A HYBRID TEXT SEGMENTATION APPROACH

Xiaojun Li[1], Weiqiang Wang[1,2], Qingming Huang[1, 2], Wen Gao[2,3] , Laiyun Qing[1]

[1]Graduate University of Chinese Academy of Sciences, Beijing, China
[2] Key Lab of Intell. Info. Process., Inst. of Comput. Tech., Chinese Academy of Sciences, Beijing China
[3]Institute of Digital Media, Peking University, Beijing, China
{xjli, wqwang, qmhuang, wgao, lyqing}@jdl.ac.cn

ABSTRACT

In this paper, we present a hybrid text segmentation approach for embedded text in images, aiming to combining the advantages of the difference-based methods and the similarity-based methods together. First a new stroke edge filter is applied to obtain stroke edge map. Then a two-threshold method based on the improved Niblack thresholding technique is utilized to identify stroke edges. Those pixels between the edge pairs above the high threshold are collected to estimate the representative of stroke color, so that stroke pixels are further extracted by computing the color similarity. Finally some heuristic rules are devised to integrate stroke edge and stroke region information to obtain better segmentation results. The experimental results show that our approach can effectively segment text from background.

Index Terms—Text segmentation, edge filter.

1. INTRODUCTION

Texts in images and videos can play significant role in many applications, such as automatic image annotation, content-based image retrieval, as well as video indexing, summarization, and retrieval. Embedded text detection, segmentation, and recognition [1] are three major key techniques to be addressed to transform pixel-based text into encoded characters which natural language analysis and understanding techniques can effectively process.

This paper investigates the issue of text segmentation which separates text pixels from background in an image. Just as image segmentation is the public challenging task for computer vision community, text segmentation is the same when text is embedded in complex background. In recent years, diverse approaches for text segmentation have been investigated. We categorize the approaches into two main categories: the difference-based methods and the similarity-based methods. The former is based on the color difference between text and background. Some specific filters are used such as canny edge filter and stroke filter [10], and then various thresholding techniques are exploited to form binary images, e.g., Otsu's adaptive thresholding method [2], global & local thresholding method [3], and Niblack's method [4] *et al*. Although they are simple and fast, they generally fail when the color of text and partial background are similar. The latter extract text pixels by clustering pixels with homogeneous color into regions. For example, Fu *et al.* [5] used the K-means clustering algorithm in YCbCr color space to generate several layers, and heuristic rules were employed to select text layer. Wang *et al.* [6] clustered text images into some maps by color and scale of text strokes, and constructed a probability model online to identify text pixels. In Ye *et al.*'s approach [7], they obtained text pixel samples by a reasonable hypothesis, and then used them to train the Gaussian Mixture Models of intensity and hue in HSI color space to characterize text pixels. These methods perform well when all characters have similar color, but sometimes the color inconsistency of characters and noises degrade their performance.

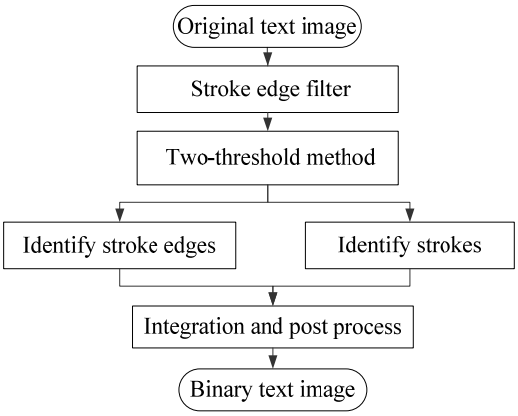


Fig.1. The framework of our approach

The paper presents a hybrid text segmentation approach to combine the advantages of difference-based methods and similarity-based methods together. The framework of our approach is summarized in Fig.1. First, we apply a new stroke edge filter to obtain a stroke edge map. Then an improved two-threshold method derived from Niblack's thresholding is utilized to generate stroke edges. Those pixels between the edge pairs above the high threshold are collected to estimate the representative of stroke color, so that stroke pixels are further extracted by computing the

color similarity. Finally the system integrates stroke edge and stroke region information to achieve better segmentation results.

The rest of the paper is organized as follows: we present the details of the proposed text segmentation approach in section 2. The experimental results are reported in section 3. Section 4 concludes our work.

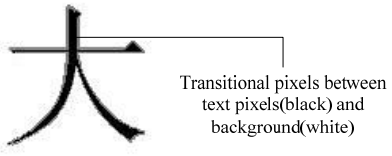
2. DESCRIPTION OF OUR APPROACH

Our system first uses the text detection algorithm in [8] to detect text in images or video frames. Then each located text region is rescaled to a proper size using linear interpolation. When the height of text is less than 20 pixels, the rescaling process can increase the text resolution and at the same time it also makes character strokes have similar expected thickness. In our implementation, text regions are rescaled to a height of 40 pixels and their aspect ratio are preserved. Fig.2 gives three examples of text regions detected from video frames of TV news.

(a)Chinese (b)English (c)Korean

Fig.2. Examples of detected text regions

2.1. Stroke edge filter



(a)An enlarged Chinese character

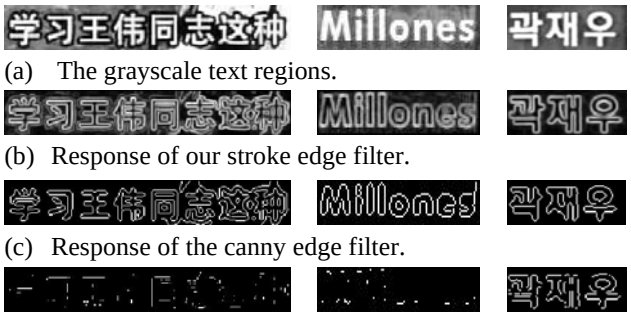
(b)Response of our filter (c)Response of canny filter (d)Response of sobel filter

Fig.3. Reponses of different filters for transitional pixels

As shown in Fig.3 (a), it is common that there exist transitional pixels between strokes and background. Detecting the transitional pixels can help to detect stroke edges. To better identify edges of strokes consisting of transitional pixels, we devise a stroke edge filter as follows:

$$R(p) = \max_{v \in N(p)} \{P(v)\} - \min_{v \in N(p)} \{P(v)\} , \quad (1)$$

where $R(p)$ is the response of the stroke edge filter at location p, $N(p)$ denotes the neighborhood of the pixel p, and $P(v)$ denotes the intensity value of pixel v. Fig.3 (b) ~ (d) show the responses generated by the three edge detectors, i.e., our filter, canny filter, and sobel filter. Apparently our stroke edge filter generates the greatest response and have clearer edges compared with the canny filter, and the sobel filter, especially in the intersection of two strokes. From Eq(1), it is easy to understand that, If a suitable size of neighborhood is chosen, the response of our filter at the locations of transitional pixels is actually the difference of text and background. Fig.4 gives the comparison of stroke edge maps generated by the above three filters for the real image examples in Fig.2. It can also be seen that the proposed edge filter generate more desirable response for character strokes subjectively, compared with the canny or sobel edge detectors.



(a) The grayscale text regions.

(b) Response of our stroke edge filter.

(c) Response of the canny edge filter.

(d) Response of the sobel edge filter.

Fig.4. Comparison of stroke edge maps generated by our filter, canny filter, and sobel filter

2.2. Identify stroke edges

After applying the stroke edge filter, a grayscale edge map is obtained. Then an improved two-threshold method derived from Niblack's thresholding is utilized to classify pixels in the grayscale edge map into stroke edges, candidate stroke edges, and non-stroke edges. Finally, a tracking procedure is applied to find out stroke edge pixels from the generated candidate stroke edges, which employs the connection property among stroke edge pixels and can efficiently remove noises.

Niblack in [9] describes an adaptive thresholding method which computes the threshold for each pixel location p based on the local mean m_p and local standard deviation σ_p of pixels in the neighborhood of p, i.e.,

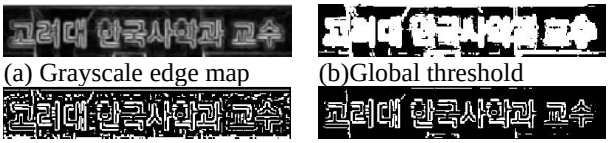
$$t_p = m_p + \alpha * \sigma_p , \quad (2)$$

where parameter α adjusts the ratio of edge pixels. The Niblack's thresholding method generally results in many false stroke edges as shown in Fig.5 (c), especially in dim regions. So we also consider the global information in adaptively choosing a threshold by

$$T_p = \max(m + \alpha_g \sigma, m_p + \alpha_l \sigma_p) , \quad (3)$$

where m and σ denote the mean and standard deviation of the whole grayscale edge map respectively, and α_g, α_l are

predefined parameters. A suitable neighborhood size is very significant to preserve enough local details and at the same time suppress noises. In our implementation, a 10 by 10 square neighborhood is used. As shown in Fig.5 (b), if only a global threshold is used, too many false edge pixels are also generated in bright regions. Fig.5 (d) shows our method can effectively solve these problems.



(a) Grayscale edge map (b)Global threshold

(c)Local threshold (d)Improved threshold

Fig.5. The performance comparison between global, local Niblack's thresholding methods and our method.

To more robustly detect stroke edges, we devise a two-threshold strategy to identify stroke edges. For each pixel, the high threshold T_p^h and the low threshold T_p^l are obtained by setting $\alpha_g = \alpha_l = 0.4$ and $\alpha_g = \alpha_l = 0.2$ respectively in our implementation. Those pixels with intensity values higher than T_p^h are explicitly labeled as stroke edge pixels, those with values lower than T_p^l are explicitly labeled as non-stroke edges, and the remaining pixels are labeled as candidate stroke edge. A line tracking procedure is followed to further find some stroke edge pixels out from the candidates. Concretely, only the pixels labeled as stroke edge pixels are chosen as seeds. For each seed pixel s, an edge line across the pixel will be marked as a stroke edge from the lines in the horizontal(h), vertical(v), diagonal(d), and sub-diagonal(sd) directions, if exist. We define the evaluation function $l_i(s)$ to measure the reliability of line i as a stroke edge. $l_i(s)$ is computed by accumulating votes from stroke edge pixels and candidate stroke edge pixels during the line tracking procedure. For example, starting with the seed pixel s, to compute $l_h(s)$, $l_h(s)$ is updated by collected votes from joined pixels in the extension of a horizontal line, i.e., stroke edge pixels (2 votes), candidate stroke edge pixels (1 vote). The extension of the horizontal line at any end is terminated when a non-stroke pixel is met. Then the line with maximum votes is labeled as stroke edges, and correspondingly all the candidate stroke edge pixels on it become stroke edge pixels. The experimental results for the images in Fig.2 are shown in Fig.6.

(a)Chinese (b)English (c)Korean

Fig.6. The detection of stroke edges.

2.3. Identify strokes

In Fig.6, we observe that each stroke corresponds to a pair of edges, so we can estimate the stroke color by inner pixels between such edge pairs, if the edge pairs can be reliably located. In our implementation, only stroke edges composed of pixels with values above the high threshold are used. For each such edge, the most close one among its parallel edges, if exist, is chosen as its dual edge. The pixels lying on the mid-line of the edge pair are selected as representative samples. Based on the samples, the representative stroke color c can be estimated by color histogram. Those pixels with very similar color with c are classified as stroke pixels. The corresponding results are shown in Fig.7.

(a)Chinese (b)English (c)Korean

Fig.7. The detection of strokes.

2.4. Integrating stroke edge and region information

For the generated binary stroke edges (Section 2.2) and strokes (Section 2.3), four heuristic rules are devised to integrate them so as to obtain better segmentation results. A text region is first partitioned into small square blocks, e.g, 8 by 8 pixels. Then according to the existence of stroke edges or strokes, four possible block types are (a) none, (b) both, (c) only stroke edges, and (d) only strokes. For case (a), all pixels in the block are marked as 0 (non-text). For case (b), all stroke pixels plus the stroke edge pixels which are adjacent to stroke pixels are marked as 1 (text pixels), and the remaining pixels in the block are marked as 0. For case (c), if stroke edge pixels are adjacent to stroke pixels in other blocks and the fill ratio of them for current block is lower than a predefined threshold, the stroke edge pixels are marked as 1, otherwise 0. For case (d), only if stroke pixels are connected with stroke pixels in other blocks, they are marked as text pixels.

Finally connected component analysis is further applied to eliminate non-text blocks based on geometry properties of characters. For instance, connected components whose sizes are smaller than 10 pixels are eliminated. If the pixels lying on the boundaries of the image do not evenly distributed among columns, they are eliminated. Fig.8 gives the final segmented text for the example images in Fig.2.

(a)Chinese (b)English (c)Korean

Fig.8. Results after integration and post process.

3. EXPERIMENTAL RESULTS

To evaluate the effectiveness of our approach, 464 text regions located in 300 color images are selected as test data. The test images come from the Web, recorded broadcast videos, or digital videos, and the embedded text has different sizes, colors, and diverse background. The characters in the dataset involve Chinese, English, and Korean. The distribution of the number of characters and text regions for different languages are tabulated in Table 1.

Table 1. Composition of test dataset

Language	Text regions	Characters
Chinese	225	2161
English	206	2802
Korean	33	249

The performance of the proposed approach is evaluated according to the character extraction rate (CER) and the character recognition rate (CRR). They are defined as:

$$CER = N_e / N, \quad CRR = N_r / N, \tag{4}$$

where N_e is the number of characters completely extracted without obviously lost strokes or connected background residues, N_r is the number of characters which are correctly recognized by an OCR engine, and N is the number of all the characters.

We compare the performance of our method with a difference-based method and a similarity-based method. For the difference-based method, we choose Liu *et al.*'s approach [10], which first applied stroke filter, and then used thresholding method as well as some constraints to segment bright strokes and dark strokes. For the similarity method, we choose Ye *et al.*'s approach [7]. Text recognition is carried out by the commercial OCR software ShangShu 7.0. The related experimental results are summarized in Table 2.

Table 2. Performance comparison of three algorithms

	CER	CRR	Speed(chars/s)
Our approach	88.4%	82.1%	252
Liu *et al.* [10]	81.3%	75.4%	225
Ye *et al.* [7]	82.3%	76.5%	245

The results show that our approach has better performance than the other two approaches according to both CER and CRR. Since our approach utilizes both stroke edges and strokes instead of only one of them as the difference-based methods or the similarity-based methods do, it is less sensitive to the complexity of background and the inconsistency of text color. The proposed stroke edge filter and the improved thresholding method make the extraction of stroke edges and the estimation of stroke color more reliable, which also contribute a lot to the performance of our approach.

4. CONCLUSION

Today most approaches to text segmentation mainly exploit the strong difference between text and background or the color similarity of text pixels. Our preliminary work shows the joint usage of the two kinds of information is an effective way to obtain better segmentation results, since our experiments show results in a gain of about 6% in CRR. The proposed two-threshold method using stroke edge filter can effectively identify stroke edges in subjective evaluation. Reliably detecting stroke edges and strokes is the basis for the hybrid approach to obtain better segmentation performance.

ACKNOWLEDGEMENTS

This work was supported in part by National Natural Science Foundation of China under Grant 60873087, National Key Technologies R&D Program under Grant 2006BAH02A24-2, National Hi-Tech Development Program (863 Program) of China: 2006AA01Z117, and Beijing Natural Science Foundation: 4092042.

5. REFERENCES

[1] R. Lienhart and A. Wernicke, "Localizing and segmenting text in images and videos," *IEEE Trans. Circuits Syst. Video Technol*., Vol.12, No.4, pp.256-268, 2002.

[2] N. Otsu, "A threshold selection method from gray-scale histogram," *IEEE Trans. Syst. Man Cybern*. Vol. 9, pp.62–66, 1979.

[3] F. Chang, G.C. Chen, C.C. Lin, W.H. Lin, "Caption analysis and recognition for building video indexing system," *Multimedia Syst*. Vol. 10 (4) , pp.344–355, 2005.

[4] C. Wolf, J. Jolion, "Extraction and recognition of artificial text in multimedia documents," *Pattern Anal. Appl*. Vol. 6 pp.309–326, 2003.

[5] Libo Fu, Weiqiang Wang, and Yaowen Zhan, "A Robust Text Segmentation Approach in Complex Background Based on Multiple Constraints," *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg, Vol. 3767. pp.594–605, 2005.

[6] Weiqiang Wang, Libo Fu, and Wen Gao, "Text Segmentation in Complex Background Based on Color and Scale Information of Character Strokes," *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg, Vol. 4810. pp. 397-400, Dec. 2007.

[7] Qixiang Ye, Wen Gao, Qingming Huang, "Automatic text segmentation from complex background," *IEEE International Conference on Image Processing*, Singapore, Vol.5, pp.2905-2908, Oct. 2004.

[8] Xiaojun Li, Weiqiang Wang, Shuqiang Jiang, Qingming Huang, Wen Gao, "Fast and Effective Text Detection," *IEEE International Conference on Image Processing*, San Diego, California, U.S.A., pp.969-972, Oct. 2008.

[9] W.Niblack., *An Introduction to Digital Image Processing*, Englewood Cliffs, N.J.: Prentice Hall, pp.115-116, 1986.

[10]Qifeng Liu, Cheolkon Jung, Youngsu Moon, "Text Segmentation based on stroke filter", *Proceeding of the 14th annual ACM International Conference on Multimedia*, Santa Barbara, California, USA, pp. 129-132, Oct. 2006.