MVA '92 IAPR Workshop on Machine Vision Applications Dec. 7-9,1992, Tokyo

SEPARATION OF TEXTUAL AND NON-TEXTUAL INFORMATION WITHIN MIXED-MODE DOCUMENTS*

Frank Hoenes & Rainer Zimmer

German Research Center for Artificial Intelligence (DFKI)
P.O.Box 20 80, D-6750 Kaiserslautern, Germany
Phone: (+49) 631 205 3480, Fax: (+49) 631 205 3210, e-mail:hoenes@dfki.uni-kl.de

ABSTRACT

An increasing number of comfortable publishing systems nowadays leads to documents containing more than just textual information. Graphics and images are combined with text and often overlap one another. In this paper we present a robust algorithm for separating textual information from non-textual within multi-mode documents without recognizing individual characters. The approach generates connected components and classifies them as text or non-text. As result, a credibility for each connected component is calculated which expresses its similarity to text or graphics. Moreover, strings are generated that represent sequences of connected components classified as text. Strings can be aligned in any direction. The main processing steps of our system are connected component generation, neighborhood analysis, and the generation of strings.

INTRODUCTION

The aim of document image analysis is to transform the information of a digitized document image into an equivalent symbolic representation usable by post-ordered services e.g. filing and retrieval systems. In order to guarantee an adequate processing of mixed-mode documents, the discrimination of text and non-text parts is required.

A number of techniques for separating text from non-text within mixed-mode documents are proposed in literature in the past years. But there is no technique considering normal and inverse text in one analysis pass, handling low quality text, where characters are split into several connected components, and tracking text in any orientation. The systems referenced can be classified considering their basic image objects and the corresponding relations.

In [Wahl,Wong,Casey82] an approach is presented that classifies areas of a document as text, graphic, or image. The basic objects are blocks generated by a segmentation procedure. The approach is restricted to a class of documents that can be segmented in rectangle areas of homogenous information modi. [Wang&Srihari89] proposed a similar approach for classifying newspapers. Additionally, this approach enables a further subdivision of text regions based on different font sizes.

A survey of different iconic approaches that classify single image points (pixels) can be found in [Bartneck89]. Here, some elementary image processing techniques are described that consider each single pixel and its local neighborhood. The aim in this approach is to reduce image noise and therefore separate noise from text.

* This work has been supported by Daimler Benz Research Lab, Ulm and is part of the project WIDAN - a cooperation project between Daimler Benz Research Lab, ULM and DFKI.

Another approach that uses alternative primitives is introduced in [Fisher,Hinds,D'Amato90]. The underlying rule-based algorithm classifies connected components generated from a smeared image. For these objects document-specific parameters are calculated which control - in combination with the rule-set - the separation and segmentation process.

The algorithm of Fletcher and Kasturi [Fletcher&Kasturi88] uses another input. Sets of eight-connected black pixels build the input of that system. After filtering very small and large objects, a Hough transformation is performed to non-filtered objects grouping objects associated with a particular line. This transformation has the advantage of globally considering the entire document, but the produced Hough space is difficult to analyze. While clustering the Hough space, text strings are generated where all objects belonging to a line are grouped to strings and designated as text.

Another symbolic approach using black connected components as input is described by [Bixler88]. In the first step a filtering is performed where knowledge about the exact font size of a document is necessary. This phase corresponds to the proper separation task. Errors produced in this phase are not removed. The next steps try to find strings of any orientation. Therefore, every connected component is represented by its center and the resolution of the original image is reduced by a factor depending on the actual font size of the document. A second "image" is produced containing only the centers of non-filtered objects. Consequently, strings are generated by detecting pixels in a local neighborhood lying on a common line. In the last step, connected components are rotated depending on the skew of the line.

The algorithm we propose also uses connected components as basic primitives and tries to classify them as text or non-text. It has some similarities to those referred in the literature, but differs in some essential aspects.

After this overview of existing approaches, the following section describes the single analysis phases of our system and the classes of documents we can handle. The last two sections of the paper engage experimental results and conclude with a summary of the main characteristics of our approach.

AN ALGORITHM FOR TEXT/NON-TEXT SEPARATION

A robust and efficient algorithm for text separation is introduced which is able to extract text of any orientation. Text may also be nested in graphics or images. Text means not only black characters on white background, but also inverse printings (white characters on black background). Generally, text should be oriented on a line in any direction, or to a certain degree aligned along an arc. As mentioned above the algorithm performs no character recognition. It classifies con-

nected components only by their size and arrangement.

A few requirements exist for proper operation: text and non-text pixel groups as well as text pixel groups belonging to different characters should not merge. Generally, text should be aligned along a common orientation (approximate straight line) and every text string should consist of at least three characters. At last text parts should contain less noise.

The algorithm we introduce has similarities to a few techniques proposed in literature. But it differs in two important aspects from all approaches: we are able to handle inverse text and at some critical analysis steps we are fault-tolerant. Therefore, not every preceding analysis step has to produce optimal (intermediate) results. Instead, the combination of the phases guarantees the quality of the approach because mistakes of pre-ordered phases may be corrected [Zimmer92].

Our analysis is divided into 6 phases:

- *Connected Component Analysis*
- *Filtering*
- *Neighborhood Determination*
- *String Generation*
- *Inverse Filtering*
- *Assessment.*

In the first phase we filter both very small and very large connected components. The threshold values for large and small connected components are system parameters and can be modified. To decide whether a connected component is of type text or non-text, not only image object features have to be considered. Objects closely located to a connected component under consideration are also of interest. Therefore, we determine the neighborhood of each object. For reasons of expense we prefer a local neighborhood instead of a global one. Consequently, neighbored objects having a position along a common orientation are grouped to strings. For that purpose we concentrate on distances and orientations between objects. Because the filter in the second phase is very rough, there are connected components of type text that are filtered or not belonging to a string. However, in the inverse filtering phase strings are completed picking up such components.

Because in some cases unreasonable strings are generated, the next phase calculates credibilities of the strings. These credibilities effect - beside the object size - the individual credibility values of the corresponding connected components, too. Based on the resulting value, a decision for text/non-text classification of connected components is made.

In the following sections we explain the single phases in more detail.

Connected Component Analysis: For connected component analysis we use an algorithm called SPRLC (single-pass contour line coding) developed by [Mandler&Oberlaender90]. It is a very efficient algorithm that generates a hierarchy of four-connected black respectively eight-connected white components. All connected components have the same rights, which means: all can be classified as text. Therefore, the image is considered as a hierarchy of several layers where the lowest level represents the white document page and the highest, e.g., an i-dot. As final representation we obtain a tree representation where each node is a connected component and the root represents the entire page.

Filtering*: The aim of this phase is to reduce the huge set of connected components to those which likely seem to be text. Instead of performing a reliable separation, we only want to extract the typical components of a text line. If some text objects are filtered or some non-text ones pass the filter, it does not matter. In later analysis stages, we remove this defect.

Our filter uses very simple and easy-computable features. Tests show that it is sufficient to consider the arithmetic average of width and height as feature. In our tests we filter text objects less than 8 points and greater 24 points. Because text can have any orientation a circumscribing rectangle of a skewed character is often larger than those of an unskewed. In our system we select a corresponding higher $size_{max}$. For filtering, each connected component is marked as non-relevant if the average of width and height is larger than $size_{max}$ or smaller than $size_{min}$.

This phase is necessary because documents we are typically considering contain between 2.000 and 10.000 connected components. A lot of components are very small and consequently are filtered. If we would compute all neighborhoods and all possible strings for each connected component it would be too time consuming. Thus, the next two analysis phases can only be efficiently performed on the reduced set.

Neighborhood Determination: One main task of our separation algorithm is the generation of text strings. If connected components can be grouped to strings, this is a good criterion to classify as text. The difficulty of this phase is that strings can have any orientation. Thus, we cannot predict which objects belong to a specific string. For this reason, we define a local neighborhood for each connected component and all objects lying in this neighborhood are potential string neighbors of that component. The size of the neighborhood area depends on the size of a connected component (arithmetic average of width and height). Consequently, the neighborhood relation is not symmetrical.

For neighborhood determination, the original resolution of the document image is reduced using a certain factor. This reduction factor reduces a German standard letter to a 400x280 image where every connected component within the scanned image is represented by one element in a corresponding two dimensional matrix of pointers. Each element points to a list of connected component it represents. Now, for every object the neighborhood area is combed and the contents of all non-empty cells in the neighborhood are added to its neighbor list. It is important that only objects having the same color can be neighbored. In other words, black objects have only black neighbors and white objects have only white neighbors. Figure 1 shows the neighborhood for one connected component and the corresponding reduced matrix.

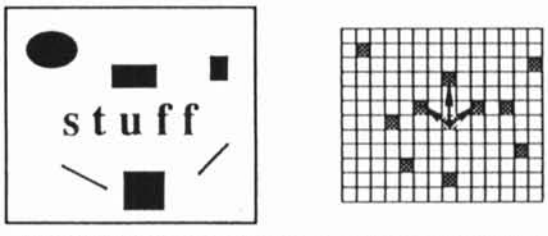

Figure 1: Connected components and the corresponding reduced matrix with neighborhood relations for one object.

The procedure proposed is a simple but very efficient heuristic for determining a local neighborhood area. Instead of considering all objects and comparing them with each other, we can reduce the comparisons to only a few relevant objects close to the current object.

The exact distance between objects is calculated considering the coordinates of the centers of the components in the original image. Based on this distance, the neighborhood lists are sorted in an increasing manner.

*. To avoid misinterpretation of filtering: we understand filtering as hiding irrelevant information, i.e. information unimportant at the current analysis state.

Using these lists we are able to select all objects possibly being right or left neighbor of a connected component within a string. In a next step we have to verify which pairs of objects belong to one and the same string.

String Generation: Before we start to describe the string generation procedure we have to define the term *string*. In our system we consider a string as a sequence of at least three connected components having a common orientation and the distance between these components must be less than D_{String}*average component size. D_{String} is a system parameter which has typically the value 3. Additionally, every connected component can only belong to one string.

As mentioned above, in the string generation phase we only concentrate on objects which are non-filtered - more precisely on their neighborhood relations. Based on these relations we try to generate strings by testing the *start condition*. If an object satisfies the start condition, we generate a three component string which we try to expand on both ends using a *continuation condition* until all components are found belonging to the current string.

As start condition we check if one connected component has two neighbors which do not belong to any string and the two lines - beginning at the current object and ending at its two neighbors - have nearly the same gradient g. Nearly means, that $diff(g_1 - g_2) < grad_{initmax}$, where $grad_{initmax}$ is a system parameter typically defined as $\pi/12$. If the start condition is satisfied, a string is generated consisting of these three objects. The next task is to expand this string capturing all corresponding objects. For that purpose, the current focus is switched to a neighbor object. For string expansion we use the continuation condition. It restricts adding new objects to a string to those objects that do not belong to any string as well as maintain the orientation of the current string. In other words, the gradient g_i from the current object to a stringless neighbor has to satisfy the condition $|g_i - g_{string}| < grad_{contmax}$ for string expansion. If no more neighbors exist, the string expansion is finished at this part of the string. For an entire string expansion, the other end of the string is completed in the same way.

It should be mentioned that not all neighbors of a connected component are tested using the start as well as continuation condition. We only check if they have similar sizes, i.e. the size of a neighbor object must be at least 33% of the current object. Therefore, in most cases we avoid that strings of different font sizes are merged. The string generation procedure ends if no triplet can be found satisfying the start condition.

Inverse Filtering: As mentioned above, sometimes our filtering procedure also filters text objects i.e. i-dots and punctuation marks or if the quality of text is poor, e.g. characters are split into several connected components. In this phase we try to remove these mistakes picking up the erroneously filtered components that belong to text strings.

For that purpose we consider every small connected component that was filtered. We check whether there is another connected component with the same color in its neighborhood belonging to a string. If this condition can be satisfied the current component is attached to the string. For the neighborhood determination we use the reduced matrix described in section *Neighborhood Determination*.

But we have to overcome another problem concealed until now. If characters are split into several connected components, not all components are positioned in a way that the orientation of the corresponding string is kept (e.g. large i-dots or punctuation marks). Consequently, there are small non-filtered text objects not belonging to a string. These objects are also inspected in this phase and treated in the same way as filtered objects.

It is important that we distinguish two different kinds of string memberships. The first one established during string generation attaches an object as a main component to a string. We say a main component expands the string. The other kind of connected components that are considered in this analysis phase are secondary string elements which complete a string. These elements have a relative small size. The distinction in these two kinds of string elements is important for assessing strings.

String and Connected Component Assessment: Based on the information produced we have to decide whether a connected component is text or not. For that purpose we use a decision function *f(cc)* that assigns a credibility to each component. If the credibility is higher than a threshold, the object is classified as text, otherwise as non-text.

A very simple decision function would be the classification of all connected components as text if they belong to any string. But this fails if, for example, there are a lot of lines consisting of only one or two components. Another problem is background noise. If we know that a document contains a lot of noise, it is not adequate to classify all connected components near to a string as text.

For this purpose, we classify connected components using a decision function and therefore are able to adapt our system to specific document classes or demands. To classify connected components we first assess strings. As criteria we consider the number of connected components, the density of connected components within strings, and the number of secondary objects of a string.

As criteria for connected component assessment we use following attributes: *filtered*, *being main component*, *being secondary component*, or *being stringless*.

For each string or connected component criteria, weights exist that allow to individually consider the single features. The assessment function calculates the final credibility of each component which is subsequently compared with a threshold value. Thus, as final result, each connected component gets a text or non-text label which is the basis for the two output representations.

EXPERIMENTAL RESULTS

For testing and evaluating the performance of our system we have analyzed more than 40 documents of different types. Therefore, we digitized the documents with a scanner (interface to a Macintosh). The typical resolution is 200 up to 300 dots per inch. Afterwards, the scanned images were transmitted to a SUN Workstation (IPX) where in a separate phase connected components are generated and stored in a file. Afterwards the separation task was done.

We tested the set of documents with one and the same parameter set. For nearly all documents the separation algorithm yields correct results, that means all characters of a string of more than 3 characters are marked as text components. Problems only occur, if characters of large fonts are merged and therefore are filtered. A second similar problem arises if characters touch graphic elements such as line drawings. These problems can be weakened increasing the scan resolution. Also, some of our documents contain single characters that do not belong to a string. Such isolated characters are not identified as text, because in the parameter set for assessment we only allow text components within a string.

In Table 1, a breakdown of the processing time for single analysis phases applied to three images is given. The

documents are shown in Figure 2 and 3.

Table 1: Breakdown of CPU times for processing of three test images

	Document 1	Document 2	Document 3
Resolution	200 dpi	300 dpi	200 dpi
# CC	3732	2594	4937
CC-Generation (without I/O)	22,1 s (10,3 s)	26,2 s (12,9 s)	50,9s (33,5 s)
Separation	9,9 s	8,4 s	20,5 s
Total Runtime	32,0 s	24,6 s	71,4 s

SUMMARY AND CONCLUSION

A robust and effective algorithm for separating text from mixed-mode documents has been presented. The algorithm detects arbitrarily oriented text and accepts text in various font sizes. Even low quality printings where characters are split into several connected components can be analyzed.

The main components of our system are the determination of a local neighborhood and the generation of strings. The first one detects pairs of objects possibly being neighbors within a string while the second one establishes sequences of connected components having a common orientation.

The advantages of our approach are that we do not perform a definite filtering and thus, are able to correct potential mistakes. We do not restrict that one character is represented by one connected component. Considering German text, one character often consists of two or three connected components. Additionally we are able to consider normal (black) as well as inverse (white) text without performing a separate analysis pass. At last, curved text strings are approximated by short text lines.

REFERENCES

Bartneck89: N.Bartneck; *Methods for Photo Noise Extraction*; Daimler Benz Research Report, Ulm, 1989 (in German).

Bixler88: J.P.Bixler; *Tracking Text in Mixed-Mode Documents*; Proc. Conf. on Document Processing Systems, Santa Fe, New Mexico (1988), pp.177-185.

Fisher,Hinds,D'Amato90: J.L.Fisher, S.C.Hinds, D.P.D'Amato; *A Rule-Based System for Document Image Segmentation*; Proc. of the 10th ICPR, Atlantic City, USA (1990), pp.56- 572.

Fletcher&Kasturi88: L.L.Fletcher, R.K.Kasturi; *A Robust Algorithm for Text String Separation from Mixed Text/ Graphics Images*; IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 10, No. 6, 1988, pp. 910-918.

Mandler&Oberlaender90: E.Mandler, M.Oberlaender; *One-pass Encoding of Connected Components in Multi-Valued Images*; Proc. 10th ICPR, Atlantic City, USA (1990), Vol. II, pp. 64-69.

Wahl,Wong,Casey82: F.M.Wahl, K.Y.Wong, R.G.Casey; *Block Segmentation and Text Extraction in Mixed Text/ Image Documents*; Computer Graphics and Image Processing, Vol. 20, 1982, pp.375-390.

Wang&Srihari89: D.Wang, S.N.Srihari; *Classification of Newspaper Image Blocks Using Texture Analysis*; Computer Vision, Graphics, and Image Processing, Vol. 47, 1989, pp.327-352.

Zimmer92: R.Zimmer; *Separation of Text and Non-text within Mixed-mode Documents*; Diploma Thesis, University of Kaiserslautern, 1992 (in German).

Figure 2a: Example of a scanned document image (document 1 of table 1)

Figure 2b: Resulting image after connected component classification

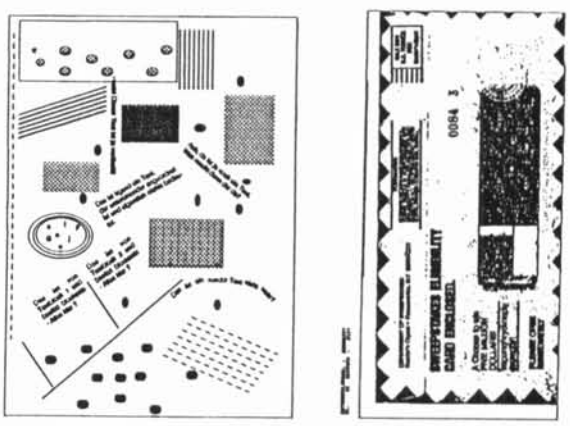

Figure 3: Example documents (document 2 and 3) of table 1.