Text Detection in Nature Scene Images Using Two-stage Nontext Filtering

Qingqing Wang, Yue Lu, Shiliang Sun
Shanghai Key Laboratory of Multidimensional Information Processing
Department of Computer Science and Technology
East China Normal University, Shanghai 200241, China

Abstract— **We present a text detection method in natural scene images based on two-stage nontext filtering. Firstly, we detect multi-channel maximally stable extremal regions (MSERs) as character candidates. To reduce the amount of repeating components, we merge the MSERs by choosing the most character-like ones when overlap happens. Then nontext components are filtered out by a two-stage labeling procedure, wherein we combine random forests with CRF. Finally, components labeled as text are grouped into words by an edge-cut strategy, and false positives are eliminated by a HOG-based classifier. The experimental results on the ICDAR2013 database show the effectiveness of the proposed method.**

Keywords—text detection; MSERs; CRF; nontext filtering; random forests; edge-cut

I. Introduction

Text in images conveys important information, which helps to make images more understandable. Applications based on text reading in natural scene images, such as content-based image retrieval, automatic navigation, signboard/plate recognition, and computerized aid for visually impaired, are receiving intensive attention in recent years. Generally speaking, text reading involves two steps: text detection and text recognition. Text detection is to extract text regions from a given image, and text recognition is to translate pixel-based text into readable code. The performance of text detection is crucial to the whole text reading system. But due to the cluttered background and a great variety of text patterns, scales, and fonts, text detection in natural scene images is a challenging task. Additionally, camera captured images often suffer from distorted view, uneven illumination, curved, and shadow effects.

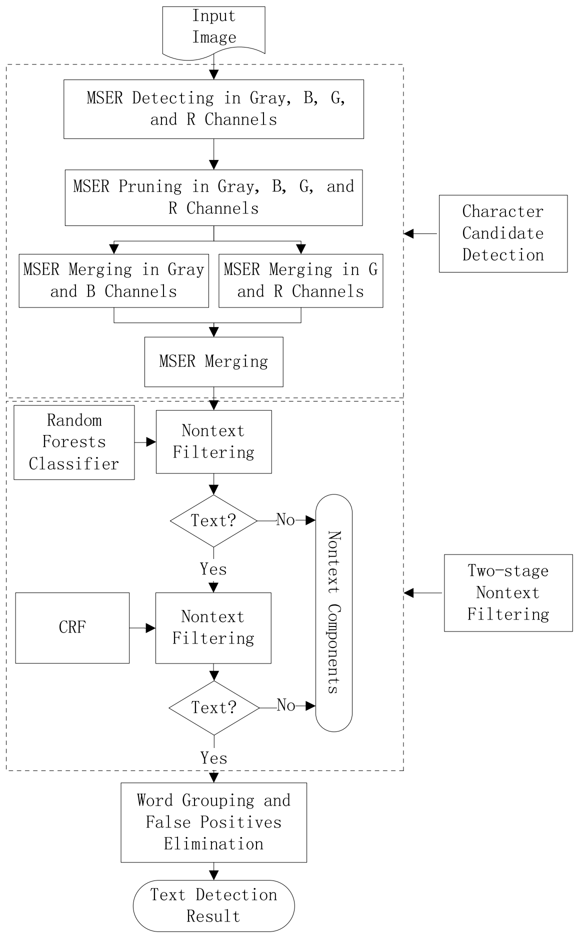
Numerous methods have been proposed to address this issue. Most existing methods can be categorized into two groups: region-based methods [1,2] and connected component-based (CC-based) methods [3, 4, 5, 6, 7, 8]. Region-based methods often utilize sliding window to localize regions with high confidence to be text. Text's distinct features make it possible to differentiate text regions from nontext ones. This kind of methods is robust to noise. However, to adapt to different sizes of scene text, they often leverage multi-scale operators, which possibly leads to long time consuming. On the other hand, CC-based methods are more efficient. They assume that pixels in the same CC have similar local properties, such as gray level, stroke width, and intensity. Additionally, characters can be cropped from images by generating connected components (CCs). The amount of CCs is less than regions, so CC-based methods take shorter time than region-based ones. However, it is a tough task to find a robust CC generator that can deal with blur, skew, low resolution, and uneven illumination.

Gao et al. [1] proposed a region-based method that employed a cascade Adaboost with weak learners of classification and regression tree to decide whether a window contained text or not. A drawback of region-based methods is computation complexity. To alleviate the computation burden, Neumann et al. [2] considered only the regions with text-like strokes. They claimed that their strategy reduced the number of rectangles by three orders of magnitude when compared with the standard sliding window methods. CC generating is a key step of CC-based methods. Bai et al. [3] and Epshtein et al. [4] utilized the appearance of stroke edge pixels and the consistence of stroke width to generate CCs. These CCs were classified as text and nontext in the downstream steps. Recently, ER detector and MSER detector are widely used as CC generator [5, 6, 7, 8] thanks to their high character detection recall. As reported in [5], the character detection recall of ER detector in multi-channels was 94.8%. Yin et al. [6] claimed that their MSER detector achieved a recall of 95.2% in multi-channels. However, the character detection precision was only 7.1% in [5], which implied that many nontext components were detected as character candidates. In MSER-based methods, the elimination of nontext components is the most challenging task. In [5], a sequential classifier combining with incrementally computable descriptors was used to eliminate nontext components. Characters often appear in images coherently, so their adjacency relationship is an available characteristic for character detection. Yin et al. [6] used distance metric learning to compute the distance between MSER pairs, and clustered MSERs into groups according to the distance. Then posterior probabilities of text candidates were estimated to eliminate false positives. Kim et al. [7] employed an Adaboost classifier to learn the relationship between MSER pairs. Then MSER pairs were clustered into groups. Finally, multilayer perceptron learning was exploited to eliminate false positives. Yin's [6] and Kim's [7] methods won the first place in ICDAR2013 and ICDAR2011 competition, respectively.

In this paper, we continue to explore MSER-based text detection in natural scene images. Firstly, multi-channel MSERs are extracted as character candidates, and are merged together to reduce the repeating components. Then a two-stage labeling procedure combining random forests with CRF is used to filter out nontext components. Finally, text components are grouped into words and false positives are eliminated.

II. THE PROPOSED METHOD

Our text detection method consists of three steps: 1) Character candidate detection. 2) Two-stage nontext filtering. 3) Word grouping and false positives elimination. Fig. 1 shows the flowchart of the proposed method.



Input Image
MSER Detecting in Gray, B, G, and R Channels
MSER Pruning in Gray, B, G, and R Channels
MSER Merging in Gray and B Channels
MSER Merging in G and R Channels
MSER Merging
Character Candidate Detection
Random Forests Classifier
Nontext Filtering
Text?
No
Yes
CRF
Nontext Filtering
Text?
No
Yes
Nontext Components
Two-stage Nontext Filtering
Word Grouping and False Positives Elimination
Text Detection Result

Fig. 1. Flowchart of the proposed method.

A. Character Candidate Detection

Inspired by the effectiveness of MSER detector, we extract MSERs from images as our character candidates. Repeating components is the major pitfall of MSER detector. Yin et al. [6] proposed an efficient MSER pruning algorithm to solve this problem. Detecting MSERs in multi-channels can improve the character recall at the cost of a significant increase in the number of nontext MSERs. But it is time consuming to perform the downstream steps on all of the multi-channel MSERs. To deal with this issue, we merge multi-channel MSERs together after pruning algorithm has been performed. This strategy leads to non-overlap of the MSERs.

We use Algorithm 1 to merge MSERs of two images. Considering component sets $R_1 = \{r_1, r_2, ..., r_n\}$ of image I_1 and $R_2 = \{r_{1'}, r_{2'}, ..., r_{n'}\}$ of image I_2, we pick components from R_1 and R_2 to form the resulting component set R.

Algorithm 1: Merging components of image I_1 and I_2

```
Input: component sets $R_1$ and $R_2$
Output: component set $R$
Procedure:
While there exists component $r_i \in R_1$
    If there exists no component $r_j \in R_2$ satisfied $r_i \cap r_j \neq \varnothing$
    Then set $R \leftarrow R \cup \{r_i\}, R_1 \leftarrow R_1 - \{r_i\}$
    Else
        Set $R_1 \leftarrow R_1 - \{r_i\}$
        Initialize $E_1 \leftarrow \{r_i\}$, $E_2 \leftarrow \varnothing$
        Repeat
            Step1: Find component $r_k \in R_2$ where $\exists r_p \in E_1 \wedge r_k \cap r_p \neq \varnothing$
                Set $R_2 \leftarrow R_2 - \{r_k\}$, $E_2 \leftarrow E_2 \cup \{r_k\}$
            Step2: Find component $r_{k'} \in R_1$ where $\exists r_{p'} \in E_2 \wedge r_{k'} \cap r_{p'} \neq \varnothing$
                Set $R_1 \leftarrow R_1 - \{r_{k'}\}$, $E_1 \leftarrow E_1 \cup \{r_{k'}\}$
        Until $|E_1|$ and $|E_2|$ are stable.
        If $\max\{p(r_i)\} > \max\{p(r_j)\}$ where $r_i \in E_1$ and $r_j \in E_2$
        Then set $R \leftarrow R \cup E_1$
        Else set $R \leftarrow R \cup E_2$
        End if
    End if
End while
If $R_2 \neq \varnothing$
Then set $R \leftarrow R \cup R_2$
End if.
```

Where $p(r_i)$ denotes the probability of component r_i to be text, and is estimated by a character classifier. Fig. 2 shows a two-channel merging sample and Fig. 3(b) shows the four-channel merging result of Fig. 3(a). Pixels belonging to the same component are drawn with the same color. Note that, in Fig. 2, letters "S" and "N" in the first row of the board in R channel are interfered by background pixels. The "K" in the first row and "VE" in the second row in G channel are the same cases. And the merging result shown in Fig. 2(d) picks more regular letters "S" and "N" from G channel and "K", "V", and "E" from R channel.

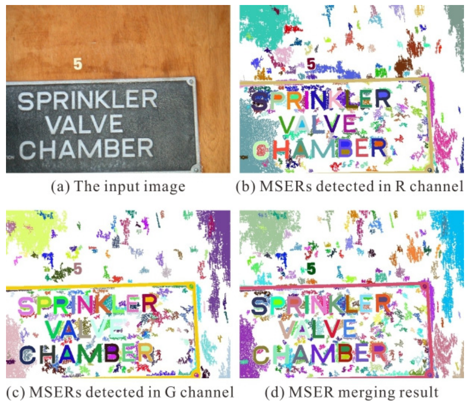
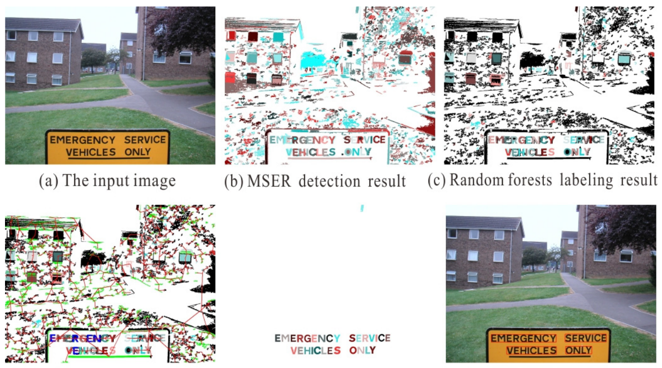


Fig. 2. Example of MSER merging result



(a) The input image (b) MSER detection result (c) Random forests labeling result

(d) CRF model (e) Final labeling result (f) Text detection result

Fig. 3. Details of the proposed text detection method.

B. *Nontext Filtering Using Random Forests Classifier*

According to our statistics, only 6.1% of the resulting MSERs corresponds to character. Picking up text components from such an unbalanced database is a challenging task. So we use a two-stage nontext filtering strategy to address this issue. In the first stage, a random forests classifier [10] is employed to filter out most of the nontext components. Random forests classifier is widely used in text detection due to its fast speed and relatively better generalization performance. We detect the following component features which are then fed into the classifier:

- **Regularity.** Regularity is defined as the ratio of the pixel numbers between skeleton and contour. Since the relative regular structure of characters should not be too complex or too random, the regularity can distinguish characters from non-characters who have too many burrs, twists and turns.
- **Aspect ratio.** This feature is defined as the ratio of component's width and height. Characters tend to have aspect ratio within a certain range.
- **Occupation ratio.** This feature, defined as the ratio between component pixel number and its bounding box area, is expected to exclude components occupying too many or too few pixels in the bounding box.
- **Compactness.** Compactness is defined as the ratio between the square of component's perimeter and its bounding box area. The purpose is to remove components with too complex contour shapes.
- **Stroke width variance.** Generally, pixels in the same text component have uniform stroke widths.
- **Euler number.** Euler number represents the difference between the number of connected components and the number of holes. It is a topological feature of a binary image and can be calculated by a very efficient yet simple algorithm introduced in [11].

With the above processing by the random forests classifier, our experiments found that 85.47% of the nontext components are eliminated. The labeling result can be seen from Fig. 3(c), where components drawn with black color are nontext while others are text.

C. *Component Relabeling with CRF Model*

The wrongly labeled nontext components by the random forests usually have higher similarity with characters. To exclude them, we model the binary contextual component relationship into a CRF framework. The efficient probabilistic graphical model is proposed by Schmidt et al. [12].

1) Graph Model Construction

To incorporate different MSERs into a framework, we construct them into an undirected graph model $G=\{V,E\}$, which is composed of nodes V and undirected edges E. Each component corresponds to a node of the graph and undirected edges are built to link neighboring nodes. Firstly, we initialize edge set $E=\varnothing$ and $E'=\varnothing$. Edges satisfying the following criterions are added into E':

$$\begin{aligned} & s_1=\frac{\sqrt{(x_i-x_j)^2+(y_i-y_j)^2}}{\min(\max(h_i,w_i),\max(h_j,w_j))}<2 \\ \wedge\ & s_2=\frac{\min(w_i,w_j)}{\max(w_i,w_j)}>0.4 \\ \wedge\ & s_3=\frac{\min(h_i,h_j)}{\max(h_i,h_j)}>0.4 \\ \wedge\ & s_4=\left|g_i-g_j\right|<50 \end{aligned} \tag{1}$$

where (x_i,y_i) is the center position of node i, w_i and h_i are the height and width, respectively, and g_i is the average gray level of component pixels. The weight of edges is defined as:

$$w=\frac{s_1}{2}+|1-s_2|+|1-s_3|+\frac{s_4}{50} \tag{2}$$

We sort edges in E' according to their priorities (the smaller the weight, the higher the priority), and process them orderly (from high priority to low priority) according to the following rules: 1) If the nodes linked by the current edge belong to the same tree, we ignore the current edge; 2) If the nodes linked by the current edge belong to two different trees, we merge the two trees by adding the current edge into E. Different from other undirected graph models with loops ([8], [9]), our graph model is a set of minimum spanning trees.

2) The CRF Model

The component labeling problem can be described as follows: given a set of observation variables $\boldsymbol{x}=\{x_i\}$, we need to find the best label set $\boldsymbol{y}=\{\mathrm{y_i}\}$, where $\mathrm{y_i}\in\{1,2\}$. We consider our CRF with pair-wise potentials:

$$p(\boldsymbol{y}\mid\boldsymbol{x})=\frac{1}{z(\boldsymbol{x})}\prod_{<i,j>}\varphi_{ij}(y_i,y_j,\boldsymbol{x})\prod_i\varphi_i(y_i,\boldsymbol{x}) \tag{3}$$

where $<i,j>$ is a product over all edge. φ_i and φ_{ij} are node potential and edge potential with the following form:

$$\varphi_i(\cdot,\boldsymbol{x})=\left(e^{v_1^T x_i},e^{v_2^T x_i}\right),\quad \varphi_{ij}(\cdot,\cdot,\boldsymbol{x})=\begin{pmatrix} e^{w_{11}^T x_{ij}}, & e^{w_{12}^T x_{ij}} \\ e^{w_{21}^T x_{ij}}, & e^{w_{22}^T x_{ij}} \end{pmatrix} \tag{4}$$

where $x_i=[1,f_i]$ is a set of node features and $x_{ij}=[1,f_{ij}]$ is a set of edge features. The element "1" in x_i and x_{ij} is a single "bias" feature for node potential and edge potential. The bias feature is used to reflect any effects on the states that are independent of the features. Besides that, when the states are not balanced in the training data, a bias feature also makes sense. We set $w_{12} = w_{21}$ to ensure the identifiability, otherwise the model would be over-parameterized. If we rewrite $\theta = [v, w]$ for all the parameters and $F(\boldsymbol{x}, \boldsymbol{y})$ for all features, we can write the model more succinctly as $p(\boldsymbol{y} \mid \boldsymbol{x}) = \frac{\exp(\theta^T F(\boldsymbol{x}, \boldsymbol{y}))}{z(\theta, \boldsymbol{x})}$ where $z(\theta, \boldsymbol{x}) = \sum_{y'} \exp(\theta^T F(\boldsymbol{x}, \boldsymbol{y}'))$. The negative log-likelihood and gradient considering L_2 regularization are now given by:

$$nll(\theta) = \sum_{n=1}^{N} -\theta^T F(x_n, y_n) + \sum_{n=1}^{N} \log z(\theta, x_n) + \sum_{k=1}^{K} \frac{\theta_k^2}{2\sigma^2} \tag{5}$$

$$\nabla nll(\theta) = -\sum_{n} [F(x_n, y_n) - E_{y'} F(x_n, \boldsymbol{y}')] + \frac{\theta}{\sigma^2} \tag{6}$$

where $E_{y'} F(x_n, \boldsymbol{y}') = \sum_{y'} p(\boldsymbol{y}' \mid x_n, \theta) F(x_n, \boldsymbol{y}')$ are the expectations for the features.

We do conditional inference with sum-product belief propagation algorithm when computing the marginal probability. Nodes are observed according to the random forests classification results. A node will be observed as nontext if it is classified as nontext with a probability greater than a threshold α . And a node will be observed as text if it is classified as text with a probability greater than a threshold β . A 36-D HOG based verification classifier, which indicates the confidence of components to be text, is carried out to prevent text components from being wrongly classified by the CRF. A component labeled as nontext by the CRF will be discarded only when it satisfies $p(text) < \gamma$, where $p(text)$ is the output of the verification classifier, and γ is a threshold. The CRF graph model is demonstrated in Fig. 3(d), wherein red lines represent edges, green components are observed nontext nodes, and blue ones are observed text nodes. The final filtering result is shown in Fig. 3(e).

3) *Features Used in CRF*

The node features used in CRF are the same as the ones utilized in the random forests classifier, and binary features for the edge linking node i and node j are defined as follows:

- **Color difference**. Characters belonging to the same word usually share similar component pixel colors and background pixel colors. The similarity can be measured as follows:

$$F_1 = \frac{\sqrt{\sum_{k=1,2,3} (CIn_{ki} - CIn_{kj})^2}}{255}, \; F_2 = \frac{\sqrt{\sum_{k=1,2,3} (COut_{ki} - COut_{kj})^2}}{255} \tag{7}$$

where CIn_{ki} and $COut_{ki}$ denote the average component pixel color and background pixel color in k^{th} channel.

- **Spatial distance.** This feature, defined by (8), reflects the spatial distance of two components.

$$F_3 = \frac{\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}}{\min(w_i, w_j)} \tag{8}$$

- **Gray level difference**. We consider about the gray level difference both in component pixels and background pixels as follows:

$$F_4 = \frac{\max(\phi_{Ii}, \phi_{Ij})}{2\min(\phi_{Ii}, \phi_{Ij})} + \frac{\max(\phi_{Oi}, \phi_{Oj})}{2\min(\phi_{Oi}, \phi_{Oj})},$$
$$F_5 = \frac{\max(\mu_{Ii}, \mu_{Ij})}{2\min(\mu_{Ii}, \mu_{Ij})} + \frac{\max(\mu_{Oi}, \mu_{Oj})}{2\min(\mu_{Oi}, \mu_{Oj})} \tag{9}$$

where ϕ_{Ii} and μ_{Ii} denote the gray level variance and mean of component pixels while ϕ_{Oi} and μ_{Oi} denote the gray level variance and mean of background pixels.

- **Shape difference.** Shape difference is used to measure the shape similarity of two components.

$$F_6 = 0.4 \times \frac{|w_i - w_j|}{\min(w_i, w_j)} + 0.6 \times \frac{|h_i - h_j|}{\min(h_i, h_j)} \tag{10}$$

- **Stroke width difference**. Characters belonging to the same word usually share similar stroke width.

$$F_7 = \frac{\max(\overline{SW_i}, \overline{SW_j})}{\min(\overline{SW_i}, \overline{SW_j})} \tag{11}$$

where $\overline{SW_i}$ is the average stroke wide of component i .

D. *Word Grouping and False Positives Elimination*

We have observed that: 1) Characters belonging to the same word usually share similar height, and width; 2) The within-word distance between two adjacent characters should be smaller than between-word distance; 3) If the number of characters in a word is greater than three, the start and the end characters have only one closest neighbor while others have two closest neighbors. Inspired by these observations, we propose an edge-cut strategy to group text components into words. Firstly, we find the two closest neighbors who satisfy (12) for each component. Thus we will get graphs by linking each component with their closest neighbors. Then we cut off edges who do not satisfy $D_{ij} > \tau \overline{D}$ when the number of components in the graph is greater than three, where D_{ij} is the gap distance between two components linked by the edge, $\overline{D}$ is the average gap distance of the graph, and τ is a threshold. If there are only two components in a graph, we cut off the edge linking them when the gap distance satisfies $D_{ij} > 3\max(w_i, w_j)$. Finally, we get some sub-graphs, each of which corresponds to a word.

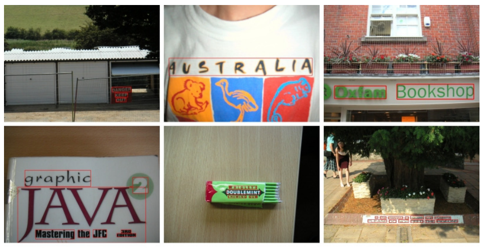
$$\arctan\left(\frac{y_i - y_j}{x_i - x_j}\right) < \pi/6 \wedge |y_i - y_j| < \min(h_i, h_j) \tag{12}$$

The nontext filtering process can remove most of the nontext components, but there are still some false positives. Shi's [8] elimination method is utilized to remove these false positives. Concretely, we normalize each text block with a height of 24pixels, and classifier is trained to classify sub-images of size 24×24 scanned with steps of 12 pixels from the normalized text block. But different from Shi's method, we define the confidence of the whole text block as

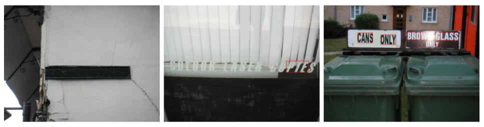
$conf(r) = \frac{1}{l}\sum_{i=1}^{l} F_i$, where l is the number of sub-images and F_i is the output of the classifier. For sub-image classified as text, we set $F_i = 1$, otherwise, $F_i = 0$. Finally, the text block r is preserved if $conf(r) > \kappa$. The random forests classifier combing with 8-D HOG is employed as the sub-image classifier. Fig. 3(f) shows the final text detecting result of Fig. 3(a).

III. EXPERIMENTS

We evaluate the proposed method on the benchmark ICDAR2013 Robust Reading Competition dataset and compare the performance with several state-of-the-art methods reported in [13]. To evaluate the performance of our proposed method and fairly compare with others, the DetEval [14] evaluation software is utilized with the same parameter setting as in the competition. The result is shown in TABLE I. As we can see, our method achieves the highest recall and f-measure though the precision is lower than some other methods. Higher recall implies more text is detected from the images. The strategy of merging multi-channel MSERs leads to more text being detected, which contributes to a higher recall. However, more nontext components are extracted at the same time, which makes the nontext filtering more difficult. More examples of our text detection result can be seen from Fig. 4(a). Our method fails to detect text in images with low contrast, or be affected by strong highlight as shown in Fig. 4(b).



(a) Successful text detection samples



(b) Failed text detection samples

Fig. 4. Text detection samples of the proposed method

TABLE I. EXPERIMENTE RESULT ON THE ICDAR2013 DATASET

	Recall(%)	Precision(%)	F(%)
USTB_TexStar	66.45	**88.47**	75.89
Text_Spotter	64.84	87.51	74.49
CASIA_NLPR	68.24	78.89	73.18
Text_Detector_CASIA	62.85	84.70	72.16
I2R_NUS_FAR	69.00	75.08	71.91
I2R_NUS	66.17	72.54	69.21
Proposed method	**73.86**	80.34	**76.96**

IV. CONCLUSIONS

This paper proposed a two-stage nontext filtering-based text detection method in scene images. MSER detector was used to extract character candidates. To detect more characters and reduce the repeat components, we detected MSERs in multi-channels and merged them together before further process. Then a two-stage labeling strategy combining random forests with CRF was utilized to filter out nontext components. Text components were grouped into words, and false positives were eliminated finally. The proposed method achieved the state-of-art in dataset ICDAR2013 compared with the existing methods.

ACKNOWLEDGMENTS

This work is jointly supported by the Science and - Technology Commission of Shanghai Municipality under research grants 14511105500, 14DZ2260800, and Shanghai Collaborative Innovation Center of Trustworthy Software for Internet of Things (ZF1213).

REFERENCE

[1] S. Gao, C. Wang, B. Xiao et al., “Adaptive scene text detection based on transferring adaboost,” in 12th Intenational Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 388-392.

[2] L. Neumann, J. Matas, “Scene text localization and recognition with oriented stroke detection,” in IEEE International Conference on Computer Vision (ICCV), 2013, pp. 97-104.

[3] B. Bai, F. Yin, C.L. Liu, “Scene text localization using gradient local correlation,” in 12th Intenational Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 1412-1416.

[4] B. Epshtein, E. Ofek, Y, Wexler, “Detecting text in natural scenes with stroke width transform,” in IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 2963-2970.

[5] L. Neumann, J. Matas,”Real-time scene text localization and recognition,” in IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 3538-3545.

[6] X.C. Yin, X. Yin, K. Huang, H. Hao, “Robust text detection in natural scene images, ” in IEEE Transaction on Pattern Analysis and Machine Intelligence, 36(5): 970-983, 2014.

[7] H. Koo, D. Kim, “Scene text detection via connected component clustering and nontext filtering,” in IEEE Transaction on Image Processing. 22(6): 2296-2305, 2013.

[8] C. Shi, C. Wang, B. Xiao, Y. Zhang, S. Gao, “Scene text detection using graph model built upon maximally stable extremal regions,” in Pattern Recognition Letters. 34(2): 107-116, 2013.

[9] Y. F. Pan, X Hou, C.L. Liu, “A hybrid approach to detect and localize texts in natural scene images,” in IEEE Transaction on Image Processing, 20(3):800-813, 2011.

[10] L. Breiman, “Random Forests,” Machine Learning, 45(1): 5-32, 2001.

[11] W. K. Pratt. Digital Image Processing: PIKS Inside. John Wiley & Sons, Inc., New York, NY, USA, 3rd edition, 2001.

[12] M. Schmidt, K. Murphy, G. Fung, R. Rosales, “Structure learning in random fields for heart motion abnormality detection,” in IEEE Conference on Computer Vision and Pattern Recognition, 2008, pp. 1-8.

[13] D. Karatzas, F. Shafait, S. Uchida, et al. “ICDAR 2013 robust reading competition, ” in 12th International Conference on Document Analysis and Recognition (ICDAR), 2013, pp. 1484-1493.

[14] http://liris.cnrs.fr/christian.wolf/software/deteval/index.html.