2 and 3, or 4 and 5 in Fig. 5. A more reliable numerical evaluation of the segmentation quality will be designed. It is essential to anyone wishing to fine tune segmentation parameters or test available techniques on various kinds of input documents.

A ground truthing scheme was also proposed in [9]. An agreement upon a standard ground truth representation scheme in order to enable a broad use of our system by the document recognition community, is necessary. 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.

Acknowledgements The first author was a post-doctoral fellow supported by the French Ministry of Research and Technology.

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 International Conference on Pattern Recognition (ICPR'90)*, pages 820–825, Atlantic City, NJ, June 1990.

[4] 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.

[5] 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.

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

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

[8] 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.

[9] 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, California, USA, February 6-10 1994.

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

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

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