Proc. ICDAR'95, Third International Conference on Document Analysis and Recognition, Montreal, Canada, pp. 601–604, August 1995

Ground-truthing and Benchmarking Document Page Segmentation

Berrin A. Yanikoglu & Luc Vincent

Xerox Imaging Systems
berrin@xis.xerox.com lucv@xis.xerox.com

Abstract

We describe a new approach for evaluating page segmentation algorithms. Unlike techniques that rely on OCR output, our method is region-based: the segmentation output, described as a set of regions together with their types, output order etc., is matched against the pre-stored set of ground-truth regions. Misclassifications, splitting, and merging of regions are among the errors that are detected by the system. Each error is weighted individually for a particular application and a global estimate of segmentation quality is derived. The system can be customized to benchmark specific aspects of segmentation (e.g., headline detection) and according to the type of error correction that might follow (e.g., re-typing).

Segmentation ground-truth files are quickly and easily generated and edited using GroundsKeeper*, an X-Window based tool that allows one to view a document, manually draw regions (arbitrary polygons) on it, and specify information about each region (e.g., type, parent).*

1 Introduction

Page segmentation (also called *page decomposition* or *zoning*) is the process of decomposing a document page into its structural and logical units (called *regions* or *zones*), such as headlines, graphics etc. In typical modern document recognition systems, this step is performed first and provides a coarse-level document understanding. Segmentation is an essential part of the whole recognition process. For example, the discrimination between text and graphics allows the OCR module not to loose time trying to recognize text in halftones. Also, successful separation of a column of text from an adjacent one in multi-column pages (magazine articles, newspapers, etc.) is essential, since otherwise the OCR module would not be able to differentiate between these galleys and would produce a useless output (for many applications) —even though the recognition quality may otherwise be excellent.

Once the regions are found (*zoning*) and their types identified (*labeling*) they need to be ordered according to the natural reading order(s) of the page. This step, called *region ordering*, is sometimes considered part of page segmentation itself.

Until recently, OCR accuracy was the only aspect of document recognition systems that was commonly benchmarked. Nowadays however, sophisticated document recognition systems are able to handle increasingly complex documents, like newspapers, magazines, junk mail, etc. Due to the complex layout of such documents, these systems have to rely more heavily on automatic page segmentation. It becomes, therefore, important to be able to benchmark page segmentation.

In the next section, we briefly review literature on benchmarking page segmentation. Section 3 introduces ground-truthing issues and *GroundsKeeper*, the ground-truthing tool we developed. In Section 4, we describe our segmentation benchmarking system. Section 5 gives a summary and describes the future work.

2 Previous work

One approach to benchmarking page segmentation is to compare the OCR output of the segmented image to the ground-truth OCR output. This is the approach developed at the University of Nevada in Las Vegas (UNLV) [2]. The number of operations (character insertion and deletion, block move) needed to transform the OCR output into the correct text, is used to measure the performance of the segmentation algorithm. Since the cost of this transformation also includes the cost of OCR errors, this latter cost is computed and subtracted from the overall cost.

This approach has several drawbacks. First, the total error includes OCR errors as well as segmentation errors, and the assumption that OCR errors are independent of segmentation errors (hence can be subtracted) does not always hold. Second, some segmentation mistakes may not be detected if OCR errors also occur at those same characters. Hence, high OCR error rates may fool the system to give unreliable estimates of segmentation quality. Finally, only a global measure of segmentation quality is derived, which does not provide much information about the types or location of the segmentation mistakes that were made.

Our approach is based on previous work done by Randriamasy and Vincent [1], which has been improved significantly in several respects and has become a usable system.

3 Ground-truthing page segmentation

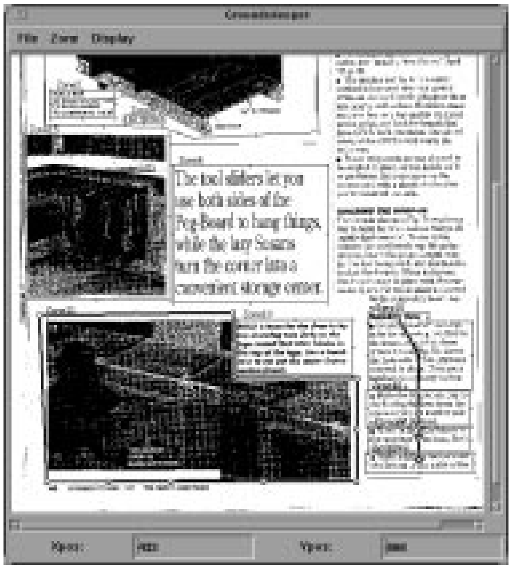
3.1 Ground-truth description

The segmentation ground-truth of a page is not unique, because what constitutes a region is not well-defined (e.g., a text column with two subsections can be zoned into one region or split vertically into two) and there is more than one correct reading order of the page (e.g., an inset or a figure caption can be read anywhere in the reading order).

In order to represent all possible segmentations of a region as correct, we define both the minimal and maximal segmentations of a region in the ground-truth file. Additionally, the user of the benchmarking system can specify whether a vertical split or merge should be penalized or not, in a start-up file. All possible segmentations of the page can hence be derived by implicit or explicit rules. For instance, two paragraphs that are merged vertically are considered as correctly segmented if they are both plain-text regions and one follows the other in the image (vertically aligned) and in the reading order, unless specified otherwise in the start-up file.

More specifically, the page is segmented into its maximal units that are homogeneous both structurally and logically. For instance, two separate columns of regular text are zoned separately and so are headlines from regular text. We zone paragraphs individually, since that information is needed in benchmarking, for certain applications. Tables and images that contain more specific information (cells, text) inside, are zoned as a whole and also the regions inside are zoned separately. This allows us, for instance, to benchmark table detection with or without cell detection.

In order to specify all possible reading orders, we use a partial ordering of the regions, as opposed to a total ordering. The partial order specifies only the necessary order among the regions, and thus handles the ambiguity of reading order. Previously developed benchmarking systems ([2, 1]) expect a particular order.



The tool sliders let you use both sides of the Peg-Board to hang things, while the lazy Susans turn the corner into a convenient storage center.

Figure 1: *A snapshot of GroundsKeeper showing some zones and a partial ordering sequence indicated by arrows on the right hand corner.*

3.2 GroundsKeeper

In order to quickly and conveniently create and update segmentation ground-truth files, we developed an X-window based tool, called *GroundsKeeper*. This tool allows a user to display a document image and draw zones of various types around the different page features. For convenience, several drawing methods can be used, and the drawn zones are fully editable. Note that the shape of a region is not important, since regions are considered equivalent to the set of ON-pixels they contain, as described in Section 4. Each zone can be assigned a type, a parent-region etc., and the region ordering can be specified easily with few mouse clicks. The ground-truth information is saved under the *rdiff* file format, which is a tag-value based ASCII file, with a header and a description for each region, followed by one or more region ordering rules. It is based on the one described in [1].

The capability of editing previously created ground-truth files is very useful, since one can run an automatic segmentation algorithm on several document images, and use those ground-truth files as a start, to further decrease ground-truthing time (about 10 min per page).

GroundsKeeper can be customized in various ways with a resource file that is read at start-up time—for instance to add new region types. A sample snapshot of *GroundsKeeper* is shown in Figure 1.

4 Benchmarking segmentation algorithms

Benchmarking segmentation algorithms is a complex problem. It significantly depends on what the segmentation will be used for and what sort of tools the user has to correct the errors (if there will be any post-correction, such as manually re-zoning some part of the image etc.) For example, if the user of the segmentation output is an OCR system, then the reading order is to be evaluated, in addition to zoning. However, if the segmentation is done for rendering of the image, then only correct classification of each pixel type (e.g., text, halftone) is necessary. Similarly, if the user will manually correct the segmentation output, we may evaluate the segmentation in terms of how easy it is to correct it. On the other hand, if the segmentation output is to be directly fed to the OCR system, this is not a useful metric and we should be concerned with the amount of text area (measured in terms of the number of pixels or characters) that is out of order in the lexical ordering, and the images that are classified as text etc. This represents how good a segmentation is, without taking into account the amount of effort needed to correct it. It resembles the OCR-based benchmarking, while avoiding its shortcomings.

Our benchmarking system is able to evaluate segmentations under various such assumptions, by simply setting few switches in a start-up file. The start-up file includes three sections. The first section indicates the contents of the *rdiff* file (e.g., all possible region types, all the information given about each region etc.). This makes it possible to benchmark, for instance, the detection of region types previously unknown to the system. The next section specifies what needs to be benchmarked (e.g., headline detection, region ordering). Finally, the last section specifies how segmentation mistakes should be penalized. In other words, how the segmentation would be corrected (if at all) and what the weights for each mistake should be etc.

A segmentation algorithm is evaluated by comparing its output on several, previously ground-truthed documents. The segmentation output, described as a set of regions together with their types, output order etc., is matched against the pre-stored set of ground-truth regions. Misclassifications of the region type, splitting and merging of regions, missed pixels, noise regions identified as valid, and wrong region orderings are the main errors that are detected by our system. We are mainly interested in page segmentation in the context of an OCR system, which is why splitting and merging mistakes are important. Each error is weighted individually for a particular application (as specified in the start-up file) and a global estimate of segmentation quality is derived. More details are given in the following sections.

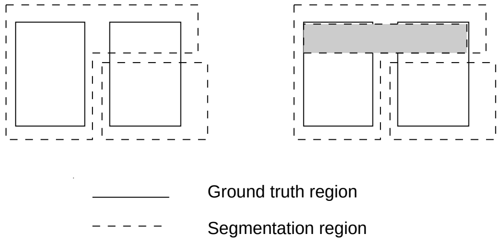


Figure 2: *Only the shaded area is considered badly segmented.*

4.1 Badly segmented areas of the image

In [1], the segmentation errors are measured in terms of the *number* of bad regions (regions with some segmentation mistake) over the total number of regions. Since this percentage is usually different for the ground-truth and the segmentation, both of them are computed as an estimate of the quality of the segmentation, without further interpretation.

This duality can be eliminated by taking a different approach. Instead of the number of badly segmented regions, segmentation quality should be measured in terms of the *size* (in characters or pixels) of the text area that is badly segmented. For instance in Figure 4.1, the shaded area does not follow the reading order and is marked as badly segmented.

This approach of classifying the parts of the image as badly segmented, also gives a better measure of segmentation quality. With the previous approach, horizontally merging two small text columns is considered as bad as merging two large ones, or identifying two large text regions as noise. With our new approach, each of these mistakes can be penalized according to the size of the regions involved.

4.2 Finding the segmentation mistakes

To find the mistakes in a given segmentation, we first find all the segmentation regions that correspond (overlap in ON-pixel content) to a given ground-truth region, and vice versa. To do this, we first form two maps, one for the ground-truth and one for the segmentation, that labels each pixel with all the regions that pixel belongs to. For instance, an image that has some text in it is zoned as a whole, as an image, and the text inside is zoned as text. The text region's pixels in this case will be marked as belonging to both regions. This region map structures allows us to deal with overlapping regions and even very messy segmentations, in a compact and efficient manner. Note that doing the overlap analysis in terms of ON-pixel contents removes any region shape dependency. Two regions will match completely as long as their ON-pixel content is the same, no matter what their shapes are.

After finding the correspondences, we do the error analysis. The main segmentation errors are the splitting and merging of ground-truth regions, classifying valid regions as noise or noise as valid regions, and misclassifying region types. In this process, we use an error map that labels each pixel in the page with the type of error (or the costliest error) it is associated with, such as split or merged. The use of an error map ensures that we do not charge a pixel with more than one error (e.g., when it is both split and merged) and that the benchmarking is accurate.

Each ground-truth and segmentation region is analyzed in the following manner:

- If a ground-truth region does not overlap with any segmentation regions, it is missed, and its pixels are marked as such.
- If a ground-truth region and a segmentation region matches 1-to-1, then we check if all the ground-truth pixels are covered; the ones that are covered are marked as missed. We also check whether the region type is correctly identified.
- If a ground-truth region matches more than one segmentation region, it is split. We mark all the pixels in that region, that are on a split line (a line that is not covered by a single segmentation region) as split.
- If a segmentation region does not overlap with any ground-truth regions, it is noise, and its pixels are marked as such.
- If a segmentation region matches more than one ground-truth regions, it is merging them. We then mark all the pixels in that region, that are on a merging line (a line that is not covered by a single ground-truth region) as merged.
- The ground-truth regions are also analyzed to determine whether they are vertically split or merged (undetected above), by analyzing the alignment of the regions involved.

After the identification of the mistakes, we compute the overall error that occurred in the page. One approach (the default) used to compute the overall error is to sum the weighted (as indicated in the start-up file) cost of each different error type. The normalized cost of a segmentation error is found by finding the percentage of all ON-pixels with that particular segmentation error, over all the ON-pixels on the page.

Shown below is one such (summary) output of the benchmarking system. The segmentation was a bad one where several regions were split and merged. Note that when the split and merge costs are equal, pixels can be marked as split during the detection of split regions, and not changed later to merged, even if they are merged as well.

```
Cost of missed regions = 0.0000
Cost of noise regions = 3.1568
Cost of horizontal merges = 0.8953
Cost of horizontal splits = 64.9288
Overall segmentation quality = 31.0192
```

5 Summary and future work

We proposed a complete and flexible system for automatically evaluating the accuracy of document page segmentation algorithms. Our system compares segmentation results, described as sets of regions, to predefined ground-truth segmentation information. Considering regions as equivalent to the ON-pixels they contain allows us to compare any region shapes, and the use of region and error maps provide accurate and efficient ways to handle the task.

We are currently creating databases of ground-truthed document images of various types. Concurrently, we are refining the benchmarking algorithms and making them able to deal with increasingly complex configurations. We are very close to having a fully operational system that could be used for large-scale experiments.

Beyond document segmentation, we believe that some of the techniques we have developed offer some interesting potential for benchmarking other segmentation problems, as well.

Acknowledgements

We would like to thank Gregory Arzoomanian for his work on the *GroundsKeeper* X-Window based tool.

References

[1] Sabine Randriamasy, Luc Vincent, and Ben Wittner. An automatic benchmarking scheme for page segmentation. In Luc Vincent and Theo Pavlidis, editors, *SPIE/SPSE Vol. 2181, Document Recognition*, San Jose CA, February 1994.

[2] Stephen V. Rice, Junichi Kanai, and Thomas A. Nartker. A preliminary evaluation of automatic zoning. Technical report, ISRI, 1993.