Page Segmentation Using the Description of the Background

Apostolos Antonacopoulos

Department of Computer Science, University of Liverpool, Peach Street, Liverpool, L69 7ZF United Kingdom

Received January 31, 1997; accepted January 17, 1998

There is an ever increasing number of publications which do not have the “traditional” layout where printed regions are rectangular. Text paragraphs and areas of graphic type may be of any shape, individually rotated and in any arrangement. Previous document analysis techniques are not well suited to such complex layouts. This paper introduces a new method for the segmentation of images of document pages having both traditional and complex layouts. The underlining idea is to efficiently produce a flexible description (by means of tiles) of the background space which surrounds the printed regions in the page image under all the above conditions. Using this description of space, the contours of printed regions are identified with significant accuracy. The new approach is fast as there is no need for skew detection and correction, and only few simple operations are performed on the description of the background (not on the pixel-based data).

1. INTRODUCTION

To perform any useful task toward the understanding of the content of a document page, the focus of attention must first be directed toward identifying the areas in the document image that constitute the information.

Page segmentation is the process of identifying the areas of interest in the image of a document page. For a conventional document page with material printed in dark ink on a light-colored paper, the areas of interest in the (binary) image will be neighborhoods of black pixels. Page segmentation produces a description of the geometrical aspects of the areas of interest. The most common aspects are spatial extent and position on the page. Page segmentation can be thought of as a mapping from the pixel-based image data to a description of the areas of interest.

It should be noted that the problem of page segmentation is relevant only to images of document pages whose main information components are textual. Although illustrations of various shapes and sizes may be interspersed between the text areas, it is in many aspects a different problem from that of the segmentation of images of drawings. In drawings, text strings are present as labels, dispersed in many directions, and may be far apart. Most of the algorithms for page segmentation are only applicable when the majority of different types of printed material are segregated.

There exists a variety of page segmentation methods, each of them based on different hypotheses about the image of a document page. They exploit different properties of a scanned document, some of which are restricted to only certain types of documents. The segmentation performance can be measured in terms of the accuracy of the description and the processing speed. One further significant aspect, which is often overlooked, is the amount of knowledge acquired about the areas of interest during page segmentation. This knowledge can be exploited in the subsequent page classification stage and possibly during the later stages of document understanding. The more useful measurements of attributes of areas that are available as by-products of page segmentation, the faster and more efficient the subsequent processes will be.

The accuracy of the description of the areas of interest is of fundamental importance. It reflects the ability of a method to identify these areas and is critical to the processes that follow. Current document preparation systems are capable of producing complex layouts with paragraphs of different shapes interspersed among graphical elements also of varying shapes. Such systems are becoming widespread, resulting in an ever increasing number of publications with complex layouts. A wrongly recognized and described region as a result of page segmentation will inevitably lead to poor results in subsequent stages of document analysis or understanding.

Practicality is another important issue. The implementation of a page segmentation method to solve a real world problem must be realistic. A successful method must cope with as many variations (e.g., shapes of regions, skew) in the scanned document as possible. In doing this, it must also not compromise in processing speed.

In this paper, a new approach to page segmentation is described. One of the advantages of this new approach is its ability to process and analyze documents whose printed regions may not be rectangular. The methods involved are designed to be practical in order to be applied to real-world problems. With this in mind, apart from being able to handle complicated layouts, they are also efficient and fast. Extra consideration has been given so that no time-consuming operations are required. Even the presence of considerable skew does not introduce any complications in terms of accuracy and processing time. Furthermore, the page segmentation described in this paper can be tightly coupled with
a new page classification method [1] under a unified approach so that the whole document analysis approach benefits from the added efficiency resulting from this cooperation.

A brief description of some of the most dominant previous approaches follows in Section 2. The rationale behind the new approach is presented next, in Section 3. The page segmentation method itself is described in Section 4 where each stage and aspect is analyzed in a separate subsection. In Section 5, some representative results are presented together with an appraisal of the performance of the methods involved. Finally, conclusions and further discussion are contained in Section 6.

2. PREVIOUS APPROACHES

Several methods and their variants have been employed in page segmentation approaches. The most dominant are briefly mentioned below and their performance is discussed. It should be noted that it is considered outside the scope of this paper to survey all page segmentation methods.

A well-known approach is connected components aggregation. Connected components (connected pixels of the same color and intensity) are extracted from the image and an attempt to deduce the type of each one (character, part of graphics, etc.) is made. Then, components of the same type are iteratively grouped together to form progressively higher-level descriptions of the printed regions of the document (words, text lines, paragraphs, etc.). A number of approaches have used connected components to various degrees. Of these approaches, most recently that of O’Gorman [2] is the most flexible in that it can correctly process skewed images and regions of nonrectangular shapes. A disadvantage is that the identification, analysis, and grouping of connected components can be, in general, a slow process, especially when there are a lot of components in the image (e.g., a newspaper page). Furthermore, the general approach can suffer from the problems of bottom-up strategies, such as incorrect segmentation due to wrong early groupings.

On the other end of the spectrum, page segmentation methods start with larger regions of the image in a top-down manner. Some examples include the application of transforms on the image (e.g., Jain and Bhattacharjee [3]), morphological operations (e.g., Haralick et al. [4]), and the analysis of projection-profiles (histograms of black pixel occurrences along parallel lines in a given direction) [5]. The application of image transforms is quite time-consuming and in some cases regions of different types but with similar texture are confused or merged. The computational cost of the morphological operations is also relatively high with the further potential problem of having to choose an appropriate structuring element to globally process the image. The analysis of projection-profiles involves the iterative subdivision of the page image along horizontal and vertical “gaps” between foreground elements (e.g., space between text columns and paragraphs). As the dividing lines have to be identified in the image data at each iteration, it can be a slow process. Furthermore, the vast majority of the above approaches assume explicitly or implicitly that the printed regions in a document page are rectangular (i.e., they only process Manhattan layouts). This fact restricts the application of these methods to a very useful subset of document images which seems to be of decreasing size.

An approach which is unique in the analysis of Manhattan layouts is that of Baird [6]. It is based on the identification of background space which can be described by horizontal and vertical rectangles. After the connected components in the image have been extracted, all maximal rectangles fitted to the background are enumerated. A suitable subset is then selected using a heuristic shape score and a stopping rule for the search. The rectangles belonging to the selected set covering the background are unified and used as a mask to extract the foreground areas from the image data. This approach shares a similarity with the one presented in this paper in that it is based on the philosophy of identifying and analyzing the background region separators first. A more flexible approach that copes with skewed page images but still requires the printed regions to be rectangular is that of Pavlidis and Zhou [7] who coined the term “white streams” to denote the horizontal and vertical white space separators in the relevant class of document images. It should be pointed out, however, that this approach differs from the one presented here in that it is not based on the analysis of the background structure; it identifies regions by performing an aggregation of horizontal strips identified as printed matter between two white space gaps.

3. THE WHITE TILES APPROACH

There is an ever increasing number of publications which do not have the “traditional” layout where printed regions are rectangular. Text paragraphs and areas of graphic type may be of any shape and in any arrangement. An example of a page whose layout includes printed regions of different shapes can be seen in Fig. 1. In this figure, the irregularly shaped graphic area at the lower part of the page has text regions of varying shapes “wrapped” around it. Also in Fig. 1, it can be observed that printed regions are surrounded by streams of white space. This is a typesetting convention [8] used in practically all documents. The white space plays the dual role of both a delimiter of each printed region and of a separator between adjacent (but separate) regions. Looked at from a distance, this delimiting space can be visualized as an irregular white net, with the printed regions thought of as the holes.

The objective of the segmentation method described here is to reconstruct (by means of white tiles) the irregular net of background space in order to identify the precise contours of the printed regions. This is a very flexible approach since, by identifying the space around it, a printed region can be segmented, whatever its shape. Furthermore, regions can be identified even in the presence of severe skew since they will still be surrounded by space. This fact makes unnecessary the time-consuming operation of skew correction, which is required in almost all other page segmentation methods. In addition to its flexibility, the white tiles approach produces very accurate descriptions of the
printed regions. This is very important for subsequent stages such as page classification, the reconstruction of the document structure, the logical analysis of the layout and OCR, to name but a few.

Another significant advantage of obtaining a description of the background at the segmentation stage is the fact that this information can be usefully exploited in subsequent stages. For instance, the data produced as by-products of the segmentation (e.g., description of space inside printed regions) can be used in the derivation of features for page classification, thus, making it unnecessary to perform time-consuming computations at that stage. Furthermore, the structure of the background space can provide useful information about the geometric and logical relationships between printed regions.

4. THE METHOD

To segment the printed regions, the background space surrounding them is identified and described by white tiles. A pre-processing step is usually necessary before that, if the textual printed regions required correspond to paragraphs or columns, rather than text lines. The tiles describe the background completely, in a flexible way which ensures that the shape of the delimiting streams of white space is faithfully preserved. Having a representation of the background in terms of white tiles, the contours of the printed regions can be selected by constructing, for each region, the appropriate edges of the white tiles that border with that region.

In the following sections, the three steps (pre-processing, identification of white tiles, and segmentation) involved in the white tiles page segmentation approach are described in detail. Throughout this paper it will be assumed that the desired result of page segmentation is to identify text paragraphs or columns (rather than text lines or characters) and graphic areas. In this case, it should be noted that for the method to be applied successfully, a single page image should contain at least a small number of text lines (for effective pre-processing as well as for obvious reasons according to the definition of page segmentation as given in Section 1). For the purposes of the description of the method, it will be assumed that text lines are printed in a loosely horizontal direction (intended to be read without rotating the paper). This assumption does not restrict the capability of the method to process documents in which the text is printed along vertical lines. Should there be any need to process such documents, only an adjustment (manual or automatic) in the direction in which the document is analyzed will be needed.

4.1. Pre-processing

For humans, it is quite straightforward to identify the streams of white space that form the irregular net. However, it requires more thought to do the same by computer. The reason is that in a document page there is an abundance of white space apart from that belonging to the delimiting streams. There is also space between text lines of the same paragraph, between words and characters in the same text line, and inside characters themselves. The goal of this pre-processing step is to simplify the effort of the subsequent steps in identifying the appropriate streams of white space. This is done by blocking out or isolating white space areas that do not form part of the delimiting streams.

In some approaches, static smearing processes are performed in which the value is pre-specified (e.g., Wahl et al. [9], Pavlidis and Zhou [7]). This may produce erroneous results as different documents are encountered. In other approaches, time-consuming computations such as the Hough transform are performed to derive the smearing value. In the approach taken in this method, it will be assumed that text lines are printed in a loosely horizontal direction (intended to be read without rotating the paper). This assumption does not restrict the capability of the method to process documents in which the text is printed along vertical lines. Should there be any need to process such documents, only an adjustment (manual or automatic) in the direction in which the document is analyzed will be needed.

4.1. Pre-processing

For humans, it is quite straightforward to identify the streams of white space that form the irregular net. However, it requires more thought to do the same by computer. The reason is that in a document page there is an abundance of white space apart from that belonging to the delimiting streams. There is also space between text lines of the same paragraph, between words and characters in the same text line, and inside characters themselves. The goal of this pre-processing step is to simplify the effort of the subsequent steps in identifying the appropriate streams of white space. This is done by blocking out or isolating white space areas that do not form part of the delimiting streams.

Text lines in a document are printed horizontally (as mentioned in the previous section, at least for the documents referred to here). Consequently, they can be joined by vertical smearing [9]. As a result, this will isolate the horizontal streams of spaces between text lines from the region-delimiting streams. At the same time, the white space inside characters will be filled. However, care must be taken in choosing an appropriate smearing value so that different printed regions are not merged vertically when they should not be.

In some approaches, static smearing processes are performed in which the value is pre-specified (e.g., Wahl et al. [9], Pavlidis and Zhou [7]). This may produce erroneous results as different documents are encountered. In other approaches, time-consuming computations such as the Hough transform are performed to derive the smearing value. In the approach taken in this method, it will be assumed that text lines are printed in a loosely horizontal direction (intended to be read without rotating the paper). This assumption does not restrict the capability of the method to process documents in which the text is printed along vertical lines. Should there be any need to process such documents, only an adjustment (manual or automatic) in the direction in which the document is analyzed will be needed.

4.1. Pre-processing

For humans, it is quite straightforward to identify the streams of white space that form the irregular net. However, it requires more thought to do the same by computer. The reason is that in a document page there is an abundance of white space apart from that belonging to the delimiting streams. There is also space between text lines of the same paragraph, between words and characters in the same text line, and inside characters themselves. The goal of this pre-processing step is to simplify the effort of the subsequent steps in identifying the appropriate streams of white space. This is done by blocking out or isolating white space areas that do not form part of the delimiting streams.
described here, the smearing value is obtained directly from the image data by performing few simple computations.

The starting requirement is that the smearing value should be adequate to cover the space between text lines but it should not exceed that. Therefore, the space between text lines should be identified and examined. In a page image, text lines belonging to the same paragraph are vertically spaced apart by practically the same inter-line distance (or leading). It can also be observed that different printed regions on the page are vertically spaced apart by a distance larger than the inter-line one.

An efficient technique for the determination of the space between text lines is to obtain the vertical projection-profile of a vertical strip in the image. The height of the strip should equal the number of scan-lines of the image (i.e., full height) in order to maximize the potential of passing through text regions. The width of the strip should not exceed the width of one large or two medium-sized characters. In other words, the strip should be as narrow as possible to minimize computation time, but wide enough to ensure that it will contain useful information for a sufficient range of character widths that may be encountered.

When scanned at 300 dots per inch (dpi), the width of the 12-pt character will be represented by up to 50 pixels while that of the 24-pt one will be represented by up to 100 pixels. The latter value is used as the width of the vertical strip (at 300 dpi) as it will contain enough information for the usual font sizes in a document. Although the strip could be chosen to be wider to cope with larger characters, it was observed during experiments that it is not necessary to do so if only a few characters in the document are actually larger than 24 pt. Furthermore, an important consideration is that in the presence of skew the vertical space between text lines may not show correctly (or not at all) in the profile if the strip is wider.

In a profile, the inter-line distance can be identified as the space between nonzero count rows, i.e., the height of zero-valued row intervals. The inter-line distance, however, cannot be accurately determined directly from the profile of a vertical strip due to the variability in the presence or absence of characters with ascenders and descenders. Nevertheless, characters in a text line sit on the same baseline (e.g., *a* in Fig. 2). Therefore, the baselines of vertically adjacent text lines in the same paragraph will be equally spaced apart. The problem, thus, of identifying the distance between text lines, for the determination of the vertical smearing value, can be substituted by that of identifying the baseline distance. The baseline is easily identifiable in a vertical profile as the large peak at the bottom of a nonzero count region (see Fig. 3). This peak is typical of the character(s) contained in the part of a text line inside a strip. The distance between two such peaks can be measured reliably, as it is not affected by the presence of ascenders and descenders. The baseline distances in a part of a profile are shown in Fig. 3.

In the histogram of distances between lower peaks of consecutive nonzero count regions, the baseline difference shows as a clearly distinguishable peak. The result from the consideration of one strip may be combined with those of additional ones in order to enhance confidence. This method is also effective for images containing relatively little text as can be seen in the example of Fig. 5 which shows the result from the image of Fig. 4.

In the presence of skew, however, the parts of the text lines in the vertical strip do not retain their distinctive appearance in the profile. A part of a profile of a vertical strip in an image with 10° skew can be seen in Fig. 6a. Nevertheless, it can be observed that there is a distinct peak in the profile for each part of a text line in the strip. This peak is the result of the projections along the horizontal lines indicated in Fig. 6b. It is evident that the distance between such peaks in the profile is equivalent to the baseline distance. Therefore, by measuring the distance between consecutive maximum peaks, an equivalent value to the baseline distance is obtained. To cope with the case of nonseparated text line regions in the profile, an attempt is made to break the profile at the appropriate (low and sharply deepening) valleys before identifying the peaks and measuring the distances. If the skew is so excessive that no distinguished peak exists in the histogram (typically over 15°), profiles can be taken in rotated directions until a peak appears. However, in practice this kind of skew is quite rare. According to experiments carried out, values of more than 5° are not common.

The method for the automatic determination of the baseline distance proceeds as follows. First, for each vertical strip to be examined, the valleys are identified in the horizontal-projection
histogram. Each valley is represented as a triplet (left local maximum, local minimum, right local maximum). Second, to recover from the possible effects of severe skew, the local minima of the valleys of each strip are examined. Meta-valleys are identified in the series of these minima. Each meta-valley is a similar triplet to a valley but the three values correspond to local minima instead of histogram counts. If a meta-valley is deep enough (the ratio of its minimum over each of its two maxima is greater than a fixed threshold, 0.6 here) the profile is broken at that place, i.e., the count is set to zero. At the end of this stage, the valleys corresponding to interline gaps on the page are assumed to have local minima equal to zero. Therefore, a baseline peak in the histogram can be identified as the right local maximum of the rightmost valley between two interline gaps (see Fig. 3). The possibility of mistakenly selecting the peak corresponding to character descenders (see peak at the bottom of Fig. 3) is avoided by comparing consecutive peaks starting from the rightmost one. If the right/left peak ratio is less than a fixed threshold (0.5 here) then the left peak is chosen as the baseline peak.

The occurrences of different distance values between consecutive baseline peaks are counted. As mentioned above, the most dominant peak in the histogram of these counts (see the example of Fig. 5) is taken as the main baseline distance. As the overhead of identifying the baseline distance needs to occur only once for each batch of similar pages, the method at present uses all strips of the image. However, an appropriate sampling scheme together with a suitable confidence measure can be found to optimize the identification of the baseline distance for speed, using only a small number of strips. It is considered beyond the scope of this paper to further analyze this issue.

Having determined the baseline distance, the vertical smearing value can now be calculated. As mentioned before, the objective is to identify the minimum value that will result in isolating the space between text lines in a paragraph from the streams of space surrounding that paragraph. Considering the two text lines in Fig. 2, one possibility would be to consider the distance between the descenders of the top text line and the top of the body of the characters, $b$, in the bottom one (or, equivalently, the distance between the baseline of the top text line, $a$, and the ascenders of the bottom text line). This will result in joining two vertically adjacent text lines only at places where descenders (or ascenders) exist, thus leaving intact most of the white space between the text lines. Under this arrangement, white gaps will

---


FIG. 5. Histogram of baseline differences in the image of Fig. 4.

FIG. 6. (a) Profile in a skewed image ($10^\circ$). (b) Example projection direction.
still exist between the corresponding ends (at least) of vertically adjacent text lines. Hence, since the white space between consecutive text lines should not form part of the surrounding space, there is a need to join the text lines more tightly. This can be achieved by selecting, as smearing value, the distance between the baseline of the top text line and the body of the characters in the bottom text line (i.e., the distance between $a$ and $b$ in Fig. 2). This distance is computed as two thirds of the baseline distance. In experiments with various documents, smearing with this value has produced the desired smearing effect while joining of different printed regions is avoided.

The result of smearing the image in Fig. 1 with this smearing value can be seen in Fig. 7. In addition to the fact that the method described here has been designed with the maximization of efficiency in mind, it is only necessary to determine the smearing value for the first document of a batch typeset with the same spacing between the majority of vertically adjacent text lines. It should also be noted that, for text regions, by using this smearing value the white space between characters in text lines is preserved. This enhances the texture of the regions containing text, and it is very important for the efficiency of the page classification stage. Indeed, using this smearing value gives rise to the best results in that stage too [1].

To reduce the computational load and consequently the run-time, the images submitted to the segmentation process can be subsampled. Subsampling a full-resolution image is timewise favorable to scanning a page first at a low resolution for the segmentation process and then scanning it again at a higher resolution for OCR and subsequent processes. The most straightforward and fast way is to discard a number of pixels in each direction (horizontal and vertical). For instance, in a 300 dpi image discarding two pixels after reading one and two scan-lines after reading one corresponds to a reduction of resolution to 100 dpi. The relation of average run-times for segmenting typical A4 page images to different subsampling values is shown in Table 1. In the tests, subsampling to 100 dpi has given equally good results as in the full-resolution image in the sense that the same regions were correctly segmented. Therefore, this scheme was followed in preparing the input image for pre-processing. One exception is that some straight lines, slightly skewed, may appear fragmented in places that have become thinner (due to the sampling of the scanner) in the full-resolution image. Although it is possible to further reduce the image by more subsampling, it does not always produce reliable results for complex layouts. As many black pixels will disappear, the identified printed regions in some cases appear fragmented. An alternative subsampling method would be to reduce the resolution by representing a group of pixels in the original image by a black one in the subsampled image if one or more pixels in the group are black, and with a white one otherwise. However, this could lead to loss of white pixels which are needed for the accurate determination of white tiles, used for segmentation as well as for the later stage of page classification.

### 4.2. Identification of White Tiles

After pre-processing, the white space in the smeared image of the document is examined and described by white tiles. A white tile is represented by a rectangle whose horizontal and vertical dimensions are adjusted to fit the longest possible white space in the horizontal direction. There is no restriction in the height of the rectangle. If the shape of the stream of white space to be described varies sharply in the vertical direction, a white tile may be effectively reduced to a horizontal line (i.e., the two horizontal sides will coincide). To obtain the description of the streams of background space in terms of white tiles, the white runs of each scan-line of the image are identified.
and the white tiles are constructed sequentially from top to bottom. The algorithm proceeds as follows.

1. Obtain the white runs of the first scan-line containing background space
2. Start a white tile for each of the runs
3. For each following scan-line containing white runs
   a. For each white run in the scan-line
      i. If there is an overlapping white run in the above scan-line
         1. If the corresponding ends of the two runs lie very close (see below)
            append current run to the tile to which the above run belongs
         2. Else
            start a new white tile with the current run
      ii. Else
         start a new white tile with the current run.

When a new white tile is created, it has, initially, the horizontal width of the starting white run. As new runs are appended and it becomes larger in the vertical direction, it is also allowed to grow slightly in the horizontal direction if new runs are a little bit wider than the initial one. This small width tolerance is allowed in order to give some flexibility to the fitting process. While a close fit of the description to the actual space in the image is desired, the computation time will be longer in the subsequent steps if a very large number of distinct white tiles are created.

The width tolerance is fixed for any size of white tiles. It reflects the total amount of difference between corresponding ends of consecutive runs, allowed over the whole height of a given white tile. In the above algorithm, when a white run is compared with an overlapping one in the above scan-line, the difference between their corresponding ends is calculated. For instance, if \( x_i \) and \( x'_i \) are the horizontal coordinates of the ends of a run and \( x_{i-1} \) and \( x'_{i-1} \) the corresponding coordinates of the ends of the above run,

\[
\Delta_i = |x_i - x_{i-1}| \quad \text{and} \quad \Delta'_i = |x'_i - x'_{i-1}|
\]

then \( \Delta_i + \Delta'_i \) will be the difference between their corresponding ends (\( i = 1, \ldots, k \) for a white tile containing \( k \) runs). The accumulated difference

\[
\Delta_{\text{acc}} = \sum_{i=2}^{k} (\Delta_i + \Delta'_i)
\]

is stored for each white tile. When a new run \((k + 1)\) is considered to be appended to the white tile, the difference it would contribute to the accumulated difference of the tile is examined and if

\[
\Delta_{k+1} + \Delta'_{k+1} \leq T - \Delta_{\text{acc}}
\]

the run is appended to the white tile \((T)\) is the pre-specified width tolerance). Otherwise, a new white tile will be started with that run in order to follow more accurately the streams of white space.

In the example of Fig. 8, white tile \( A \) was started by white run 1. When run 2 was considered, it was found that the ends difference it contributed was acceptable and, therefore, was appended to white tile \( A \). Note that the dimensions of the white tile have been adjusted and its width is now larger. However, when run 3 is considered, the difference between its ends and those of the above run would cause the accumulated sum of end differences \((\Delta_{\text{acc}})\) to exceed the width tolerance threshold \((T)\). Hence, white run 3 is not appended to white tile \( A \) and a new tile will be started with it as the first run.

The value of the width tolerance \( T \) depends directly on the degree of faithfulness required for the description of the background. For the experiments described here it was set to be 6 pixels in a 300 dpi image. This particular value is set to achieve a very high accuracy of description. It corresponds to an overall width change of 0.5 mm (0.02 in.) on the actual document page. Although a very high accuracy is still achieved, the resulting number of white tiles is significantly less than that resulting without the tolerance. Furthermore, in practice, the number of white tiles produced with this value of width tolerance was not found to introduce any noticeable delay in computations during subsequent steps. Table 2 shows how the average run-times for a typical A4 page image (subsampled to 100 dpi) relate to different values of the width tolerance (the values are quoted for full-resolution 300 dpi images). Furthermore, the relative decrease in the length (number of straight line segments) of identified contours for different width tolerance values is shown in Table 3.

Another point to be made is that a large number of very narrow white areas in the vertical direction usually result from smearing (previous step) regions of text (see Fig. 7). Since the delimiting streams of white space are observed to have at least some minimum horizontal width, white tiles whose width is less than that

### Table 2

<table>
<thead>
<tr>
<th>Width tolerance value (number of pixels at 300 dpi)</th>
<th>CPU time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.67</td>
</tr>
<tr>
<td>3</td>
<td>0.58</td>
</tr>
<tr>
<td>6</td>
<td>0.56</td>
</tr>
<tr>
<td>9</td>
<td>0.55</td>
</tr>
</tbody>
</table>

The average run-times for fully segmenting a typical A4 size page image using different values of width tolerance.
Table 3: Average Decrease in Size of Contour Representation (Number of Line Segments) for Different Values of Width Tolerance

<table>
<thead>
<tr>
<th>Width tolerance value (number of pixels at 300 dpi)</th>
<th>Decrease in contour size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0%</td>
</tr>
<tr>
<td>3</td>
<td>24.9%</td>
</tr>
<tr>
<td>6</td>
<td>36.5%</td>
</tr>
<tr>
<td>9</td>
<td>43.3%</td>
</tr>
</tbody>
</table>

Note. The contours of noise regions are excluded from the calculation.

4.3. Segmentation

At the segmentation step, the contours of the printed regions are constructed from the description of the delimiting streams. In this description, the white tiles that belong to a stream that surrounds each printed region are examined and the border of a stream with the corresponding region forms the contour of that region. Therefore, the contour of a printed region is represented by a list of the white tile edges that border with that particular region. The edges in the list are appended in sequence and each one is connected to the previous and the next ones. The edge list is cyclic (the last edge connects to the first) and each element of any list is unique. The objective is to trace the appropriate edges of the white tiles that constitute the delimiting streams of white space in the image.

As mentioned in the previous section, the white tiles description of the background can be traced as a graph. The vertices in this graph are the white tiles and the edges of the graph represent the vertical adjacency relationships (above or below) between white tiles. An example of a part of a graph along with a visualization of the corresponding white tiles is shown in Fig. 10.

To identify the white tiles that encircle regions, the graph can be traced to identify cycles, one for each region. The cycles required are the minimum ones. In this context, a minimum cycle is defined as one that surrounds a single region. In the example of Fig. 10 the required cycles are AEFGA, surrounding the small region, and ABCDAGFEA which corresponds to the stream encircling the larger region.

![FIG. 9. The white tiles description.](image)
Particular attention is needed in situations such as that of Fig. 10 where a part of a graph could be wrongly thought to consist of two cycles, one inside the other. In fact, if the standard graph theory definition of a cycle is followed, the cycles identified would be $AEFGA$ and $ABCD$A (which encloses both regions). This is because, in the second cycle, the starting vertex $A$ is encountered again and it is considered as its end. To ensure that the appropriate cycles are followed, the tracing method devised here follows simple rules to select the appropriate white tile each time so that only the stream that borders each region is traced.

A problem with tracing cycles is that it can be very time-consuming. Various algorithms to trace minimum cycles first need to identify all possible cycles in the graph and then select the shortest ones from each vertex. In contrast, the tracing algorithm described here sequentially traces all the required cycles one after the other without the need for exhaustive searches. This advantage is based on exploiting the fact that the white delimiting streams on the document page are connected. Another advantage is that each part of a cycle is traced only once (no backtracking), enhancing thus the speed of the algorithm.

The algorithm starts by identifying the starting white tile of the first cycle and, while tracing this cycle, the start(s) of the adjacent cycle(s) will be identified. Therefore, after the first cycle, all the rest will be traced sequentially. At each node (white tile), the appropriate edges or parts thereof are appended to the edge list that represents the contour of the corresponding region.

A white tile with more than one tiles below it has been chosen as the starting point for tracing a cycle. It is easy to identify the first such white tile starting from the top of the image: it is the first or the second one (the first if a region borders with the top edge of the image, the second otherwise). Hence, the first step in the algorithm is to reach this white tile and identify the starting segment of the contour. The segments of the bottom edge of any starting white tile that are not covered by the white tiles below it are the candidate starting elements for an edge list. These are called potential starts. They are not definite starts because they may prove to be parts of a contour that has started elsewhere.

The first potential start is chosen as the starting point of the first contour. The next white tile to contribute part of its edges will be the one below and to the left of the potential start. The tracing continues then as follows.

```python
if direction was DOWN
    if previous white tile was the rightmost above
        if there are white tiles below
            direction will be DOWN
            next will be the rightmost below
            append edge(s) to the contour  (case 1)
        else
            direction will be UP
            next will be the leftmost above
            append edges to the contour
    else
        direction will be UP
        next will be the leftmost above
        append edge(s) to the contour
    if there are more than one white tiles below
        identify new potential starts
    else {previous was not the rightmost above}
        reached a local minimum
        direction will be UP
        next will be the one above and to the right
        append edge to the contour
        identify new potential starts below
else {previous was not the rightmost above}
    reached a local minimum
    direction will be UP
    next will be the leftmost above
    append edge to the contour
    identify new potential starts
```

After reaching the local minimum (at a low part of the cycle) or just circumventing a white tile protruding downward at a high part of the cycle (case 2), the tracing changes direction and continues upward as follows.

```python
if direction was UP
    if previous white tile was the leftmost below
        if there are white tiles above
            direction will be UP
            next will be the leftmost above
            append edge(s) to the contour
        else
            direction will be DOWN
            next will be the rightmost above
            append edge(s) to the contour
    else {previous was not the leftmost below}
        reached a local minimum
        direction will be DOWN
        next will be the one above and to the right
        append edge to the contour
        identify new potential starts below
```

This statement refers to major regions (information components) of the document page. To identify regions inside other regions (e.g., paragraphs inside a sidebar), the segmentation part only of the method must be applied once more on the composite region only (e.g., area enclosed by a frame) as identified in the first pass.
append edge(s) to the contour
else
direction will be DOWN
next will be the rightmost below (case 5)
append edges to the contour
else {previous was not the leftmost below}
{reached a local maximum}
direction will be DOWN
next will be the one below and to the left (case 6)
if it is not the end of the cycle
append edge to the contour
else
close cycle
if there are more than one tiles below and it is not the end of the cycle
identify potential starts

While tracing downward or upward, every time a white tile with more than one tiles below it is found, an attempt is made to identify new potential starts. In all cases, apart from the local minimum situation (case 3), the potential starts of the current tile are examined. Additionally, in the case of the local minimum, if that white tile has only one tile below it then all tiles below that which have only one tile below them are examined (in sequence) until one that has two or more below is found. The potential start(s) of that white tile are then examined. All untried potential starts are kept in a queue while all the identified potential starts, untried and tried (successful or not), are put in an array (the “used” bin).

As more potential starts are identified each time, it is very important that only the new ones are kept in the queue. This will ensure that cycles are not traced more than once, avoiding thus any wasting of computing time. Therefore, each newly identified potential start is checked as to whether it has appeared in the “used” bin. If it has, it is discarded, otherwise it is appended in the queue to be tried in due course. To ensure that all cycles are traced, it is necessary to identify all neighboring potential starts during the tracing of a cycle. This inevitably leads to the identification of a large number of potential starts which have already appeared in the “used” bin. Searching repeatedly the whole “used” bin array each time would be time consuming and inefficient. However, it is observed that the vast majority of the repeated identifications of potential starts occur locally, i.e., during tracing neighboring cycles. Hence, a cache has been implemented to hold the most recent “used” potential starts which have been searched for in the bin. It was found that a cache size of 10 is sufficient as it can adequately hold the required repeatedly occurring “used” potential starts without being large itself to affect the searching performance. This technique reduces the amount of full searches inside the “used” bin very significantly.

Furthermore, it should also be noted that whenever a local maximum is reached (case 6), the horizontal segment of the current white tile is, according to the definition, a potential start. If it is not the start of the cycle currently being traced, it is not considered further as it will definitely not be the start of another cycle. However, as this potential start may have been identified earlier, it may already be in the queue to be tried. Therefore, the queue of potential starts is searched and, if this particular segment is found, it is removed to prevent tracing the same cycle again. Other potential starts possibly present at the same white tile are examined in the usual way and are added or not to the queue.

An example of the way the algorithm proceeds is as follows. Consider the white tile arrangement around the regions in Fig. 11. First of all, the potential starts ab and ef of white tile 1 are identified. The cycle tracing starts and ab is the first segment of the contour. The next white tile to be visited is number 2 which contributes ap to the contour (case 1). White tile 2 has two others below it and its potential start is appended to the queue. The tracing moves to white tile 7 and the local minimum pn (case 3). White tile 7 has only one tile below it, but below that, tile 9 has two so its potential start is identified and subsequently added to the queue. The next tile to trace through is number 5 which adds the edges nm, mk, and kh to the contour (case 5). Tile 7 is encountered again next where another local minimum, hg, is found (case 3). Checking for potential starts below, that of tile 9 is identified again but, because it is in the “used” bin, it is ignored. The tracing continues upward to white tile 4 where its edge gf is added to the contour (case 4) and its potential start is appended to the queue.

When the local maximum segment ef is encountered at white tile 1 (case 6), it is checked whether it is the start of the cycle. Since it is not, and it has already been put in the queue, it is removed in order to avoid tracing the same cycle again. After tile 3 and its edges ed, dc, and cb (case 2), another local maximum, ab (or ba, as order is not important), is found. However, this is also the start of the cycle and, therefore, the tracing of this cycle stops. The next potential start is then retrieved from the queue and another cycle is traced, constructing another contour. In this case, after the contour (ba, ap, pn, nm, mk, kh, hg, gf, fe, ed, dc,
cb) has been completed, the next potential start to be tried will be that of white tile 2, as ef will have been removed from the queue.

As each white tile is traced as a member of the background stream surrounding each printed region, it contributes one or more edges or small segments to the contour under construction. Depending on the existence and horizontal position and extent of any adjacent tiles above and below, different parts of a white tile are bordering with a printed region. For each possible situation there are rules that determine these bordering parts so that they can be appended to the list describing the contour of that printed region. In the tracing algorithm described above, there are six cases arising from different situations. In each of them, the tracing direction to reach the next tile has either remained the same or is the opposite to that followed to reach the current tile. Furthermore, when the tracing direction has changed, different cases are encountered depending on the existence of vertically adjacent tiles. To decide which parts of the current white tile to extract for the contour, the position of the corresponding ends (left or right) of the vertically adjacent tiles are examined relative to the position of the appropriate end of the current one.

To illustrate the above, the different parts of a white tile, extracted in each possible situation, are shown in Fig. 12. Where the tracing continues in the same direction to the next tile (cases 1 and 4), the ends of the neighboring tiles are examined as follows. Depending on whether the end of the above tile or of the one below is to the left or to the right of the corresponding end of the current tile, part(s) of the horizontal edge(s) of the current tile may be exposed. These parts, if they exist, are appended to the contour together with the (always appended) vertical edge. In situations where the tracing direction has to change, due to the absence of a tile to move to next (cases 2 and 5), three of the edges of the current tile are always bordering with the printed region. These three edges will, therefore, always form part of the contour. In addition, parts of the fourth edge may be appended if they also border with the printed region. Finally, in cases of local minima (case 3) or local maxima (case 6) only a part of the pertinent horizontal edge is added to the contour.

The result of the segmentation step applied to the white tile representation of the example image in Fig. 1 can be seen in Fig. 13. It is evident that each contour follows the shape of the corresponding printed region very closely, constituting thus a significantly accurate geometrical description of the region. As a note, the broken (due to thresholding) curve of the perimeter of the CD-ROM depicted in the graphic at the bottom of Fig. 13 has been closely followed inside the white space of the graphic.

The segmentation step is quite efficient for a number of reasons. First, the contours of the regions are identified from the white tile description and not by using pixel-based methods in the image. Hence it is very fast since it does not deal with the vast amount of image data. Second, the cycles in the white tile graph are traced sequentially without any time-consuming

![Fig. 12. Parts of white tiles forming part of the contour in different cases.](image-url)
searches or backtracking. Only the required cycles are identified and no other cycles are. Furthermore, each cycle is traced only once, preventing thus any wasting of computation time. Finally, in the white tile description, only the tiles that form part of the delimiting streams are considered. The rest do not affect the segmentation process at all. Consequently, the computation time for segmentation depends linearly only on the number of white tiles in the streams.

5. RESULTS

The progress of the proposed page segmentation approach was followed throughout the previous sections with results from each of the steps involved applied to the example image in Fig. 1. This section is devoted to a more comprehensive presentation of results showing the performance of the new method in various aspects.
First of all, the performance of the approach on skewed images can be seen in Figs. 14 through 19. To compare the results with the nonskewed example image in Fig. 1, the same image has been used, intentionally scanned with different amounts of skew. In Fig. 14 the skew angle of the image is 5\(^\circ\), it is 10\(^\circ\) in the image in Fig. 16, and the image of Fig. 18 was scanned with 15\(^\circ\) skew. It should be noted that no extra processing was required due to skew in each of these images. It is evident that the results resemble very closely that of Fig. 13.

The image of a document which contains a text region printed on a background of a darker color (bottom right) can be seen in Fig. 20. The thresholding operation has eliminated most of the background, but some noise is present. The result of the page segmentation method is shown in Fig. 21. Another image of a document containing text regions printed in different orientations is that in Fig. 22. In this image the vertical space between the two columns contains an extra vertical solid line. The corresponding result can be seen in Fig. 23. Finally, the result of the method on the traditional layout of a newspaper image (Fig. 24) is shown in Fig. 25. To illustrate the handling of regions in the presence of small fonts, the full newspaper page (A3 size) was first reduced to A4 by photocopying before it was scanned.

As an automated system for large-scale evaluation is not yet available for this type of documents (an initial effort is reported in [11]), the method was manually (by visual examination) tested on a carefully selected sample of 40 pages including a variety of layouts and regions of different shapes and sizes (also text regions were present with fonts of various sizes). The test sample was collected from different publications in a way that ensured that it was representative of the several types of documents encountered. More specifically the sample included page images of advertisements (5\%), newsletters (5\%), technical journals (10\%), newspapers (5\%), and magazines (75\%, of which 15\% were from technical publications and 60\% from general circulation). In terms of layout complexity, 30\% of the page images were Manhattan layouts, 15\% were non-Manhattan but with rectangular printed regions, and 55\% were complex layouts. In terms of the type of the contents of regions, 10\% of the documents contained pure text and 90\% contained combinations of text with nontext regions. Finally, in terms of variation of font sizes, 35\% of page images had a low variation (up to...
3 pt difference), 35% had medium variation (between 4 and 8 pt difference), and 30% had a high variation (over 8 pt, occurring usually in title pages and advertisements).

To reflect the conditions occurring in reality, photocopied versions of the test pages were also scanned in and used in tests. The photocopying process introduced some noise in the image but not an excessive amount. Furthermore, during photocopying as well as during scanning the alignment of the paper page was careful but no particular consideration was given to it, as is the case in an average office environment. Therefore, apart from the purposely skewed test images, all others have a small skew angle. The scanning resolution for the test images was 300 dpi. For segmentation purposes, the resolution is higher than needed. A resolution of 100 dpi gives equally good results. However, with the requirements of OCR in mind, 300 dpi is practically needed.

For the page segmentation method, the images were subsampled to achieve an effective resolution of 100 dpi (see Section 4.1). The parameters of the method are summarized in Table 4.

The images were thresholded by the scheme used by a commercial OCR package (Omnipage Professional) which was HP Accupage. This is an adaptive thresholding algorithm designed to enhance the images of characters printed on backgrounds that have different grey levels (see the bottom-right region in Fig. 20). The above thresholding scheme was adopted in order to obtain images of realistic quality applicable to an everyday task.

The evaluation was carried out by identifying the number of regions which were:

- **e1.** missed (not identified by the method),
- **e2.** noise (regions introduced that do not exist in the input image),
- **e3.** split (the ground-truth region has been identified as more than one region),
- **e4.** merged (more than one ground-truth regions have been identified as a single region).

The following should be noted about the selection of ground-truth regions:

- **g1.** Regions of text separated by normal inter-line spacing are considered as a single composite region. For instance, this is
frequently the case with paragraphs in a column of text in which case the column will be a single ground-truth region.

g2. Nontext regions constituting a distinct logical entity are grouped into one ground-truth region. As an example, the graphic illustration at the top of Fig. 1 is considered a single ground-truth region, irrespective of it comprising disconnected regions.

g3. Words constituting titles are considered together as a composite ground-truth region.

g4. Tables (text entries and graphic lines) are considered as single composite ground-truth regions.

The results show no missed (case e1) and no noise (case e2) regions. The figures for case e3 require some elaboration. Of the regions, 12.3% were reported as horizontally split. These were all instances of isolated text-lines which, with very few exceptions, denote an error of a single text-line belonging to a column of text (case g1) containing a relatively large number of text-lines. A further 8.3% of regions were reported as split in both directions. Of these, 5.8% were splits in graphics (naturally discontinuous or due to thresholding of continuous regions, see Section 6) and 2.5% were title text or table regions. Images of advertisements contributed a higher number of splits to the results than other page images. Finally, the number of merges was quite low with 0.4% of regions reported as horizontally merged and 0.7% of regions vertically merged. Both types of merge were due to a text region being merged with a straight line separator.

The discussion in Section 6 reflects on issues raised by the above results. At this point it should only be noted that the results show that the errors are mostly splits. This is important as merges are more difficult to recover from after page segmentation and have a negative effect in all subsequent document analysis and understanding stages. In contrast, split regions can be joined after page classification [1] and, possibly, after further analysis steps have taken place.

With the conditions outlined in the earlier part of this section, and with the setting for very high accuracy in the description of regions (very small width tolerance), typical run-times (without image I/O) are in the region of about half (0.55) CPU second on a standard multi-user HP 9000/735 workstation. The code is written in C and is not optimized for speed. Issues that will
affect run-time reduction are the following. First, the extent of subsampling; as greater subsampling occurs, there is less image data and the operations involving this data takes less time. Second, but not as significant, the reduction in the faithfulness of description results in some reduction of execution time. In both cases, it depends on the end application to determine the required level of accuracy of segmentation.

6. DISCUSSION

The result images shown in the previous section illustrate the most important issues involved in page segmentation in general and in the application of the white tiles method in particular. In this section, these issues are discussed in more detail in terms of their nature and their impact on segmentation and on later stages. Furthermore, ways to improve the performance of the page segmentation method are examined based on consideration of the discussed issues.

6.1. Special Considerations and Post-processing Possibilities

Throughout this paper, it was assumed that the required size of printed regions of any given type (text, graphics, line-art) is the largest possible one. Each of the required regions is assumed to represent a separate information component of the document. For instance, in the case of text regions, such a component would be a paragraph or, if there is no inter-paragraph space, a column (or part of it) of text. As explained earlier, vertical smearing is performed to join together the parts of each information component. However, when isolated parts of the intended regions have no immediate neighbor above or below, the corresponding regions may appear fragmented. The obvious case is that of graphics and line-drawing regions which may include disjoint components. It must be noted that the smearing approach is primarily designed to group together text lines. This is a conscious decision since in most applications the focus of attention, in terms of recognition and understanding, is primarily on the textual components of the page.

Isolated text lines that are not merged with vertically adjacent ones may appear fragmented if some inter-word space inside them is wider than the threshold set for the minimum width of


FIG. 21. Segmentation result.
delimiting streams (see Section 4.2). A similar situation may also occur with headlines which occupy a single line. Since the vertical distance between headlines and other components is in most cases greater than that between text lines in a paragraph, the description of the headline region may consist of the contour of more than one character regions. This can be seen in Fig. 21, where the two contours in the top-left part describe the headline region (see Fig. 20). Furthermore, if a headline spans more than one line, and the space between adjacent lines is larger than the smearing value, the headline region will be described as a set of contours of the words that comprise it. This last situation will occur in newspapers, for example, where the headline font is very much larger than the one used in the article body text (see Figs. 24 and 25).

However, the description of headline regions, for instance, as a set of contours of words, can be used to an advantage in order to characterize them as such and distinguish them from the main text for the purpose of document understanding. In any case, it will not be difficult for a post-processing mod-

ule to associate the fragments belonging to the same region. This corrective step should preferably be taken after the page classification stage [1] based not only on adjacency and shape-similarity criteria, but also on the type of the contents of each fragment.

Another idea is to use the white tiles method with the requirement for the size of segmented regions changed, for example, to words in text regions, instead of the whole paragraphs or parts of columns (with no difference in the completeness of representation of regions). In this case, there would be no need for vertical smearing. To integrate the isolated parts into a single representation of each region, grouping may be performed guided by rules of similarity and distance. Depending on the information required, such groupings may be performed either at the end of the segmentation stage or after all regions have also been classified.

Some further exceptions, related to the distance between regions of different types, are situations where this distance is smaller than expected, i.e., similar to the distance between parts of the same region. A familiar example is that of small text regions connected to drawings or inside regions of graphics. This
is a general problem in document analysis. In this case, at the segmentation stage described here, these regions are assumed to be part of the whole graphic information component. The understanding of these regions as well as the recognition of the text in them is the responsibility of later stages.

Moving on to a different aspect of document layout, in some documents certain text regions are printed on background of different tones of grey, or in color. To compound the problem, the color in which the text is printed may vary between different regions. In the case of dark text printed on a lighter background, an efficient thresholding process will turn the text into black and almost remove the background. An example is the bottom-right region in Fig. 20, where the grey background has been largely eliminated. However, such a thresholding scheme is tuned to enhance the regions of characters and, therefore, some parts of grey-level images may also disappear (see, for example, the image regions in Fig. 1). In the case of color documents, more problems are encountered as the image is binarized. Although the performance of thresholding itself is not one of the issues addressed by the white tiles approach, the fact that some information is lost affects all subsequent document analysis stages. For instance, a thresholded image region may appear fragmented and correspondingly its description will be in terms of the contour of its various (now disjoint) components.

The description of each region in terms of its contour depends on the boundary between the surrounding space and the printed region. As smearing joins components of the same region together in the vertical direction, the top and the bottom borders of the region may have a jagged appearance in the image. This is due to the vertical space inlets which have not been sealed in the region. As the description of the region is designed to follow the shape of it, the inlets that are wider than the minimum delimiting stream width threshold are also described in the contour. The jagged appearance is more common in text regions, where it is created by the space between words (in the first and the last text lines of paragraphs), and in threshold images where some part of them has been turned white (see above). An example of the representation of regions in the presence of such space inlets can be seen in Fig. 24.
### 6.2. Advantages of Method

Overall, the segmentation by white tiles is in many aspects advantageous over previous approaches. First, the new method is very flexible. The input documents may contain printed regions of nonrectangular and often complex shapes. In addition, the orientation of these regions in the document image may not be horizontal either intentionally, as a characteristic of the layout, or accidentally, as skew may have been introduced during photocopying or scanning. Regions that have characteristics as above are handled by the method described here as efficiently as simple regions in nonskewed images. No extra processing is required and the complex regions are segmented and described with the same high degree of accuracy as the simple rectangular ones.

This is in contrast to previous page segmentation approaches. For instance, apart from some computationally intensive methods based mainly on image transforms, all other region-based methods assume nonskewed images, with the exception of the method of Pavlidis and Zhou [7] which performs skew detection first and corrects the skew after segmentation. Furthermore, apart from the fact that most of the past approaches are not able to segment complex-shaped regions due to the rectangularity assumption they adopt, others cannot cope with various shapes (Akindele and Belaid [12] who deal with simple isothetic-polygonal shapes only) or cannot describe the regions accurately (e.g., Pavlidis and Zhou [7] where a single region of complex shape will be identified as a set of rectangular ones).

The method described in this paper exploits the fact that the regions are surrounded by background space and efficiently identifies it and segments the regions. The most important difference with other methods analyzing the background is that it is the only one that flexibly identifies the region-surrounding streams of space, whatever their shape or orientation. Since the surrounding background space exists even in the presence of severe skew and complex-shaped regions, the performance of the method is not affected (taking into account reasonable digitization errors). The global nature of the strategy adopted by this method has the flexibility of bottom-up methods while minimizing the consequences of wrong groupings. Furthermore, it does not rely on assumptions about the regions as the top-down methods do.

Another advantage of the flexibility of the white tiles method is that each identified region can be processed and analyzed individually. This is very helpful, for instance, when different regions are printed in different orientations. Each of them can be treated individually, when and if it is needed. If the regions are treated all in the same way, as a result of applying a method on the whole of the image, errors will occur. In the case where different regions are printed in different orientations for instance, it will not be possible to globally estimate and “correct” the skew.

On the aspect of computational efficiency, the white tiles method is designed to avoid computationally intensive processes and minimize accesses and manipulation of the voluminous pixel-based image data. First, as mentioned above, there is no need for skew detection and correction. Second, the description
of the background space by white tiles is efficient, both in the number of tiles and amount of data (much fewer tiles than pixels or connected components, for instance) and in the ease of indexing and tracing of the graph. Therefore, after the white tiles are identified in the early steps of the method, the rest of the processing is performed on this description of the space, rather than on the pixel-based image. Third, the algorithms used in the method have the benefit of requiring few accesses of the data and perform simple computations, hence, they are fast. The identification of white tiles requires only one pass of the pixel-based image data. The tracing algorithm, which acts on the graph representation, identifies precisely the needed contours in a sequential way so that no exhaustive searches and neither backtracking nor unnecessary processing of nonvalid contours are performed. Finally, as part of a unified approach, the information about the white tiles produced in this segmentation stage can be used in the classification stage that follows [1], eliminating thus the need for time-consuming image accesses and complicated feature derivation operations.

REFERENCES


