A Novel Text Location Algorithm in Complex Color Images

Jiang Yang
School of Information and Communication Engineering.
Dalian University of Technology
DUT
Dalian, China
e-mail: 782282271@qq.com

Li Jian-hua
School of Information and Communication Engineering.
Dalian University of Technology
DUT
Dalian, China
e-mail: jianhual@dlut.edu.cn

Abstract—In this paper, a Connected Component-based Stroke Width Transform algorithm (CCSWT) is proposed, and it is combined with sparse representation and conditional random field (CRF) algorithm to locate texts in color images. Firstly, a color quantization algorithm is carried out to convert a color image into binary image levels. Second we select character candidates in the each binary image respectively and merge them into text line candidates. In every image level, sparse representation is used to obtain the probability of a connected component (CC) being character, and at the same time CCSWT is used to calculate the stroke width of the CC. Then we input the stroke width, other features and the probability of a CC into a CRF model to determine whether the CC is a character. This procedure is executed parallel on every binary image. At last, text line candidates obtained in every binary image are merged as final result. By this way, text in color images can be detected accurately and comprehensively. In the future, the proposed algorithm can be used in image retrieval/understanding.

Keywords-CCSWT; sparse representation; CRF; text detection; CC(key words)

I. INTRODUCTION

The localization of text in nature images is required more and more meticulous, accuracy and speediness with the rapid progress of smart phones and large demands in content-based image retrieval/understanding. The texts within natural images always contain large amounts of information. In order to analyze, recognize, comprehend the information better and apply them, such as the license plate retrieval, image/video retrieval and network filtering, and so on, many approaches for automatic detection and localization of text in images and videos have been proposed [1-14]. Existing approaches can be generally sorted into three categories: edge-based methods, texture-based methods, region-based methods.

Edge-based methods [2] just exploit the feature that the text region always has much richer edge information than the background. These methods detect edges on original image first, and then filter the weak edge pixels, keep the strong edge pixels and merge them into text candidate regions. Finally, heuristic rules are used to locate texts. These methods could exclude large background areas and detect the text regions quickly, but they do not make full use of the color information of texts. When the background is too complicated, these methods always bring a high detecting error rate. So they are usually combined with other algorithms to improve detection efficiency.

Texture-based methods [3-7] use rich texture features of text to detect texts. The texture features used widely in an image include first derivative, second derivative, edge intensity, local variance, FFT coefficient, and various statistical features such as the first moment, the second moment, histogram, co-occurrence matrix. These methods are usually time-consuming, it is because these methods always need to resize origin image many times.

Region-based methods [1, 8-14] make use of the fact that a text line always contains similarly characteristics, e.g., the similar color. So, the clustering algorithm is used to quantify a color image into an image which only contains a few colors [8]. Then the image is decomposed into several image levels. Each image level only contains one color, so it is a binary image. The text lines must be located in some of the binary image levels. The next step, heuristic rules are used to eliminate background regions. Region-based methods are sensitive to text line with color gradually changed, so subsequent processing will be affected. In these methods, texture information is not considered fully, so these methods may increase the difficulty to detect text regions. But they remain a lot of merits, e.g., they are robust to text scale and do not have to change the size of original image. So these algorithms decrease computational complexity. Our method falls into this category.

In this paper, a color quantization algorithm [8] is carried out to convert a color image into several binary image levels. In order to improve recall rate, we do invert and dilation operation to these image levels to get four kinds of derived binary images, then we hunt texts in all these binary images. To improve accuracy rate, in each image level, we use the CCSWT, spare representation and CRF algorithm to find texts. Finally, text line candidates obtained in each image level are merged as final text location results.

II. METHODS

In this section, we detail the overview flow of proposed algorithm. The overview is presented in Section *A*; the CCSWT algorithm is described in Section *B*; sparse representation algorithm is shown in Section *C*; CRF is presented in Section *D*.

A. Overview

The flowchart of proposed algorithm is shown on Fig .1, and the concrete steps are as follow:

(1) A color quantization algorithm [8] is carried out to convert the original color image into several binary image levels. The number of the levels (assumed to be *N*) is the same as the number of color classes in the quantified image.

(2) In order to get a good result, dilation operation and invert operation is done on those binary images, in total, *5N* binary images are obtained.

(3) Do the same operation to each of the *5N* binary images as follow: in a binary image, sparse representation algorithm is applied to each of the CCs to get the probability of a CC being character; The CCSWT algorithm is used to get each CC's stroke width.

(4) The stroke width and probability are inputted into a CRF model, this CRF model decides whether the CC is a character candidate.

(5) CCs judged as character candidates in one binary image are merged if they fulfill heuristic rules in [8], and other CCs are discarded. CCs merged are called text line candidates.

(6) Text line candidates are merged in all binary images as the final result.

B. Connected Component-based Stroke Width Transform

Inspired by SWT [1], we propose an algorithm named Connected Component-based Stroke Width Transform (CCSWT). The CCSWT converts a CC into a width image, and the size of the width image equal to the minimum enclosing rectangle (MER) of the CC.

After locating the MER of a CC in a binary image, the CCSWT scans pixels in the MER along the four directions, 0 degree, 45 degree, 90 degree, 135 degree. If the current pixel is the object pixel in the CC, the CCSWT counts the number of consecutive object pixels including the current pixel on the each direction, and we can obtain four values. The smallest value of them is defined as the stroke width of current pixel in its width image. If the current pixel is not the object pixel in the CC, the stroke width value of current pixel is 0. The average value of all non-zero values in the width image is called the stroke width of the current CC.

One example is showed in the Fig .2(b). The width values of pixel p on four directions are 14, 2, 2 and 2 respectively. The smallest value is 2, so its stroke width value is 2 in the width image. We can see that the values of the corner points in the width image must be 1 by the CCSWT, such as q1~q5. They have no contribution to compute the average width, so we use the following strategy to calculate average width:

In the width image, assume that the total number of non-zero pixels is *Sum* and the number of pixels with 1 is *N*. If $50\% < N/Sum < 90\%$, this CC will be considered as background pixel; else if $N/Sum < 15\%$, those pixels that values equal to 1 will not be used to compute the average stroke width of the CC. That's the CCSWT algorithm.

C. Sparse Representation

Sparse representation model has recently been used in image understanding tasks such as text detection [13] and face recognition [15]. The essence of sparse representation is to represent a signal using an overcomplete dictionary *D*. A number of algorithms can be used to learn *D*, one of the most popular algorithms is K-SVD [16] and it is used in this paper. In this paper, we use character samples and background samples to train two dictionaries *D*, They are used to decide whether a CC is character candidate, and if the CC is character candidate, we calculate probability of the CC being character. The two dictionaries are trained using following training samples:

Character samples: 744 different character binary images with size 16×16.

Background samples: 744 different background binary images with the same size as the character samples. These samples are found in internet.

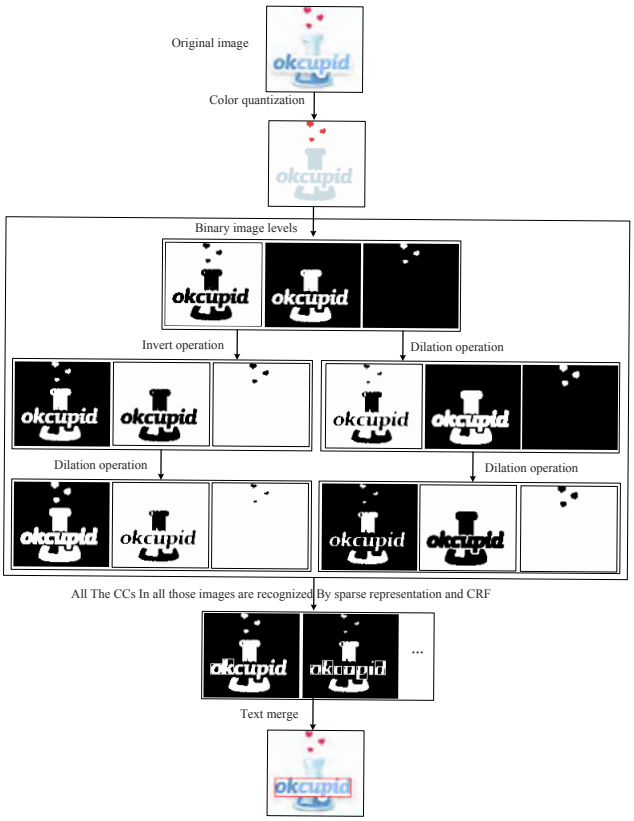
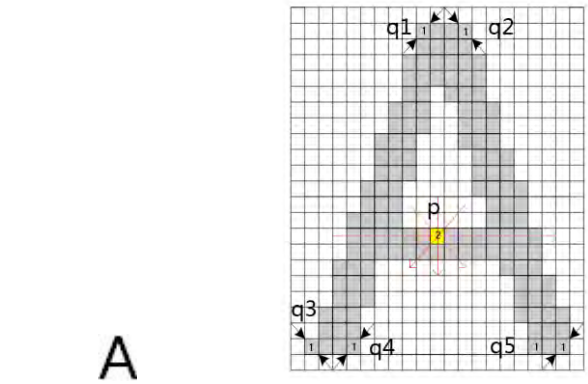


Figure 1. Flowchart of proposed algorithm.



(a) A binary image *I* (b) Pixels of image *I*

Figure 2. CCSWT operation.

Do following operation to training samples: Each training sample image is turned into 256×1 vector, there are only two values in the vector: 255 or 0. Sparseness is set as 8. The size of D is initialized to 256×512. Based on K-SVD, two dictionaries are obtained, *DFont* and *DBack*. *DFont* is the dictionary obtained by character samples training, and *DBack* is the dictionary obtained by background samples training.

Do the same operation to every CC similar to the training samples, transform the CC to a vector (256×1), and it is called C. Restore C by using *DFont* and *DBack* respectively, and the restoration vectors are named *CFont* and *CBack* respectively. Set values in *CFont* (*CBack*) which less than 128 as 0; Set values in *CFont* (*CBack*) which bigger than 128 as 255; the restore error using dictionary *DFont is* defined as:

$$eF = \| CFont - c \|_0 \tag{1}$$

The restore error using dictionary *DBack* is defined as:

$$eB = \| CBack - c \|_0 \tag{2}$$

If $eF > eB$, this CC is discarded as background, otherwise the CC is reserved as character candidate, restore error rate of the CC is calculated as following:

$$eR = \frac{eF}{\| C \|_0} \tag{3}$$

By observation, eR ranges from 0 to 0.5. So we redefine eR as:

$$eR = \begin{cases} 2 \times \dfrac{eF}{\|C\|_0} & if\ eF \le 0.5 \\ 1 & if\ eF > 0.5 \end{cases} \tag{4}$$

Inspired by [14], we use restore error eR to classify character candidates into four kinds of texts: text, probable text, undetermined CCs, non-text. If character candidates fulfill heuristic rules in [8], we call them a group. For a group, we set two thresholds: T_1 and T_2 (in this paper, T_1=0.1, T_2=0.7), and the number of CCs in a group is assumed as N. We count the number of CCs whose eR less than T_1 as N_1, the number of CCs whose eR less than T_2 as N_2, and the number of CCs which eR larger than T_2 as N_3.

If $N_1/N > 0.6$, all the CCs in this group are classified as text; otherwise, if $N_2/N > 0.55$, all the CCs in this group are classified as probable text; otherwise, if $N_3/N > 0.50$, all the CCs in this group are classified as non-text; otherwise, all the CCs in this group are classified as undetermined CCs. Character candidates' *eRs* and the classification of CCs are used to construct following CRF model.

D. Conditional Random Field

A network $G(\mathbf{V}, \mathbf{E})$ is used to model relationship between CCs in the post processing. The V is a set of vertices, and E is a set of edges. The graph G is constructed using the method proposed in [14]. The vertices correspond to CCs, and the weights of edges in the graph correspond to standard energy. Another two special vertices are terminals: source s and sink t. For the text location, terminal s represents text and t represents non-text. The edges connected with s or t are called t-link. The edges between two vertices are called n-link.

For the graph G, we define a similar standard energy to [14] in this paper, but there is a little different. The standard energy has two terms: data term and smoothed term:

$$E = E_{data} + E_{smooth} = \sum_{v \in V} P(L_v) + \sum_{(p,q) \in N} D_{p,q}(L_p, L_q) \tag{5}$$

Where L_v is some features of vertex v, $P(g)$ is a potential function, $D_{p,q}(g,g)$ is an interaction potential function, and N is a set of neighboring vertices. E_{data} is defined as follow:

$$E_{data} = \sum_{v \in V} P(L_v) = \sum_{v \in V} \{ x_v \lambda_v (-\log(eR_v)) + (1 - x_v) \beta_v (-\log(1 - eR_v)) \} \tag{6}$$

Where eR_v is restore error rate of v, x_v = 0 or 1, labels assigned to vertex v. If x_v = 1, $P(L_v)$ means the cost to predict v as text. If x_v = 0, $P(L_v)$ means the cost to predict v as background.

If v has been classified as text, as mentioned in Sec. 2.3, we assign a large cost for t-link between v and source terminal, corresponds a large λ_v in (6), and we assign cost 0 for t-link between v and sink terminal, corresponds $\beta_v = 0$ in (6).

If v has been classified as non-text, we assign cost 0 for t-link between v and source terminal, corresponds $\lambda_v = 0$ in (6), and we assign a large cost for t-link between v and sink terminal, corresponds a large β_v.

If v has been classified as probable text, we assign a larger cost for t-link between v and source terminal and a smaller cost for t-link between v and sink terminal, corresponds $\lambda_v = 3.8$, $\beta_v = 0.8$.

If v has been classified as undetermined text, we assign a smaller cost of t-link between v and source terminal and a larger cost of t-link between v and sink terminal, corresponds $\lambda_v = 0.8$, $\beta_v = 3.5$. It is defined as follow:

$$E_{smooth} = \sum_{(p,q) \in N} D_{p,q}(L_p, L_q) = |x_p - x_q| \cdot (\sigma \exp(-\mu \cdot d)) \tag{7}$$

Where $|x_p - x_q|$ takes 0 or 1. When vertices p and q are predicted belong to a same kind, all text or all non-text, it is 0, and it takes 1 when vertices p and q are predicted as different kinds. σ and μ are constant, in this paper they are set to be 5.7 and 10 respectively. d is a feature distance function between two neighboring vertices p and q. It is defined as the weighted sum of two features distance:

$$d = k_1 d_1 + k_2 d_2 \tag{8}$$

Where k_1 and k_2 are constant, $k_1 + k_2 = 1$. In this paper, k_1=2/3, k_2=1/3. d_1 and d_2 are defined as follow:

$$d_1 = | c_p / w_p - c_q / w_q | \quad (9)$$

$$d_2 = | area_p / (w_p \times h_p) - area_q / (w_q \times h_q) | \quad (10)$$

Where c_p is the stroke width of vertex p (p is a CC), w_p is the width of bounding box of p, h_p is the height of bounding box of p, $area_p$ is the area of p.

To solve the CRF model, maximum flow/minimum cut method [17] is applied to minimize the energy function E in this paper.

III. Experiment Results

In order to compare proposed algorithm with existing methods, we ran our algorithm on public available dataset ICDAR2011. The evaluation criterion of ICDAR 2003 was adopted. ICDAR 2003 proposed precision, recall and *f*-measure as performance indicators. The comparison of different methods is shown in Table Ⅰ. Experiment shows that the recall rate and precision rate of our algorithm achieve 71.20% and 76.12% respectively. It shows that our algorithm has made a great improvement in recall and precision compare with previous methods. Fig .3 shows the text detection results of some images from dataset ICDAR 2011 by our algorithm.

Figure 3. Text detection results of some images from dataset ICDAR 2011.

TABLE I. Comparison of various kinds of algorithm

Algorithm	Evaluating Indicator		
	Precision: p	***Recall: r***	***Standard: f***
Our algorithm	71.20%	76.12%	73.58%
OTCYMIST	64.05%	75.91%	69.48%
SASA	67.82%	65.62%	66.70%
Text Hunter	75.52%	57.76%	65.46%
Ren Zhi-yun	55.54%	71.40%	62.48%

IV. Conclusion

In this paper, we propose the CCSWT algorithm and apply sparse representation and CRF model to text detection field, we get a good result of text detection.

References

[1] Epshtein B, Ofek E, Wexler Y. Detecting text in natural scenes with stroke width transform [C]// CVPR. San Francisco: IEEE, 2010: 2963-2970.

[2] Liu Yang-xing, Goto S, Ikenaga T. A Robust Algorithm for Text Detection in Color Images [C]// ICDAR. Seoul: IEEE, 2005: 399-405.

[3] Chen Xiang-rong, Alan L. Yuille. Detecting and Reading Text in Natural Scenes [C]// CVPR. Washington: IEEE, 2004: 366-373.

[4] Gllavata J, Ewerth R, Freisleben B. Text Detection in Images Based on Unsupervised Classification of High-Frequency Wavelet Coefficients [C]// ICPR. Cambridge: IEEE, 2004: 425-428.

[5] Kim K I, Jung K, Kim J H. Texture-Based Approach for Text Detection in Images Using Support Vector Machines and Continuously Adaptive Mean Shift Algorithm [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2003, 25(12): 1631-1639.

[6] Lienhart R, Wernicke A. Localizing and segmenting text in images and videos [J]. IEEE Transactions on Circuits and Systems for Video Technology, 2002, 12(4): 256-268.

[7] Li Hui-ping, Doermann D, Kia O. Automatic text detection and tracking in digital video [J]. IEEE Transactions on Image Processing, 2000, 9(1): 147-156.

[8] Ren Zhi-yun, Huang Lin-lin. A Fast and Accurate Text Detection Method From Complex Images [C]// IEEE International Conference on Signal Processing. Beijing: IEEE, 2012: 1144 – 1148.

[9] Kim H K. Efficient: automatic text location method and content-based indexing and structuring of video database [J]. J Vis Commun Image Represent, 1996, 7(4): 336-344.

[10] Liu Yang-xing, Goto S, Ikenaga T. A Contour-Based Robust Algorithm for Text Detection in Color Images [J]. IEICE Transactions, 2006, 89-D(3): 1221-1230.

[11] Jain A K, Yu Bin. Automatic text location in images and video frames [J]. Pattern Recognition, 1998, 31(12): 2055-2076.

[12] Neumann L, Matas J. A Method for Text Localization and Recognition in Real-World Images [C]// ACCV. Queentown: Asian Conference on Computer Vision, 2010: 770-783.

[13] Zhao Ming, LI Shu-tao, Kwok J. Text detection in images using spare representation width discriminative dictionaries [J]. Image and Vision Computing, 2010, 28(12): 1590-1599.

[14] Li Rong, Wang Su-yu, Shi Zhi-xin. A two level algorithm for text detection in natural scene images [C]// 11th IAPR International Workshop on Document Analysis Systems. Tours: Vinci-International Convention Centre, 2014: 329-33.

[15] Wright J, Ganesh A, Ma Yi. Robust Face Recognition via Spare Representation[J]. IEEE Transactions on Pattern Analysis and Mchine Intelligence, 2009, 31(2): 210-227.

[16] Aharon M, Elad M, Bruckstein A. The KSVD: an algorithm for designing of overcomplete dictionaries for sparse representations

[J]. IEEE Transactions on Signal Processing, 2006, 54(11): 4311-4322.

[17] Boykov Y, Kolmogorov V. An experimental comparasion of min-cut/max-flow algorithms for energy minimization in vision. IEEE transactions on PAMI, 2004, 26(9), 1124-1137.