Scene Text Detection via Connected Component Clustering and Nontext Filtering

Hyung Il Koo, *Member, IEEE*, and Duck Hoon Kim, *Member, IEEE*

***Abstract*— In this paper, we present a new scene text detection algorithm based on two machine learning classifiers: one allows us to generate candidate word regions and the other filters out nontext ones. To be precise, we extract connected components (CCs) in images by using the maximally stable extremal region algorithm. These extracted CCs are partitioned into clusters so that we can generate candidate regions. Unlike conventional methods relying on heuristic rules in clustering, we train an AdaBoost classifier that determines the adjacency relationship and cluster CCs by using their pairwise relations. Then we normalize candidate word regions and determine whether each region contains text or not. Since the scale, skew, and color of each candidate can be estimated from CCs, we develop a text/nontext classifier for normalized images. This classifier is based on multilayer perceptrons and we can control recall and precision rates with a single free parameter. Finally, we extend our approach to exploit multichannel information. Experimental results on ICDAR 2005 and 2011 robust reading competition datasets show that our method yields the state-of-the-art performance both in speed and accuracy.**

***Index Terms*— Connected component (CC)-based approach, CC clustering, machine learning classifier, nontext filtering, scene text detection.**

I. Introduction

SINCE mobile devices equipped with high-resolution digital cameras are widely available, research activities using these devices in the field of human computer interaction (HCI) have received much attention for the last decades. Among them, text detection and recognition in camera-captured images have been considered as very important problems in computer vision community [1]–[3]. It is because text information is easily recognized by machines and can be used in a variety of applications. Some examples are aids for visually impaired people, translators for tourists, information retrieval systems in indoor and outdoor environments, and automatic robot navigation. Although there exist a lot of research activities in this field, scene text detection is still remained as a challenging problem. This is because scene text images usually suffer from photometric degradations as well as geometrical distortions so that many algorithms faced the accuracy and/or speed (complexity) issues [4]–[6].

Manuscript received August 22, 2012; revised November 29, 2012; accepted February 4, 2013. Date of publication February 26, 2013; date of current version April 12, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mary Comer.

H. I. Koo is with the Division of Electrical and Computer Engineering, Ajou University, Suwon 443-749, Korea (e-mail: hikoo@ajou.ac.kr).

D. H. Kim is with Qualcomm Research Korea, Seoul 135-010, Korea (e-mail: duckhoon@qti.qualcomm.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2013.2249082

A. Related Work

Most of the scene text detection algorithms in the literature can be classified into region-based and connected component (CC)-based approaches [1]–[3]. Region-based methods [7], [8] adopted a sliding window scheme, which is basically a brute-force approach which requires a lot of local decisions. Therefore, the region-based methods have focused on an efficient binary classification (text versus nontext) of a small image patch. In other words, they have focused on the following problem:

1) Problem (A): to determine whether a given patch is a part of a text region.

For efficient classification, researchers addressed this problem by adopting cascade structures. In their approaches, simple features such as horizontal and vertical derivatives were used at the early stages of the cascade and complex features were incrementally employed [7], [8]. Even though this structure enables efficient text detection, Problem-(A) is still challenging. It is not straightforward even for human to determine the class of a small image patch when we do not have knowledge of text properties such as scale, skew, and color. Multi-scale scheme using different window sizes can alleviate the scale issues, however, it makes boxes in different scales overlap. Experimental results on ICDAR 2005 dataset have shown that this region based approach is efficient, however, it yields worse performance compared with CC-based approaches [5], [9], [10].

CC-based methods begin with CC extraction and localize text regions by processing only CC-level information. Therefore, they have focused on the following problems:

1) Problem (B): to extract text-like CCs.
2) Problem (C): to filter out nontext CCs.
3) Problem (D): to infer text blocks from CCs.

In the literature, many CC extraction methods were developed to address Problem (B). For example, some methods assumed that the boundaries of text components should show strong discontinuities and they extracted CCs from edge maps. Others were inspired by the observation that text is written in the same color and they applied color segmentation (or reduction) techniques [11]. On the other hand, some researchers developed their own CC extraction methods from the scratch: the curvilinearity of text was exploited in [10], [12] and local binarization by using estimated scales was adopted in [9].

After the CC extraction, CC-based approaches filter out nontext CCs. In the end, features such as "aspect ratio," "the number of holes in a CC," and "the variance of the stroke

1057-7149/$31.00 © 2013 IEEE

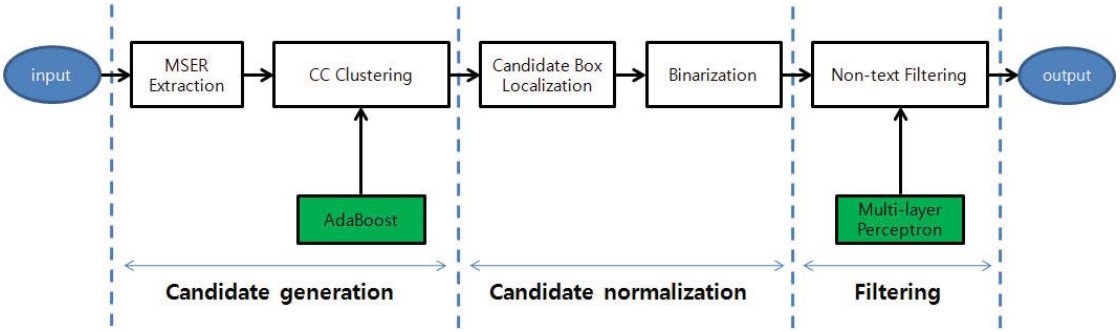


Fig. 1. Block diagram of our system.

width within each CC" were employed in [10], [13]. In [9], conditional random fields (CRFs) were adopted in order to consider binary (relational) features as well as unary features. In [14], a neural network was used to filter out nontext components.

Finally, CC-based approaches infer text blocks from the remaining CCs. This step is also known as *text line aggregation*, *text line formation*, or *text line grouping*. Interestingly, many methods were based on similar rules. For example, the height ratio between two letters and color difference have been used in a number of methods [10]–[13]. Although CC-based approaches have shown better performance than region-based ones, they usually suffer from the computational complexity. It is because their performances depend on the quality of CCs and they adopted sophisticated CC extraction and filtering methods [9], [10], [14], [15].

B. Our Approach

We have illustrated the block diagram of our system in Fig. 1. As shown in the figure, our method consists of three steps: candidate generation, candidate normalization, and nontext filtering. Our candidate generation step is based on popular CC-based approaches, however, we have focussed on Problem-(D) rather than Problem-(B) and (C). That is, we use an efficient CC extraction method [16] and we do not adopt CC filtering ideas which are usually time-consuming. Rather, we address the problems caused by the absence of low level CC processing with the ideas in region-based approaches, i.e., we address Problem-(A). We can normalize candidate regions with CC-level information and this normalization allows us to build a simple but reliable text/nontext classifier.

In our approach, both problems ((D) and (A)) are addressed based on machine learning techniques, so that our method is largely free from heuristics. We have trained a classifier that determines adjacency relationship between CCs for Problem-(D) and we generate candidates by identifying adjacent pairs. In training, we have selected efficient features and trained the classifier with the AdaBoost algorithm [17]. As mentioned, we apply the above CC clustering method to raw CC sets (that usually contains many nontext CCs) and some candidates may correspond to "nontext clusters." Therefore, nontext rejection scheme should be followed, which is the main problem of region-based methods. However, our situation is different from conventional ones because we can use normalized inputs in classification. We estimate the skew and scale of a candidate from the distribution of CCs, and estimate text and background colors from the CCs. Examples of our normalized results are shown in Fig. 8(f). Finally, we reject nontexts among normalized candidates with a multilayer perceptron that is trained with the back-propagation algorithm [18], [19].

Our CC extraction algorithm is the maximally stable extremal region (MSER) algorithm that is invariant to scales and affine intensity changes, and other blocks in our method are also designed to be invariant to these changes. These invariance allows us to exploit multi-channel information: we can apply our method to multiple channels at the same time and treat their outputs as if they are from a single source. In this way, we are able to detect text that are not salient in luminance channel images.

We submitted our preliminary results to 2011 ICDAR robust reading competition [6] and won the first prize. Experimental results on 2005 competition results have also shown that our method yields better performance with small computational complexity. The rest of this paper is organized as follows. In Section II, we explain the way of adding more information to ground truth and explain frequently used notations in this paper. We present our candidate generation method in Section III, which consists of a MSER-based CC extraction block and an AdaBoost-based CC clustering block. In Section IV and V, we present our normalization method and nontext rejection algorithm respectively. Note that normalization in our paper means not only geometric normalization but also binarization (color normalization). Finally, we show experimental results in Section VI, and conclude the paper in Section VII.

II. Augmented Ground Truth and Frequently Used Notations

A. Construction of New Ground Truth

The ICDAR dataset consists of natural images annotated with bounding boxes around each instance of a word [4]–[6]. Although it contains sufficient information in performance evaluation, we need more information for the training of our classifiers. As shown in Fig. 1, we use a classifier that tells us the adjacency relation between CCs in the candidate generation step, and CC-level information is essential for the training of such a classifier (which will be clarified in

Fig. 2. (a) Input image. (b) Illustration of our ground truth. We have pixel-level annotations (binarization results) and text-line numbers. (c) MSER results of (a). We assigned random colors to CCs for better visibility. Note that many CCs are nested due to the properties of stable regions.

Section III-B). Therefore, we augmented ground truth that was released as 2011 ICDAR training set by adding pixel-level annotations (binarization results) and text-line information. For example, in case of Fig. 2(a), there was bounding box information of four words in original ground truth. In addition to them, we build binarization results as shown in Fig. 2(b) and assign a text-line number for each CC. That is, we assign "1" to CCs in a box containing "SUMMER," assign "2" to CCs in a box containing "WHATEVER" and "THE," and assign "3" to CCs in a box containing "WEATHER."

B. Frequently Used Notations

We say that two CCs are adjacent when both are text components in the same word and the number of characters between them is less than 2. For a word "WHATEVER," we say that "W is adjacent to H" and "W is adjacent to A," however, we say that "W is not adjacent to T" because there are two characters (i.e., "H" and "A") between them. We also use a notation

$$c_i \sim c_j \tag{1}$$

when c_i and c_j are adjacent, and

$$c_i \nsim c_j \tag{2}$$

otherwise. Let $t(c_i)$ denote the text-line number of c_i. Note that $c_i \sim c_j$ means that $t(c_i) = t(c_j)$, however, $t(c_i) = t(c_j)$ does not necessarily mean $c_i \sim c_j$. In addition, $\mu_i \in \Re^3$ indicates the mean color of pixels in c_i, and

$$s_i = |c_i| \tag{3}$$

where $|\cdot|$ indicates the number pixels in a given CC.

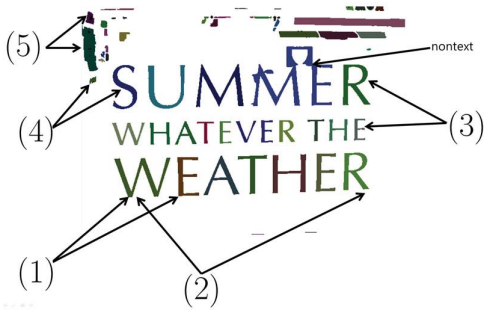


Fig. 3. Illustration of possible relations between CCs. For clarity, we have only illustrated some CCs in $\mathcal{C}$.

Let us assume that we apply a CC extraction method (the MSER algorithm in our case) to an image and get a set of CCs as illustrated in Fig. 2(c). Fig. 3 shows some CCs excerpted from Fig. 2(c) for clarity. We denote the set of CCs as $\mathcal{C}$, and partition this set into a text component set $\mathcal{T}$ and a nontext component set $\mathcal{N}$:

$$\mathcal{T} \cup \mathcal{N} = \mathcal{C} \tag{4}$$

$$\mathcal{T} \cap \mathcal{N} = \emptyset. \tag{5}$$

Here we consider an element in $\mathcal{C}$ as a text component when there is a corresponding CC in the ground truth binary image. In other words, a CC in Fig. 2(c) is considered as an element of $\mathcal{T}$ if there exists a corresponding CC in Fig. 2(b). However, it is very unlikely that $c \in \mathcal{C}$ is identical to a certain CC in the ground truth binary image, and we relax this condition. That is, $c \in \mathcal{T}$ means that there exists a CC, i.e., e, in the ground truth binary image satisfying

$$|c \cap e| \geq 0.8 \times |c| \tag{6}$$

$$|c \cap e| \geq 0.8 \times |e|. \tag{7}$$

For example, a CC "S" in Fig. 3 is considered as a text component because there is a similar CC in Fig. 2(b). On the other hand, a CC consisting of "M" and "E" in Fig. 3 is different from any CCs in Fig. 2(b) and it is considered as a nontext component.

III. Candidate Generation

For the generation of candidates, we extract CCs in images and partition the extracted CCs into clusters, where our clustering algorithm is based on an adjacency relation classifier. In this section, we first explain our CC extraction method. Then, we will explain our approaches (i) to build training samples, (ii) to train the classifier, and (iii) to use that classifier in our CC clustering method.

A. CC Extraction

Among a number of CC extraction methods, we have adopted the MSER algorithm because it shows good performance with a small computation cost [16], [20]. This algorithm can be considered as a process to find local binarization results that are stable over a range of thresholds, and this

property allows us to find most of the text components [14], [15]. The MSER algorithm yields CCs that are either darker or brighter than their surroundings. In Fig. 2(c), we have illustrated brighter CCs by assigning random colors to them. Note that many CCs are overlapping due to the properties of stable regions [16]. More MSER extraction examples can be found in Fig. 8(a) and (b), which show brighter and darker CCs, respectively.

B. Building Training Sets

Our classifier is based on pairwise relations between CCs, and let us first consider cases that can happen for a CC pair $(c_i, c_j) \in \mathcal{C} \times \mathcal{C}$ $(i \neq j)$:

1) $c_i \in \mathcal{T}$, $c_j \in \mathcal{T}$, $c_i \sim c_j$
2) $c_i \in \mathcal{T}$, $c_j \in \mathcal{T}$, $c_i \nsim c_j$, $t(c_i) = t(c_j)$
3) $c_i \in \mathcal{T}$, $c_j \in \mathcal{T}$, $c_i \nsim c_j$, $t(c_i) \neq t(c_j)$
4) $c_i \in \mathcal{T}$, $c_j \in \mathcal{N}$
5) $c_i \in \mathcal{N}$, $c_j \in \mathcal{N}$.

We have illustrated them in Fig. 3. In a strict sense, we may have to classify the case (1) from the (2) $\sim$ (5) cases (of course, this classification will allow us to find all words having more than one character). However, it is not straightforward to train such a classifier. For example, let us consider "R" and "T" in the second line in Fig. 2(b). It is not straightforward to determine whether they are in the same word without considering other characters. Therefore, rather than focusing on this difficult problem, we address a relatively simple problem by adopting an idea in region-based approaches. That is, we adopt a nontext filtering block as shown in Fig. 1, and we are no longer required to care about the case (5). If we have $c_i \sim c_j$ for some $c_i, c_j \in \mathcal{N}$, it will yield a candidate consisting of nontext CCs and this candidate will be rejected at the nontext rejection step. Also, we will perform word segmentation as a postprocessing step and the case (2) does not mean negative samples. Based on these observations, we build training sets. Specifically, we first obtained sets of CCs by applying the MSER algorithm to a training set released by [6]. Then, for every pair $(c_i, c_j) \in \mathcal{C} \times \mathcal{C}$ $(i \neq j)$, we identify its category among 5 cases. A positive set is built by gathering samples corresponding to the case (1) and a negative set by gathering samples corresponding to the case (3) or (4). Samples from other cases were discarded. Note that this process can be automated by using our augmented ground truth in Section II-A.

C. AdaBoost Learning

With the collected samples, we train an AdaBoost classifier that tells us whether $(c_i, c_j) \in \mathcal{C} \times \mathcal{C}$ $(i \neq j)$ is adjacent or not. For the presentation of our method, let us define some local properties of CCs. Given $c_i \in \mathcal{C}$, we find a bounding box of c_i and denote its width and height as w_i and h_i respectively. Given a pair $(c_i, c_j) \in \mathcal{C} \times \mathcal{C}$, the horizontal distance, horizontal overlap, and vertical overlap between two boxes are denoted as d_{ij}, ho_{ij}, and vo_{ij} respectively. We have illustrated some of them in Fig. 4.

We have used 6-dimensional feature vectors consisting of five geometrical features and one color-based feature. All of

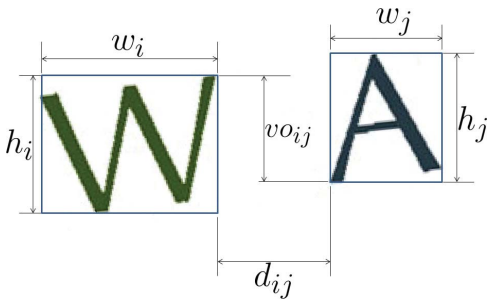


Fig. 4. Illustration of local properties between two CCs.

geometric features are designed to be invariant to the scale of an input image:

$$\frac{\max(h_i, h_j)}{\min(h_i, h_j)}, \frac{\max(s_i, s_j)}{\min(s_i, s_j)} \tag{8}$$

$$\frac{ho_{ij}}{\min(w_i, w_j)}, \frac{vo_{ij}}{\min(h_i, h_j)} \tag{9}$$

$$\frac{|h_i - h_j|}{\min(h_i, h_j)} \times \frac{d_{ij}}{\min(w_i, w_j)} \tag{10}$$

and the color feature is given by the color distance between two CCs in RGB space:

$$\|\mu_i - \mu_j\|_2. \tag{11}$$

Features in (8) reflect the relative scales (refer to (3) for the definition of s_i and s_j) between two CCs and features in (9) encode their relative locations. The scalar in (10) is the product of normalized height difference and normalized distance, which will be large when c_i and c_j are not adjacent. All of these features are informative and we consider each feature as a weak classifier. For example, the heights of adjacent English characters are similar, and $\frac{1}{2} \leq \frac{h_i}{h_j} \leq 2$ means that it is likely that c_i and c_j are adjacent. From these weak classifiers, we build a strong classifier with the Gentle AdaBoost learning algorithm [17], [21]. The Gentle AdaBoost is a variant of AdaBoost learning, and it is easy to implement and known to show good performance in many applications [17], [22].

D. CC Clustering

The AdaBoost algorithm yields a function

$$\phi : \mathcal{C} \times \mathcal{C} \to \Re \tag{12}$$

and we use this function in binary decisions:

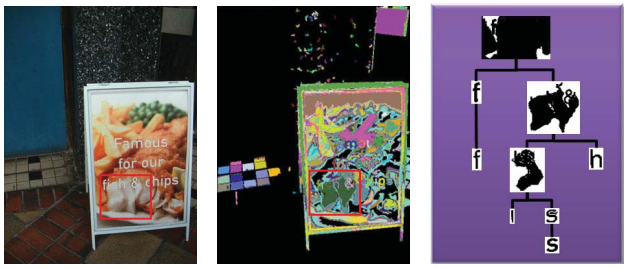
$$\phi(c_i, c_j) > \tau_1 \Longleftrightarrow c_i \sim c_j \tag{13}$$

with a threshold τ_1. Given $\phi(\cdot, \cdot)$ and τ_1, we can find all adjacent pairs by evaluating that function for all possible pairs in $\mathcal{C}$. Based on these adjacency relations, $\mathcal{C}$ is partitioned into a set of clusters

$$\mathcal{W} = \{w_k\} \tag{14}$$

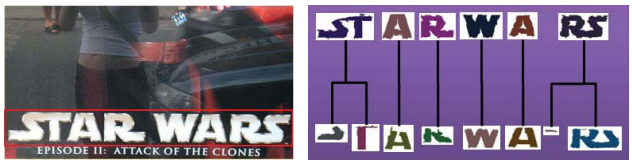
where $w_k \subset \mathcal{C}$. Formally speaking, $c_i, c_j \in w_k$ (i.e., c_i and c_j are in the same cluster) means that there exists $\{e_i\}_{i=1}^m \subset \mathcal{C}$ such that

$$c_i \sim e_1 \sim e_2 \sim \cdots e_m \sim c_j. \tag{15}$$



(a) (b) (c)

Fig. 5. (a) Input image. (b) Color-coded MSER results. (c) Hierarchical tree of MSER results in the red box in (b).



(a) (b)

Fig. 6. (a) Input image. (b) Hierarchical tree of MSER results in the red box in (a).

We build $\mathcal{W}$ by using the union-find algorithm [23]. After clustering, we have discarded clusters having only one CC.

E. Comparison to Other MSER-Based Methods

The MSER algorithm has desirable properties for text detection: (i) detection results are invariant to affine transformation of image intensities and (ii) no smoothing operation is involved so that both very fine and very large structures can be detected at the same time [16]. Therefore, the algorithm has been adopted in many methods [13]–[15]. However, unlike our approach, they focused on the retrieval of CCs corresponding to individual characters: the authors in [13] developed a variant of MSER in order to prevent the merging of individual characters, and a Support Vector Machine (SVM) based classifier was developed for the character and non-character classification in [15]. That is, they tried to develop MSER-based CC-extractors yielding individual characters (i.e., high precision and high recall).

On the other hand, we mainly focus on retrieving the text components as much as possible. As a result, redundant and noisy CCs could be involved in finding clusters. As shown in Figs. 5(c) and 6(b), due to the characteristics of the MSER algorithm, some characters are detected more than once and there are lots of nontext components. Moreover, some of them do not correspond to individual characters (e.g., "ST" and "RS" in Fig. 6). The advantages of our approach are its efficiency and robustness. Our method can be efficiently implemented because CC-level feature extraction and classification are not involved. We can also deal with the variations of characters (caused by the font variations and blurs) because we do not exploit the features of individual characters (our algorithm successfully detects texts in Fig. 6(a)). This approach has drawbacks that text regions could be overlapping and nontext regions are sometimes detected, which will be addressed in the following sections.

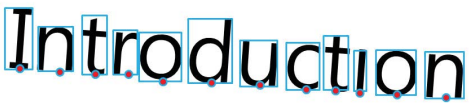


Fig. 7. We denote points in B_k as red. By computing the angles connecting two points in B_k, we can estimate the skew of a word.

IV. Candidate Normalization

After CC clustering, we have a set of clusters. In this section, we normalize corresponding regions for the reliable text/nontext classification.

A. Geometric Normalization

Given $w_k \in \mathcal{W}$, we first localize its corresponding region. Even though text boxes can experience perspective distortions, we approximate the shape of text boxes with parallelograms whose left and right sides are parallel to y-axis. This approximation alleviates difficulties in estimating text boxes having a high degree of freedom (DOF): we only have to find a skew and four boundary supporting points. To estimate the skew of a given word candidate w_k, we build two sets:

$$T_k = \{t(c_i) | c_i \in w_k\} \tag{16}$$

$$B_k = \{b(c_i) | c_i \in w_k\} \tag{17}$$

where $t(c_i)$ and $b(c_i)$ are the top-center point and the bottom-center point of a bounding box of c_i, respectively. We illustrate B_k in Fig. 7. For every pair in B_k and T_k, the slope of a line connecting the pair is discretized into one of 32 levels in $[-\frac{\pi}{8}, \frac{\pi}{8}]$, and each pair votes for the skew angle. After voting, the most common angle is considered as a skew. Localized blocks are shown in Fig. 8(c). Then, we perform geometric normalization by applying an affine mapping that transforms the corresponding region to a rectangle. During the transformation, we use a constant target height (48 pixels in experiments) and preserve the aspect ratio of the box. Geometrically normalized results of localized blocks in Fig. 8(c) are shown in Fig. 8(e).

B. Binarization

Given geometrically normalized images, we build binary images. In many cases, MSER results can be considered as binarization results as shown in Fig. 5(c). However, we perform the binarization separately by estimating text and background colors. It is because (i) the MSER results may miss some character components and/or yield noisy regions (mainly due to the blur) and (ii) we have to store the point information of all CCs for the MSER-based binarization. We consider the average color of CCs as the text color:

$$\frac{\sum_{c_i \in w_k} s_i \mu_i}{\sum_{c_i \in w_k} s_i} \in \Re^3 \tag{18}$$

and consider the average color of an entire block as the background color. Then, we obtain a binary value of each pixel by comparing the distances to the estimated text color and the estimated background color. We have used l_2 norm in RGB space. The binarization results of Fig. 8(e) are shown in Fig. 8(f).

Fig. 8. (a) and (b) Brighter and darker CCs extracted by the MSER algorithm [16]. (c) We localize candidate regions in image domains. (d) Nontext blocks are filtered out by exploiting the statistical properties of the regions (our final results). (e) Geometrically normalized results of localized boxes. (f) Our normalization results.

V. Text/Nontext Classification

In order to get final results like Fig. 8(d) from Fig. 8(c) and (f), we develop a text/nontext classifier that rejects nontext blocks among normalized images. In our classification, we do not adopt sophisticated techniques such as cascade structures, since the number of samples to be classified is usually small. However, one challenge for our approach is the variable aspect ratio as shown in Fig. 8(f). One possible approach to this problem is to split the normalized images into patches covering one of the letters and develop a character/non-character classifier as [15]. However, character segmentation is not an easy problem [24] and there are examples where this approach may fail (See Fig. 6). Rather, we split a normalized block into overlapping squares as illustrated in Fig. 9(a), and develop a classifier that assigns a textness value to each square block. Finally, decision results for all square blocks (in the right hand side of Fig. 9(a)) are integrated so that the original block (in the left hand side) is classified. In this section, we first present our training method that allows us to have a textness value for each square. Then, we explain our text/nontext classification method for normalized images such as Fig. 8(f).

A. Feature Extraction from a Square Block

Our feature vector is based on mesh and gradient features as adopted in [25]. We divide each square into 4 horizontal and vertical ones as shown in Fig. 9(b) and extract features. For a horizontal block H_i $(i = 1, 2, 3, 4)$, we consider

1) the number of white pixels,
2) the number of vertical white-black transitions,
3) the number of vertical black-white transitions

as features, and features for a vertical block is similarly defined.

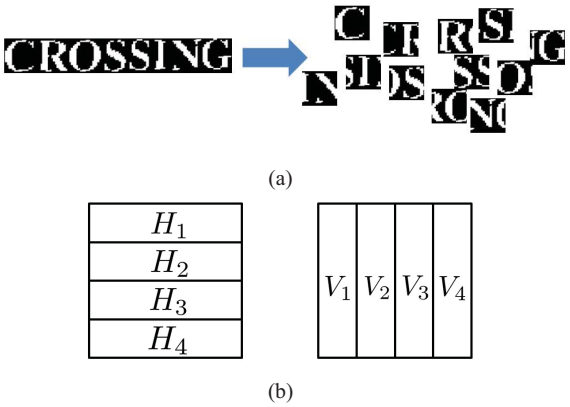


Fig. 9. (a) In order to handle variable aspect ratios, we split a block into squares. (b) For the feature extraction, we divide a square block into four horizontal and four vertical blocks.

B. Multilayer Perceptron Learning

For the training, we need normalized images such as Fig. 8(f). For this goal, we applied our algorithm presented in the previous sections (i.e., candidate generation and normalization algorithms) to the training images in [6]. Then, we manually classified them into text and nontext. We discarded some images showing poor binarization results, and collected 676 text block images and 863 nontext block images. However, we have found that more negative samples are needed for the reliable rejection of nontext components and collected more negative samples by applying the same procedure to images that do not contain any text. Finally, we have 3,568 nontext images. These text/nontext images are divided into squares as illustrated in Fig. 9(a) and we have trained a multi-layer perceptron for the classification of square patches [18]. We use one hidden layer consisting of 20 nodes and set the output value to $+1$ for text samples and 0 otherwise. To help the learning, input features are normalized.

C. Integration of Decision Results

For the integration of square classification results, we accumulate the outputs of the classifier:

$$\psi(w_k) = \sum_{i \in P} F_i \tag{19}$$

where P is the square patch set (e.g., the right hand side of Fig. 9(a)) and F_i is the continuous output of the classifier for the i-th square block in P. We consider $\psi(w_k)$ as a textness measure and classify w_k as a text region when

$$\frac{1}{|P|}\psi(w_k) > \tau_2 \tag{20}$$

where $|P|$ is the number of patches.

(a) (b)

Fig. 10. (a) Our detection results. (b) Transformed results of (a) for the comparison with the rectangle-based ground truth.

Textness measure (19) is also useful for imposing a non-overlap constraint, i.e., two text blocks should not be overlapping. When two localized regions are significantly overlapping, we simply choose a block showing a higher $\psi(\cdot)$ value.

D. Discussion

We tried to build a text/nontext classifier based on normalized gray-scale images (without binarizaton), because gray scale images seem to be more informative. To be precise, we adopted the AdaBoost learning method and gradient features [26]. However, experiments have shown that this approach yielded almost the same performance, with a considerable amount of overhead in training. We also tried to use both classifiers in a row (neural network with binary images and Adaboost with gray images), however, we could not find noticeable gains.

VI. Experimental Results

In experiments, we have used ground truth and evaluation tools released by [5], [6]. Since our method provides detection results considering skews as shown in Fig. 10(a), we have transformed these parallelograms into rectangles as shown in Fig. 10(b) for the evaluation.

A. Word Segmentation Heuristics

Although word segmentation is not the main issue of the scene text detection problem, it is essential in the evaluation. Hence, we have developed a heuristic rule that partitions detected text boxes into words. Given a cluster w_k, we sort distances between adjacent CCs in a normalized image in descending order and estimate a (minimum) word spacing distance T. If the distance between adjacent CCs is over T, a separation between them occurs. To be precise,

$$T = \alpha \times Q + \beta \tag{21}$$

where Q is the top 65% distance value. We empirically set $\alpha = 1.5$ and $\beta = 3$.

B. Exploitation of Multichannel Information

The MSER algorithm extracts CCs in a single-channel (scalar valued) image. Therefore, our method sometimes suffers from the loss of information during color reduction as illustrated in Fig. 11(a) and (b). However, such a text may become clear in another channel as shown in Fig. 11(c), and we can alleviate this problem by applying our method to multiple channels. Note that our method is invariant to affine transform of intensity and we can apply our method to the chrominance channel without any modification. In case that similar (significantly overlapping) text blocks are detected in more than one channel, we have selected one text block having the largest textness value.

One may think that CC extraction methods in multi-channel images can alleviate the same problem, even if they are much slower than single channel methods (for example, the maximally stable color region algorithm is several times slower than the original MSER algorithm [27]). However, they may yield over-segmented results. Note that characters in color (vector valued) images are not homogeneous (See Fig. 11(a)) but they consist of several clusters in color spaces.

C. Experimental Results on ICDAR 2011 Dataset

We submitted our preliminary results to ICDAR 2011 robust reading competition and we won the first prize. The results are summarized in Table I, where color spaces, such as L–a–b, H–S–V, and Y–Cb–Cr, were also evaluated to verify the effectiveness of exploiting multi-channel information. Since the Y–Cb–Cr space shows the best performance, it is selected among the color spaces for further comparison. We have also plotted our precision-recall curves in Fig. 12. A solid blue line stands for a precision-recall curve when Y, Cb, and Cr channels are adopted, and a dotted red line means a precision-recall curve when the luminance channel is only considered. We have drawn these graphs by changing a free parameter τ_2 in (20).

We have also conducted experiments showing the effects of multi-channel processing. Five points on the green dotted line are operating points when we add a new channel image sequentially. Starting from a luminance channel image, we have added Cb, Cr, R, and B channels sequentially. The performance of our method is improved when we add two chrominance channels, however, its precision drops when other channels (such as R and B channels) are added. The effect of our multi-channel scheme can be found in Fig. 11(d).

Our method takes about 50 ms for 640 $\times$ 480 luminance inputs for a standard PC with 2.8 GHz Intel processor. Also, it takes about 120 ms when three channels are used. Specifically, the complexity of our method is not linear to the number of channels since chrominance channel images usually have less CCs than luminance one.

D. Experimental Results on ICDAR 2005 Dataset

We have evaluated the performance on ICDAR 2005 dataset with the traditional measure [4]. As in Table II, color spaces, such as L–a–b, H–S–V, and Y–Cb–Cr, were evaluated and the

Fig. 11. Some text is not salient in a luminance channel image and we can detect such a text by exploiting multichannel information. (a) Color input. (b) Luminance channel image. (c) Chrominance channel image. (d) Our detection result using Y and Cr channels.

TABLE I
EVALUATION ON ICDAR 2011 DATASET (WITH A NEW MEASURE) [6]

Algorithm	Precision	Recall	f-Measure
Yi's	0.6722	0.5809	0.6232
TH-TextLoc	0.6697	0.5768	0.6198
Neumann's	0.6893	0.5254	0.5963
TDMIACS	0.6352	0.5352	0.5809
Our method (Y)	0.8123	0.6441	0.7185
Our method (L–a–b)	0.8150	0.6469	0.7213
Our method (H–S–V)	0.7070	0.6749	0.6905
Our method (Y–Cb–Cr)	0.8144	0.6868	0.7452

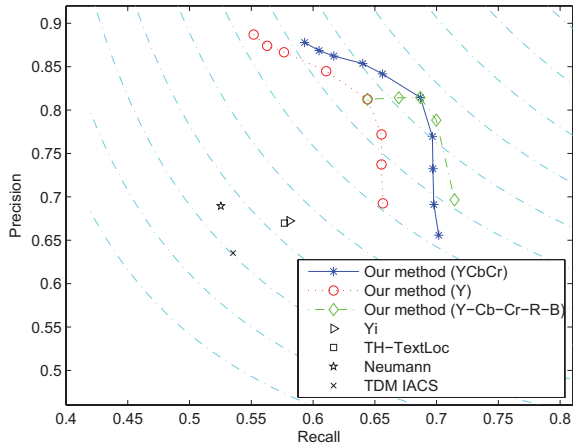


Fig. 12. Precision-recall curves on ICDAR 2011 dataset.

TABLE II
EVALUATION ON 2005 DATASET (WITH A CONVENTIONAL MEASURE) [4], [5]

Algorithm	Precision	Recall	f
Hinnerk Becker	0.62	0.67	0.62
SWT [10]	0.73	0.60	0.66
Alex Chen [7]	0.60	0.60	0.58
Hybrid [9]	0.674	0.697	0.685
Our method (Y)	0.764	0.619	0.684
Our method (L–a–b)	0.791	0.620	0.695
Our method (H–S–V)	0.678	0.666	0.672
Our method (Y–Cb–Cr)	0.778	0.649	0.708

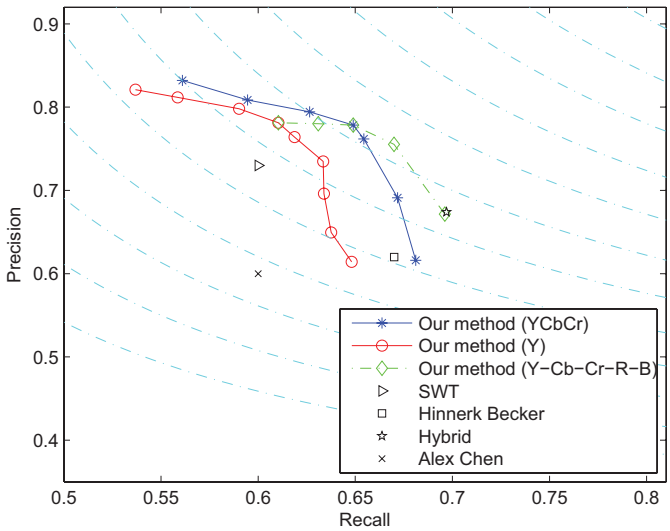


Fig. 13. Precision-recall curves on ICDAR 2005 dataset.

Y–Cb–Cr space also shows the best performance. In Fig. 13, we have plotted our precision-recall curves. Note that we have also illustrated operating points of conventional methods [4], [7], [9], [10]. Although our recall rate is worse than the method in [9], our method shows better precision and f-measure. Especially, our method is (at least) several times faster than that method.

E. Effects of Parameters

Our method has two trained classifiers, i.e., adjacency relation classifier with a parameter τ_1 in (13) and text/nontext classifier with a parameter τ_2 in (20). Then, as shown in Fig. 14, τ_1 and τ_2 can be utilized in controlling the overall precision-recall. Unfortunately, τ_1 seems not to control the precision and recall in an inversely proportional way. Specifically, Fig. 14 shows that the overall recall could drop in the case of a small τ_1 which definitely increases the recall of adjacency relation classifier. The increased recall in the adjacency relation classifier is likely to decrease its precision so the (3) and (4) cases in Fig. 3 could be classified as positive samples. This may result in the failure in localizing text boxes and drop the overall recall. On the other hand, a large τ_2 increases the overall precision (decreases the overall recall), and vice versa, so we think τ_1 should be fixed while τ_2 can be changed according to the purpose of the application (high recall versus high precision).

We have also illustrated precision-recall curves for different MSER parameters in Fig. 15. The OpenCV library provides

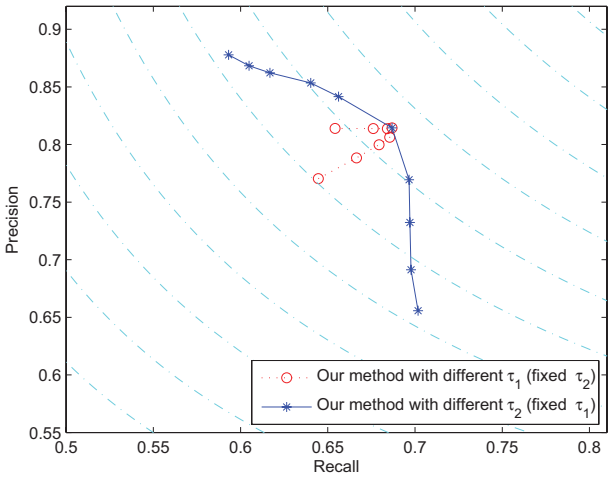


Fig. 14. Precision-recall curves with different parameters (τ_1 and τ_2). YCbCr channels are used and the measure in [6] is adopted for the evaluation.

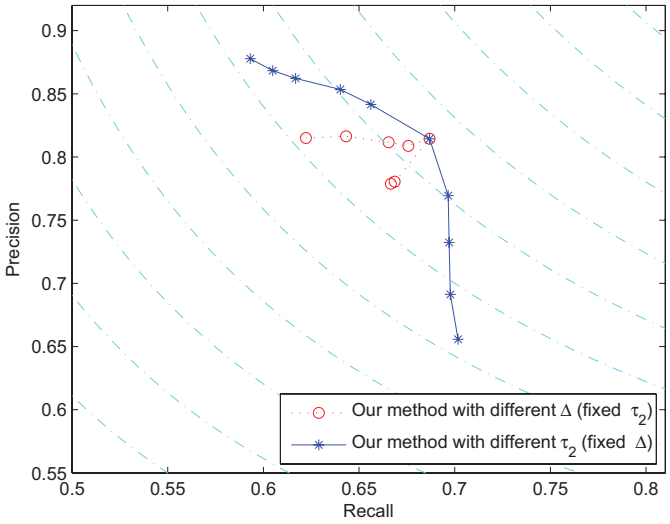


Fig. 15. Precision-recall curves with different parameters (Δ and τ_2). YCbCr channels are used and the measure in [6] is adopted for the evaluation.

an efficient implementation of the MSER algorithm [20] and experiments have shown that $\Delta = 6$ yielded the best f-measure, where Δ represents the range of intensities where the regions are stable (interested readers may refer to [28] for the definition of Δ). $\Delta = 6$ is a relatively small value compared to the whole range of intensities ($0 \sim 255$), and our CC extraction method could generate low-contrast or blurred CCs (high recall). However, as shown in Fig. 15, the Δ value seems not to control the precision and recall as τ_1.

F. Future Work

Our system is designed to address the scene text detection problem in natural images such as Fig. 2(a) and 11(a), where English alphabets are placed horizontally [4]–[6]. We have exploited these properties in our algorithm (especially in feature selection), and our method should be changed in order to detect Asian scripts and/or texts of arbitrary orientations [9], [12]. We think our general framework can be extended by developing new features and merging rules, and this is our future research direction.

VII. Conclusion

In this paper, we have presented a novel scene text detection algorithm based on machine learning techniques. To be precise, we developed two classifiers: one classifier was designed to generate candidates and the other classifier was for the filtering of nontext candidates. We have also presented a novel method to exploit multi-channel information. We have conducted experiments on ICDAR 2005 and 2011 datasets which showed that our method yielded the state-of-the-art performance in both new and traditional evaluation protocols.

References

[1] K. Jung, "Text information extraction in images and video: A survey," *Pattern Recognit.*, vol. 37, no. 5, pp. 977–997, May 2004.

[2] J. Liang, D. Doermann, and H. Li, "Camera-based analysis of text and documents: A survey," *Int. J. Document Anal. Recognit.*, vol. 7, nos. 2–3, pp. 84–104, 2005.

[3] J. Zhang and R. Kasturi, "Extraction of text objects in video documents: Recent progress," in *Proc. 8th IAPR Int. Workshop Document Anal. Syst.*, Sep. 2008, pp. 5–17.

[4] S. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," in *Proc. Int. Conf. Document Anal. Recognit.*, 2003, pp. 682–687.

[5] S. Lucas, "Icdar 2005 text locating competition results," in *Proc. Int. Conf. Document Anal. Recognit.*, 2005, pp. 80–84.

[6] A. Shahab, F. Shafait, and A. Dengel, "ICDAR 2011 robust reading competition challenge 2: Reading text in scene images," in *Proc. Int. Conf. Document Anal. Recognit.*, 2011, pp. 1491–1496.

[7] X. Chen and A. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2004, pp. 366–373.

[8] X. Chen and A. Yuille, "A time-efficient cascade for real-time object detection: With applications for the visually impaired," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Workshops*, Jun. 2005, pp. 1–8.

[9] Y.-F. Pan, X. Hou, and C.-L. Liu, "A hybrid approach to detect and localize texts in natural scene images," *IEEE Trans. Image Process.*, vol. 20, no. 3, pp. 800–813, Mar. 2011.

[10] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2963–2970.

[11] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. Image Process.*, vol. 20, no. 9, pp. 2594–2605, Sep. 2011.

[12] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1083–1090.

[13] H. Chen, S. Tsai, G. Schroth, D. Chen, R. Grzeszczuk, and B. Girod, "Robust text detection in natural images with edge-enhanced maximally stable extremal regions," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2011, pp. 2609–2612.

[14] J. Matas and K. Zimmermann, "A new class of learnable detectors for categorisation," in *Proc. Scandinavian Conf. Image Anal.*, Jun. 2005, pp. 541–550.

[15] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Proc. Asian Conf. Comput. Vis.*, 2010, pp. 770–783.

[16] J. Matas, O. Chum, U. Martin, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *Proc. Brit. Mach. Vis. Conf.*, 2002, pp. 384–393.

[17] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Ann. Stat.*, vol. 28, no. 2, pp. 337–407, 1998.

[18] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1998.

[19] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[20] D. Nistér and H. Stewénius, "Linear time maximally stable extremal regions," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 183–196.

[21] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. Cambridge, MA, USA: O'Reilly, 2008.

[22] A. Torralba, K. P. Murphy, and W. T. Freeman, "Sharing features: Efficient boosting procedures for multiclass object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun.–Jul. 2004, pp. 762–769.

[23] M. A. Weiss, *Data Structures and Algorithm Analysis in C++*, 2nd ed. Boston, MA, USA: Addison-Wesley, 1998.

[24] R. Casey and E. Lecolinet, "A survey of methods and strategies in character segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 7, pp. 690–706, Jul. 1996.

[25] I.-S. Oh and C. Y. Suen, "Distance features for neural network-based recognition of handwritten characters," *Int. J. Document Anal. Recognit.*, vol. 1, no. 2, pp. 73–88, 1998.

[26] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2001, pp. 511–518.

[27] P.-E. Forssén, "Maximally stable colour regions for recognition and matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.

[28] *OpenCV 2.4.3. Documentation*. (2008) [Online]. Available: http://docs.opencv.org/

Hyung Il Koo (S'09–M'10) received the B.S., M.S., and Ph.D. degrees from the Department of Electrical Engineering and Computer Science, Seoul National University (SNU), Seoul, Korea, in 2002, 2004, and 2010, respectively.

He was a Research Engineer with the Qualcomm Research Korea, Seoul, from 2010 to 2012. He joined the Division of Electrical and Computer Engineering, Ajou University, Gyeonggi-do, Korea, in 2012, where he is currently an Assistant Professor. His current research interests include computer vision and machine learning.

Duck Hoon Kim (S'03–M'06) received the B.S., M.S., and Ph.D. degrees in electrical engineering and computer science from Seoul National University, Seoul, Korea, in 1998, 2000, and 2005, respectively.

He was a Post-Doctoral Researcher with Seoul National University and the University of Southern California, Los Angeles, CA, USA, from March 2005 to October 2006, and a Senior Engineer with Samsung Electronics from November 2006 to March 2011. He joined Qualcomm in March 2011 and is currently with Qualcomm Research Korea as a Staff Engineer. His current research interests include computer vision, computer graphics, and multimedia application.