Extracting Text from WWW Images

Jiangying Zhou Daniel Lopresti

Panasonic Information and Networking Technology Laboratory
Two Research Way, Princeton, NJ 08540, USA

Abstract

In this paper, we examine the problem of locating and extracting text from images on the World Wide Web. We describe a text detection algorithm which is based on color clustering and connected component analysis. The algorithm first quantizes the color space of the input image into a number of color classes using a parameter-free clustering procedure. It then identifies text-like connected components in each color class based on their shapes. Finally, a post-processing procedure aligns text-like components into textlines. Experimental results suggest this approach is promising despite the challenging nature of the input data.

1 Introduction

A considerable portion of the text on the World Wide Web (WWW) is embedded in images [4]. Moreover, a significant fraction of this image text does not appear in coded form anywhere on the page (see, for example, [5]). Such HTML documents are currently not being indexed properly – image text is not accessible via existing search engines. To make this information available, techniques must be developed for assessing the textual content of Web images.

One of the difficulties to be faced in detecting and extracting text embedded in images is the prodigious use of color combined with complex backgrounds (see Figure 1). Text in WWW images can be any color and is often intermingled with differently-colored objects in the background.

Figure 1: A WWW image containing text[1].

In this paper, we present an algorithm for extracting text from Web images. The algorithm first quantizes the color space of the input image into a number of color classes using a parameter-free clustering procedure. Pixels are assigned to color classes closest to their original colors. The algorithm then identifies text-like connected components in each color class based on their shapes. Finally, character components are aligned to form text lines.

The rest of the paper is organized as follow: in Section 2, we give a brief survey of some related research. Section 3 describes our approach for text extraction. Section 4 presents some experimental results. Finally, we conclude the paper in Section 5.

2 Review of Related Work

Locating text in color images has been addressed under different contexts in the literature. Most of these methods, however, are designed to process scanned images at a much higher resolution than what is available on the WWW. In a paper by Zhong, Karu, and Jain, the authors discussed two methods for automatically locating text in CD cover images [9]. Huang, *et al.* [3] proposed a method based on grouping colors into clusters for foreground/background segmentation of color images. In a paper by Doermann, Rivlin and Weiss [1], the authors described methods for extracting text (often stylized) from logos and trademark images.

The issue of locating and extracting text from WWW images was discussed in an earlier paper by Lopresti and Zhou, which explored the idea of adapting traditional paper-based document analysis techniques to the World Wide Web [4]. In that paper, we proposed using a connected component analysis for text detection. Another related work is the genetic-algorithm-based technique proposed by Zhou, Lopresti, and Zhou for extracting text from grayscale scanned images [11].

3 Text Extraction

Conceptually, our text extraction algorithm follows a paradigm similar to Zhong *et al.*'s first method. Our approach assumes that the foreground color is roughly uniform for a given character. In other words, the color of a single character varies slowly, and its edges are distinct. Like Zhong *et al.*'s method, we first quantize the color space of the input image into color classes by a clustering procedure. We adopt a clustering method which is intuitive and parameter-free. Pixels are then assigned to color classes closest to their original colors. For each color class, we analyze the shape of its connected components and classify them

[1] The images in the figures throughout this paper are originally in color.

as character-like or non-character-like. Here we use a set of features which are different from Zhong *et al.*'s for classification. Finally, a post-processing procedure performs a "clean-up" operation by analyzing the layout of character components.

3.1 Color Clustering

The clustering method we use is based on the Euclidean minimum-spanning-tree (EMST) technique. The EMST-based clustering approach is a well-known method and has been researched extensively over the years. It has been shown that the method works well on a variety of distributions [8, 2].

Given N points in a M-dimensional Euclidean space, the Euclidean minimum spanning tree problem is usually formated as a problem in graph theory: let the N points be nodes and let there be a weighted edge joining every pair of nodes such that the weight of the edge equals the distance between the two points. The Euclidean minimum-spanning-tree is a tree connecting all nodes such that the sum-total of distances (weights) is a minimum. Several robust and efficient algorithms are available to compute the EMST from a given point set [7].

It is easy to see how an EMST can be used as a basis for clustering. By definition, each edge in the EMST of a set of points S is the shortest edge connecting two partitions, P and $(S - P)$, of S. Intuitively, points in the same cluster should be connected by shorter edges than points in different clusters. Hence, by removing the longest edges from the EMST we separate the points of S into disjoint clusters.

To apply the EMST clustering algorithm, we view each color as a point in a three dimensional R-G-B space. For a Web image in GIF format, there are at most 256 unique colors, hence, the number of nodes in an EMST is no more than 256. The distance between two colors is computed as the Euclidean distance of their R-G-B elements. Once the EMST is constructed, we compute the average distance of the edges. Edges whose distances are larger than the average are then removed from the EMST.

The EMST tree thus trimmed may contain several disjoint sub-trees, each of which corresponds to a color cluster. For each cluster, we select the color that occurs most frequently in the image as the representative color of the cluster.

3.2 Text Detection

After the color space is quantized, the next step is to find connected components belonging to each color class. Here we adopt a procedure based on line adjacency graph analysis [6]. In addition to finding connected components in a bitmap, this procedure can extract features such as how many holes a connected component contains, how many upward ends or downward ends it has, *etc.*

The algorithm then classifies connected components into two classes: text-like and non-text-like. The classification is based on the observation that connected components of characters have different shape characteristics than those from, say, graphical objects. A text component is usually composed of a few strokes, with a relatively uniform width, and typically a small number of holes. A connected component from a non-text region, on the other hand, is irregular, with varying-size segments and many holes (see Figure 2).

Figure 2: Components from text and halftone regions.

Currently, the classification process consists of a sequence of conditional tests. A component is not text if:

1) it is very small or very large, or
2) it is very long, or
3) it has very thick strokes, or
4) it has many holes, or
5) it has many branches (ends).

As an example, Figure 3 shows the result of the classification procedure applied to the image of Figure 1. 60 connected components are detected as text-like in this example. 21 of these are part of the text in the image.

Figure 3: Result of the classification process.

Finally, the algorithm performs a "clean-up" operation. It analyzes the geometric layout of the text components. Currently, we assume that there is only horizontal text in the image. In order to eliminate spurious components, we align co-linear text by a horizontal projection. The projection is done by enumerating the number of bounding boxes covering each row. We then analyze the peaks and valleys along the projection profile and separate text components into horizontal text lines accordingly. For each text line, we merge neighboring components to form words. Words which are too short are eliminated. Figure 4 shows the final result of the text detection algorithm applied to Figure 1.

4 Experimental Evaluation

We tested our text extraction procedure on 262 images chosen from real WWW pages. These were selected to cover a variety of graphic design styles commonly seen on the Web. Of the 262 samples, 68 contained fewer than 10 characters, while 53 contained more than 50 characters. The average number of characters in each image was 33.

Figure 4: Result of the text detection algorithm.

Table 1: Text detection performance.

Char. Detect Rate	# of Samples
0% – 10%	58
10% – 20%	18
20% – 30%	11
30% – 40%	13
40% – 50%	19
50% – 60%	17
60% – 70%	22
70% – 80%	25
80% – 90%	20
90% – 100%	59
Ave. detect rate = 47%	

To assess performance, we manually transcribed the original text in each test image as well as the character components correctly identified by the extraction process in the corresponding output image. We then compared the two transcriptions and computed a character detection rate for each sample. Table 1 summarizes the results.

We found that on about 1/5 of the samples, the detection algorithm extracted more than 90% of the text from the input. Figures 5-10 show some of the successful test results. For each example, the upper image is the input and the lower image is the result. (The connected components detected as text-like are shown in their original colors against a background color not appearing anywhere in the original image.) On the other hand, our method failed almost completely on 1/5 of the samples. On the remaining cases, the algorithm was able to detect a varying amount of text.

A close examination of the results reveals that the causes for failure are many-fold. Some of the samples we used in our test are challenging, even for human readers. These include cases where text is wrapped around curves or angled; "embossed" text, outline and handwritten fonts, very small (nearly unreadable) type sizes, *etc.* Figures 11-14 show some of the more difficult samples we encountered.

The color clustering procedure may fail when colors in the input image form a long, chain-like spanning tree. Figure 15 shows an example where the two seemingly opposite colors (black and white) were grouped into one cluster due to the gradually changing intermediate colors in the image. This example shows that trimming the EMST by removing long edges alone may not be sufficient to determine the correct clusters; additional constraints are needed.

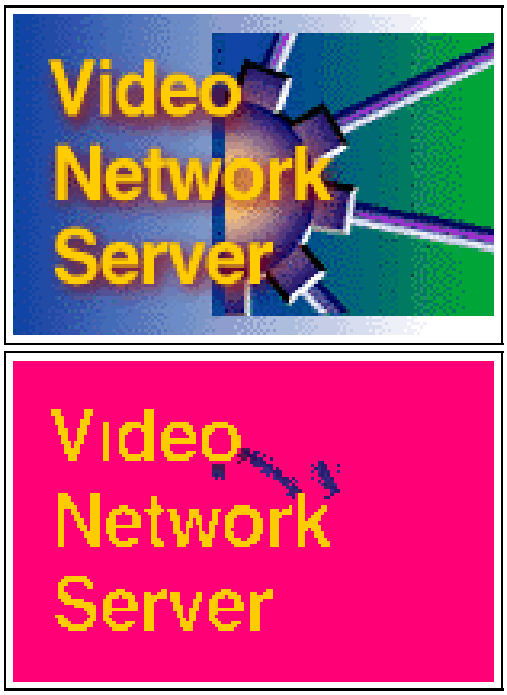

Figure 5: Test example 1.

Figure 6: Test example 2.

Our connected component classification procedure relies on several heuristics to weed out non-text components. Unfortunately, these must be tuned empirically. For our experiment, we tried to develop one "good" set of parameters that would work across a wide range of input images (the same settings were used for all the test samples). However, finding a set of rules which give acceptable results in all situations is difficult. The classification process currently assumes that characters are separable, and that text is horizontal. Since Web images may contain highly stylized graphics, this assumption often causes the classification process to miss text that has broken or touching characters, or that is curved or angled. In Figure 16, we show an example where the text is both angled and touching. Clearly, a way needs to be found to avoid relying on such heuristics during classification.

5 Conclusions

Coded (ASCII) text constitutes only a fraction of the information present in Web pages; a significant amount of the text is embedded in images. In this

Figure 7: Test example 3.

Figure 8: Test example 4.

paper, we have described a method for detecting such text.

Our approach is based on color clustering followed by a connected component analysis. The algorithm works reasonably well given the complexity of the input data, suggesting that such techniques could prove useful in Web-based information retrieval applications. We are currently working on improving its robustness.

A related issue that deserves mention is the recognition of WWW image text. This would be a difficult problem even if the text could be located perfectly. One reason for this is that it is typically rendered at a low spatial resolution (72 dpi). We are currently investigating methods for addressing this. For example, in an earlier paper we proposed using a "fuzzy" n-tuple classifier [10]. The idea is to utilize the information contained in the color bits to compensate for the loss of information due to the low spatial resolution. Initial experimental results suggest that this may be a promising approach as well.

Figure 9: Test example 5.

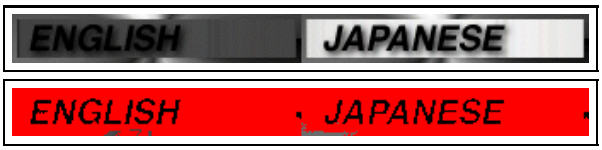

Figure 10: Test example 6.

Figure 11: Difficult example 1 – embossed text.

References

[1] D. Doermann, E. Rivlin, and I. Weiss. Logo recognition. Technical Report CAR-TR-688, Document Processing Group, Center for Automation Research, University of Maryland, College Park, MD 20742-3275, October 1993.

[2] R. Graham and P. Hell. On the history of minimum spanning tree problem. *Annals of History of Computing*, 7, 1985.

[3] Q. Huang, B. Dom, D. Steele, J. Ashley, and W. Niblack. Foreground/background segmentation of color images by integration of multiple cues. In *Proceedings of Computer Vision and Pattern Recognition*, pages 246–249, 1995.

[4] D. Lopresti and J. Y. Zhou. Document analysis and the World Wide Web. In J. Hull and

Figure 12: Difficult example-2 – wrapped text.

Figure 13: Difficult example 3 – outline font.

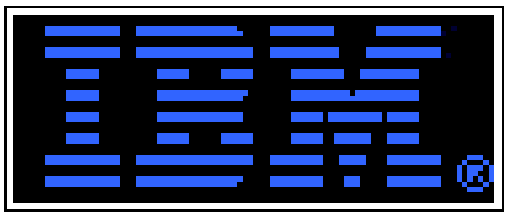

Figure 14: Difficult example 4 – stylized font.

S. Taylor, editors, *Proceedings of the Workshop on Document Analysis Systems*, pages 417–424, Marven, Pennsylvania, October 1996.

[5] The New York Times. http://www.nytimes.com/.

[6] T. Pavlidis. *Algorithms for Graphics and Image Processing*. Computer Science Press, Rockville, MD, 1982.

[7] F. P. Preparata and M. I. Shamos. *Computation Geometry – An Introduction*, chapter 5. Springer-Verlag, 1985.

[8] C. T. Zahn. Graph theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 20(1), 1971.

[9] Y. Zhong, K. Karu, and A.K. Jain. Locating text in complext color images. *Pattern Recognition*, 28(10):1523–1535, 1995.

[10] J. Y. Zhou, D. Lopresti, and Z. Lei. OCR for World Wide Web images. In *Proceedings of the IS&T/SPIE International Symposium on Electronic Imaging*, pages 58–65, San Jose, California, Febuary 1997.

[11] J. Y. Zhou, D. Lopresti, and J. Zhou. A genetic approach to the analysis of complex text formatting. In *Proceedings of the IS&T/SPIE International Symposium on Electronic Imaging*, pages 126–137, San Jose, California, January 1996.

Figure 15: Bad clustering. Left: input, right: result.

Figure 16: Angled and touching text.