Segmentation of Text and Graphics from Document Images

S. P. Chowdhury, S. Mandal, A. K. Das
CST Department
B. E. & Sc. University, Shibpur
Howrah 711 103, INDIA
{shyama, sekhar, amit}@becs.ac.in

Bhabatosh Chanda
ECS Unit
Indian Statistical Institute
Calcutta 700 035, INDIA
chanda@isical.ac.in

Abstract

Text, graphics and half-tones are the major constituents of any document page. While half-tone can be characterised by its inherent intensity variation, text and graphics share common characteristics except difference in spatial distribution. The success of document image analysis systems depends on the proper segmentation of text and graphics as text is further subdivided into other classes such as heading, table and math-zones. Segmentation of graphics is essential for better OCR performance and vectorization in computer vision applications. Graphics segmentation from text is particularly difficult in the context of graphics made of small components (dashed or dotted lines etc.) which have many features similar to texts. Here we propose a robust technique for segmenting all sorts of graphics and texts in any orientation from document pages.

1. Introduction

Text/Graphics segmentation is a classical problem in Document Image Analysis and has been reported by many researchers. However, until today we do not have any efficient method for detecting for all type of graphics and texts in any orientation from real life documents. Here, we are mainly focusing on the segmentation of the graphics as well as text from a document page, which is already half-tone segmented, and may not be even fully skew corrected.

This paper is organised as follows. Section 2 describes past research. Proposed method is detailed in section 3. Concluding section (Section 4) contains experimental results and remarks.

2. Past work

Text/Graphics segmentation from document images has been reported by many researchers [1, 4, 6, 7, 8, 9, 10, 12, 15, 16, 18] In many cases only the text strings are separated out thus indirectly segmenting graphics as left-overs and quite a few are in the domain of text graphics separation. Texture based identification is proposed by [9] exploiting the nature of the texture of regular text, which is supposedly different from that of graphics, by using Gabor filters. We have come across many references where graphics are identified from engineering drawings for the purpose of vectorization [2, 8, 15, 17]. However, engineering drawings are special cases of document images containing predominantly graphics portion.

In a nutshell we are aware of three different approaches for separating out graphics from text; they are:

1. **Directional morphological filtering:** The technique is applied to locate all linear shapes and are considered to be text; effectively leaving other objects as graphics. This works well for simple maps [13] and may have inherent problems in dealing with more complex scenes.

2. **Extraction of lines and arcs:** Relying on transform [11] or vectorization [5] many researchers have tried to isolate the graphical objects from text. This approach works well for engineering drawings.

3. **Connected component analysis:** A set of rules (based usually on the spatial characteristics of the components (text and graphics) is used to analyse connected components to filter out the components. The algorithms can handle complex scenario and can be tuned to deal increasingly complex documents. One of the best examples of this class is the work done by Fletcher and Kasturi [7].

Here we elaborate, the well known and frequently referred endeavour by Fletcher and Kasturi [7]. This is a deliberate choice as our approach, in many ways, resembles their approach based on connected component analysis They have done text-string separation from mixed graphics thereby indirectly segmenting the graphics part. On the

other hand, we tried to separate graphics as well as texts from the document containing both text and graphics using simple spatial properties.

The major steps of Fletcher and Kasturi's algorithm are as follows:

1) Connected component generation: This is done by component labeling process which also computes the maximum and minimum co-ordinates of the bounding rectangles.

2) Filtering using area/ratio: This is used to separate large graphical components. The filtering threshold is determined according to the most populated area and the average area of the connected components. The working set to be used by the next step is now reduced to text strings and small graphical components.

3) Co-linear component grouping: Hough transform is used on the centroids of the rectilinear bounding boxes of each component. This operation consolidates grouping of co-linear components.

4) Logical-grouping: Co-linear components are grouped to words by using information like position of each component, inter character and inter word gap thresholds.

5) Text string separation: The words along a text line is segmented from the rest of the image.

The algorithm is robust to changes in font size [1] and style. It is by and large skew independent but sensitive to touching and fragmented characters. The inherent connectivity assumption weakens with images of degraded/worn out documents due to a large number of touching and fragmented characters. As the aspect ratio in graphics and text vary widely the dependence on area/ratio filter is unwieldy. The algorithm is also computationally expensive like any other Hough transform based approach.

Next we present the technique used by us to separate texts as well as graphics which is based on similar techniques but computationally cheaper and more effective in segmenting line art made of dotted or dashed lines or very short line segments irrespective of their orientations.

3. Proposed Method

We have started with gray image of the document and using a texture based technique [3] half-tones are removed. Next, it is converted to binary image by the well known technique proposed by Otsu [14]; and we do the text and graphics segmentation with binary images.

[1] Assumed that the maximum font size in the page should be less than 5 times of the minimum font size used in the same page.

Our approach can be best described through a process flowchart (Fig. 1) showing different bins (text, graphics etc.) containing partly or fully processed text, graphics and other components. Segmentation starts with component labelling to find out extremely small components (likely to be noises), very big components (graphics) and moderately large components (likely to be graphics). It may be noted that for 300 dpi scanning resolution the area allotted for an 8 font character can hold 1000 pixels. We assume that a component which is one hundredth part or less (i.e., $\leq$ 10 pixels) than the said component must be noise and collected in bin N1. Similarly, linear dimension of any character in a typical document may not exceed 2 inches. Thus, component with a horizontal or vertical spread of 600 pixels or more are segmented to bin G1. Components with liner dimension varying between 1 inch and 2 inch are put in G2; they are likely to be graphics but characters in text words may be coalesced (due to poor print quality or for blurred scanning) to form components within that range. Next, a typical feature of English alphabets is used to detect graphics; components embedded within another are put to bin G3 as English alphabets do not exhibit this feature.

Classification continues next by exploiting other features of English alphabets; e.g., the no. of foreground to background transitions in horizontal and vertical directions may not exceed 5 for all alphabets. Components crossing this threshold are kept in bin G4; they are surely graphics. Bin G5 holds components that satisfies the condition of having more than 2 holes (note that 'B' and '&' have two holes) or the no. of foreground to background transitions in either horizontal or vertical direction exceeding 5. Components in bin G3 and G5 may contain text and are filtered out later.

However, graphics made of dotted (or dashed) lines or short line segments are difficult to detect as the size of the individual components are similar to text characters. Therefore, the individual connected component does not signify anything, but a sequence (or group) of such components together represent graphics. The presence of such graphics which are difficult to detect is shown in Fig. 2.

Special care is taken in our approach to segment out graphics made of dotted and dashed lines in any orientation.

Our approach to detect graphics made of small components is based on grouping of small disconnected components supposedly forming a big (spatially) component. In order to segment these cases we have made use of grouping of small components starting from any particular point by observing (i) adjacency, (ii) overlapping, (iii) pseudo continuity, and (iv) stroke width of nearby components.

We start with the characteristics of graphics made with dashed or dotted lines;

1. Number of foreground to background transitions is 1 in vertical or horizontal directions.

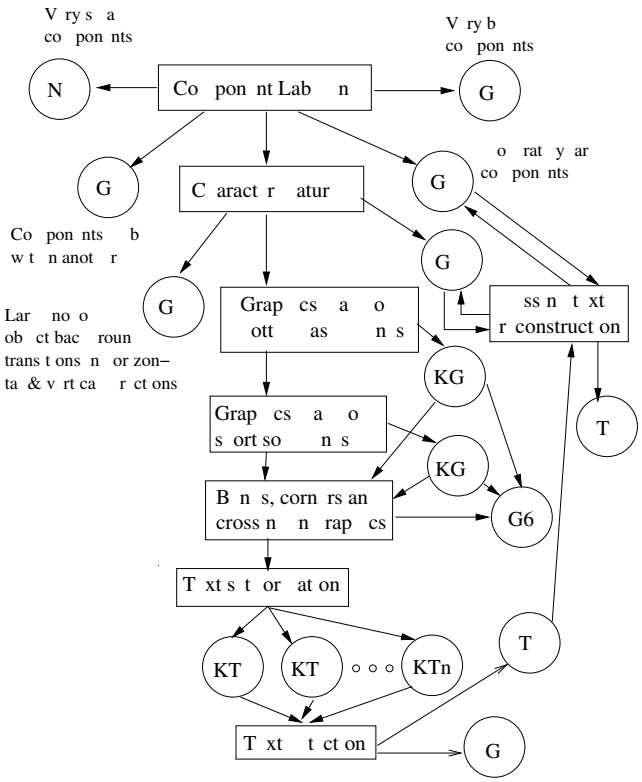

Figure 1. Process flowchart.

2. Ratio of height to width is within a range of 0.5 to 2.

3. Ratio of foreground to background pixels within the bounding box encompassing a component is more than 0.5. This rule is exclusive for dots only.

4. Two components would be treated as adjacent if their extended (5 times) bounding boxes have spatial overlap.

Applying the above rules we could group small components forming part of a graphics object. However, we have introduced more checking to rule out the possibilities of grouping small components available in normal text. We form a group of valid components (small) if the count goes more than 5. This is a check against the possibilities of grouping 5 consecutive dots (dots of i and j) together.[2]. Components grouped with the above mentioned process go to bin KG1.

For grouping of solid short arc segments we use the following rules.

1. Number of foreground to background transitions is 1 in vertical or horizontal directions.

2. The ratio of pen width of the two components will have a range of 0.5 to 2.0.

[2]Consider the word "divisibility" which has got five 'i's.

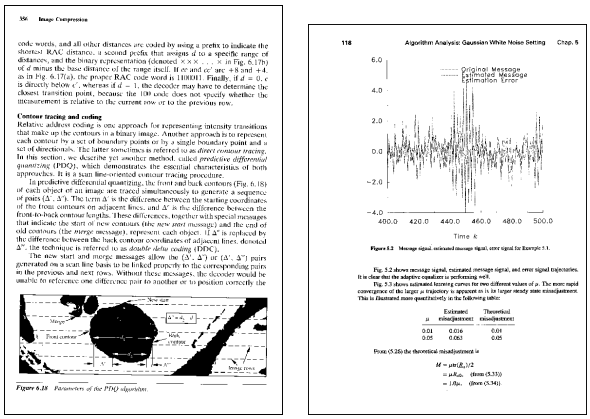

Figure 2. Dotted and dashed lines in graphics.

3. Extended skeleton of the components will have spatial overlaps.

At this juncture; the terms pen width and extended skeleton need be explained.

Pen width is a measure of the stroke width of the arcs forming a graphics. It is computed by drawing four lines (with 0, 45, 90 and 135 degree slopes) from all the points in the skeleton to the object boundary and computing average of the minimum of those radial lines. It is defined as below:

P_w = Average of all skeleton points (Minimum of $(I^0, I^{90}, (I^{45} + I^{135})/2)$)

The pen width is used to verify that adjacent components belonging to a group if the pen width has a permitted variation. Note that for text portion pen width variation is limited and the same is true in case of graphics made up of short lines and arcs where abrupt changes are unlikely.

Extension of the skeleton is done by dividing the skeleton of the component into four parts along the principal axis. So we get three control points (points of divisions) in the skeleton. We make a copy of the lower half and upper half of the skeleton. Take their mirror images and join them from the lower and upper end points (trying to maintain the slope). This effectively extends the skeleton and roughly maintains the original slope at both the ends. So, in one sense it is a pseudo window creation strategy in which the components, in their extended form, come closer to each other. Thereafter, the adjacency conditions are checked to form a group. The components filtered out through this process are kept in the bin KG2.

Adjacency conditions need be fine tuned to accommodate adjacent components in bends, crossings and corners. Without such a measure, a single curved graphics component may be segmented as multiple one as the component at the corners or bends would be the missing links. To accommodate them we will accept the number of vertical or

horizontal transitions be more than 1 but the pen width remains within the range of 0.5 to 2.0. The bends, corners and crossing accumulated are placed in bin G6. Note that this stage re-examines the components in KG1 and KG2 for bends and crossings; components other than bends and crossings in KG1 and KG2 are also placed in bin G6.

The rest of the processes are devoted to find out text components and applied over objects which are not in G or KG bins. This is done to form sets of components that obey the following rules.

1. For a component with maximum linear dimension (height or width; which one is greater) $m1$ there should be another component (belongs to the same set) whose maximum linear dimension is $m2$ such that their extended bounding boxes (i.e., boxes with dimension $m1 \times 3m1$ and $m2 \times 3m2$) overlap.
2. The extent of horizontal overlap should be 1/3 of their areas.
3. Pen width ratios of these components should be in the range of 0.5 to 2.0.
4. Cardinality of the set is at least 4.

The above rules forms set of text components in horizontal direction with slopes upto ± 45 degrees. Similar rules that check vertical overlap do the same for texts primarily in vertical direction with similar slope extensions. Thus it takes an account for the text lines with all possible slopes.

After getting 't' number of such sets KT1 to KTt we apply the following spatial checks to extract remaining graphics from those sets. Note that these sets are primarily words (as components or set members) in text lines with any orientation. The rules are

1. Total number of black to white transitions in a scan line along the principle axis of the set (presumably a text line) is greater than 5.
2. Total number of black to white transitions in a scan line along the perpendicular to the principal axis is less than 5.
3. Maximum width of the rectangle encompassing all the components in the set should be greater than 3 times of the average width of the individual components.

The sets satisfying all the above conditions are put to bin T1 as text and rest, as randomly distributed graphics, in bin G7.

The last step of reconstructing the missing text component is done to fill in the gaps; if any, in text lines. For example we have two consecutive text lines with same slope. Instead of the ideal case of two sets KTx and KTy suppose we have got three (KTx and KTy1 as well as KTy2) due to a missed component in set KTy. These missing components should come from either the bin G3 or G5. A morphological close operation with big structuring element would close components of two adjacent sets and also locate the missing component from the graphics bins. The text lines thus formed are put in bin T2.

The results of the graphics separation are shown in Fig. 3 for a number of cases.

4. Experimental Results and Conclusion

We have carried out the experiments using around 200 samples taken from the UW–I, UW–II databases as well as our own collection of scanned images from variety of sources e.g., books, reports, magazines and articles. The summary of the results is presented in Table 1 for text and graphics zones whose ground-truth information is available in the UW–I and II database. Note that the ground-truth information for our collection is done manually. All experiments are carried out in a Pentium based high end PC and all the programs are written in C. The average segmentation time excluding the half-tone removal is around 3.4 second.

		Actual	
		Graphics	Text
Classified	Graphics	97	2
	Text	3	98

Table 1. Segmentation performance (in %).

The table shows encouraging results for both graphics and text. Some of the texts cannot be clustered as they are dispersed in the graphical portion. Similarly, graphics made of dots/dashed lines may not be fully extracted for obvious reasons. It may be noted that this segmentation algorithm is fully automatic and the parameters used work satisfactorily for a wide variety of fonts and styles.

We conclude this paper with a note that the proposed solution is not radically different from many other efforts including the method proposed by Fletcher and Kasturi [7]. However, we have used a detailed analysis of different types of existing documents to find out a couple of new features best suited for the purpose and optimum adaptive thresholds to cope up with different types of pages. Finally, we claim that the novelty of this work is the detection of the graphics drawn entirely with dots or dashed lines; linking such loosely coupled spatially distributed discrete points as a single graphical unit is a challenge which we could successfully carried out.

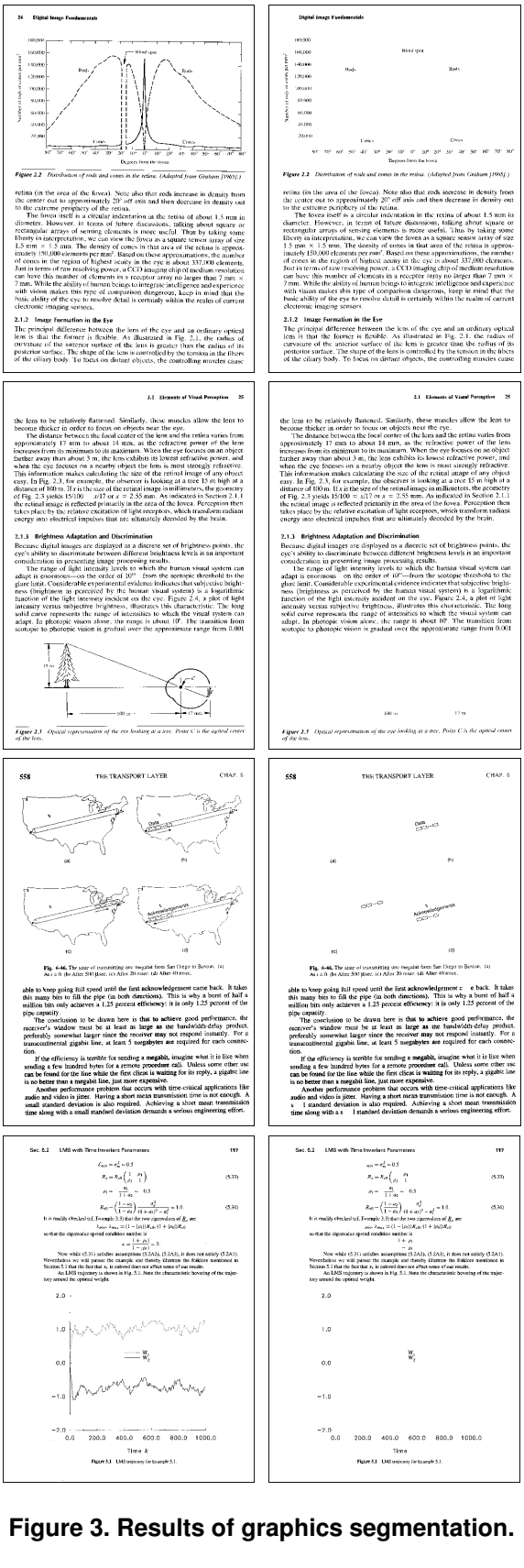

Figure 3. Results of graphics segmentation.

References

[1] O. T. Akindele and A. Belaid. Page segmentation by segment tracing. In *ICDAR'93*, pages 341–344, Tsukuba, Japan, 1993.

[2] J. Chiang, S. Tue, and Y. Leu. A new algorithm for line image vectorization. *Pattern Recognition*, 12:1541–1549, 1998.

[3] A. K. Das and B. Chanda. Extraction of half-tones from document images: A morphological approach. In *Proc. Int. Conf. on Advances in Computing, Calicut, India, Apr 6-8*, pages 15–19, 1998.

[4] A. K. Das and B. Chanda. Segmentation of text and graphics from document image: A morphological approach. In *International Conf. on Computational linguistics, Speech and Document Processing (ICCLSDP'98); Feb. 18–20,Calcutta, India:*, pages A50–A56, 1998.

[5] D. Dori and L. Wenyin. *Vector based Segmentation of Text Connected to Graphicss in Engineering Drawings*, pages 322–331. Springer-Verlag, August 1996.

[6] K. C. Fan, C. H. Liu, and Y. K. Wang. Segmentation and classification of mixed text/graphics/image documents. *Pattern Recognition Letters*, Vol. 15:1201–1209, 1994.

[7] L. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 10 (6):910–918, 1988.

[8] C. C. Han and K. C. Fan. Skeleton generation of engineering drawings via contour matching. *Pattern Recognition*, 27(2):261–275, 1994.

[9] A. K. Jain and S. Bhattacharjee. Texture segmentation using gabor filters for automatic document processing. *Machine Vision and Application*, 5:169–184, 1992.

[10] A. K. Jain and B. Yu. Page segmentation using document model. In *Proc. ICDAR'97, Ulm, Germany*, pages 34–38, August. 1997.

[11] A. Kacem, A. Belaid, and M. B. Ahmed. Automatic extraction of printed mathematical formulas using fuzzy logic and propagation of context. *IJDAR, Vol 4, no 2*, pages 97–108, 2001.

[12] W. Liu and D. Dori. A generic integrated line detection algorithm and its object-process specification. *Computer Vision and Image Understanding*, 70(3):420–437, 1998.

[13] H. Luo and R. Kasturi. *Improved Directional Morphological Operations for separation of Characters from Maps/Graphics*, pages 35–47. Springer-Verlag, 1998.

[14] N. Otsu. A threshold selection method from gray-level histogram. *IEEE Trans. SMC, 9, No. 1*, pages 62–66, 1979.

[15] T. Pavlidis. A vectorizer and feature extractor for document recognition. *CVGIP*, 35:111–127, 1986.

[16] T. Pavlidis and J. Zhou. Page segmentation and classification. *Computer Vision Graphics and Image Processing*, vol. 54:484–496, 1992.

[17] J. Song, F. Su, C. Tai, J. Chen, and S. Cai. Line net global vectorization: An algorithm and its performance evaluation. In *Proc. CVPR*, pages 383–388, Nov. 2000.

[18] F. M. Wahl, K. Y. Wong, and R. G. Casey. Block segmentation and text extraction in mixed text/image documents. *CGIP 20*, pages 375–390, 1982.