A New Approach for Overlay Text Detection and Extraction From Complex Video Scene

Wonjun Kim and Changick Kim, *Member, IEEE*

***Abstract*—Overlay text brings important semantic clues in video content analysis such as video information retrieval and summarization, since the content of the scene or the editor's intention can be well represented by using inserted text. Most of the previous approaches to extracting overlay text from videos are based on low-level features, such as edge, color, and texture information. However, existing methods experience difficulties in handling texts with various contrasts or inserted in a complex background. In this paper, we propose a novel framework to detect and extract the overlay text from the video scene. Based on our observation that there exist transient colors between inserted text and its adjacent background, a transition map is first generated. Then candidate regions are extracted by a reshaping method and the overlay text regions are determined based on the occurrence of overlay text in each candidate. The detected overlay text regions are localized accurately using the projection of overlay text pixels in the transition map and the text extraction is finally conducted. The proposed method is robust to different character size, position, contrast, and color. It is also language independent. Overlay text region update between frames is also employed to reduce the processing time. Experiments are performed on diverse videos to confirm the efficiency of the proposed method.**

***Index Terms*—Optical character recognition (OCR), overlay text, transition map, video information retrieval, video summarization.**

I. Introduction

With the development of video editing technology, there are growing uses of overlay text inserted into video contents to provide viewers with better visual understanding. Most broadcasting videos tend to increase the use of overlay text to convey more direct summary of semantics and deliver better viewing experience. For example, headlines summarize the reports in news videos and subtitles in the documentary drama help viewers understand the content. Sports videos also contain text describing the scores and team or player names [1]. In general, text displayed in the videos can be classified into scene text and overlay text [2]. Scene text occurs naturally in the background as a part of the scene, such as the advertising boards, banners, and so on. In contrast to that, overlay text is superimposed on the video scene and used to help viewers' understanding. Since the overlay text is highly compact and structured, it can be used for video indexing and retrieval [3]. However, overlay text extraction for video optical character recognition (OCR) becomes more challenging, compared to the text extraction for OCR tasks of document images, due to the numerous difficulties resulting from complex background, unknown text color, size and so on.

There are two steps involved before the overlay text recognition is carried out, i.e., detection and extraction of overlay text. First, overlay text regions are roughly distinguished from background. The detected overlay text regions are refined to determine the accurate boundaries of overlay text strings. To generate a binary text image for video OCR, background pixels are removed from the overlay text strings in the extraction step. Although many methods have been proposed to detect and extract the video text, few methods can effectively deal with different color, shape, and multilingual text.

Most of existing video text detection methods have been proposed on the basis of color, edge, and texture-based feature. Color-based approaches assume that the video text is composed of a uniform color. In the approach by Agnihotri *et al.* [4], the red color component is used to obtain high contrast edges between text and background. In [5], the "uniform color" blocks within the high contrast video frames are selected to correctly extract text regions. Kim *et al.* [6] cluster colors based on Euclidean distance in the RGB space and use 64 clustered color channels for text detection. However, it is rarely true that the overlay text consists of a uniform color due to degradation resulting from compression coding and low contrast between text and background. Edge-based approaches are also considered useful for overlay text detection since text regions contain rich edge information. The commonly adopted method is to apply an edge detector to the video frame and then identify regions with high edge density and strength. This method performs well if there is no complex background and it becomes less reliable as the scene contains more edges in the background. Lyu *et al.* [7] use a modified edge map with strength for text region detection and localize the detected text regions using coarse-to-fine projection. They also extract text strings based on local thresholding and inward filling. In [8], authors consider the strokes of text in horizontal, vertical, up-right, and up-left directions and generate the edge map along each direction. Then they combine statistical features and use k-means clustering to classify the image pixels into background and text candidates. Liu *et al.* [9] use multiscale edge detector to detect the text regions. They

Manuscript received June 13, 2008; revised October 05, 2008. First published December 16, 2008; current version published January 09, 2009. This research was supported by the Ministry of Knowledge Economy, Korea, under the Information Technology Research Center support program supervised by the Institute of Information Technology Advancement (grant number IITA-2008-C1090-0801-0017). The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Jenq-Neng Hwang.

The authors are with the Department of Electronic Engineering, Information and Communications University, Daejeon, Korea (e-mail: jazznova; ckim@icu.ac.kr).

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.2008.2008225

compute the edge strength, density, and orientation variance to form the multiscale edge detector. Texture-based approaches, such as the salient point detection and the wavelet transform, have also been used to detect the text regions. Bertini *et al.* [10] detect corner points from the video scene and then detect the text region using similarity of corner points between frames. Sato *et al.* [11] apply an interpolation filter based on vertical, horizontal, left diagonal, right diagonal directions to enhance the performance of text extraction. Gllavata *et al.* [2] employ the high-frequency wavelet coefficients and connected components to detect the text regions. However, since it is almost impossible to detect text in a real video by using only one characteristic of text, some methods take advantage of combined features to detect video text [12], [13].

After the text detection step, the text extraction step, which can be classified into color-based [7], [14] and stroke-based methods [11], should be employed before OCR is applied. Since color of text is generally different from that of background, text strings can be extracted by thresholding. Otsu method [14] is a widely used color-based text extraction method due to its simplicity and efficiency of the algorithm. However, Otsu method is not robust to text extraction with similar color of background due to the use of global thresholding. To solve this problem, the detected text regions are divided into several blocks and then Otsu method is applied locally to each block, such as the adaptive thresholding introduced in [7], where a dam point is defined to extract text strings from background. On the other hand, some filters based on the direction of strokes have also been used to extract text in the stroke-based methods. The four-direction character extraction filters [11] are used to enhance the stroke-like shapes and to suppress others. However, since the stroke filter is language-dependent, some characters without obvious stripe shape can also be suppressed.

In this paper, we propose a new overlay text detection and extraction method using the transition region between the overlay text and background. First, we generate the transition map based on our observation that there exist transient colors between overlay text and its adjacent background. Then the overlay text regions are roughly detected by computing the density of transition pixels and the consistency of texture around the transition pixels. The detected overlay text regions are localized accurately using the projection of transition map with an improved color-based thresholding method [7] to extract text strings correctly. The rest of this paper is organized as follows. We generate the transition map and refine the detected text regions in Section II. The overlay text extraction from the refined text regions is explained in Section III. The experimental results on various videos are shown in Section IV, followed by conclusion in Section V.

II. Overlay Text Region Detection

The proposed method is based on our observations that there exist transient colors between overlay text and its adjacent background (see Fig. 1) and overlay texts have high saturation because they are inserted by using graphic components. The overall procedure of proposed overlay text detection method is shown in Fig. 2, where each module is explained in Sections II-A–E. The overlay text extraction method will be clearly explained in Section III.

Fig. 1. Examples of overly text.

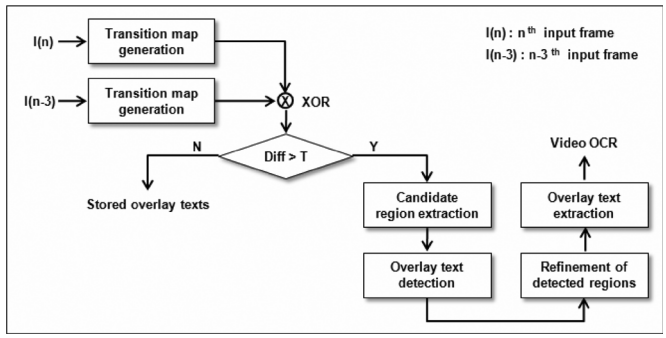


Fig. 2. Overall procedure of the proposed detection method.

A. Transition Map Generation

As a rule of thumb, if the background of overlay text is dark, then the overlay text tends to be bright. On the contrary, the overlay text tends to be dark if the background of overlay text is bright. Therefore, there exists transient colors between overlay text and its adjacent background due to color bleeding, the intensities at the boundary of overlay text are observed to have the logarithmical change.

As shown in Fig. 3(a), the intensities of three consecutive pixels are decreasing logarithmically at the boundary of bright overlay text due to color bleeding by the lossy video compression. It is also observed that the intensities of three consecutive pixels increases exponentially at the boundary of dark overlay text. Graphical illustration of intensity change in the transition region is also shown in Fig. 3(b). We first prepare three videos containing "Bright to Dark (B-D)" and "Dark to Bright (D-B)" transitions, respectively. 20 transition areas sampled from each video are averaged and illustrated in the figure.

Since the change of intensity at the boundary of overlay text may be small in the low contrast image, to effectively determine whether a pixel is within a transition region, the modified saturation is first introduced as a weight value based on the fact that overlay text is in the form of overlay graphics. The modified saturation is defined as follows:

$$S(x,y) = 1 - \frac{3}{(R+G+B)[\min(R,G,B)]} \tag{1}$$

$$\tilde{S}(x,y) = \frac{S(x,y)}{\max(S(x,y))}$$

$$\text{where } \max(S(x,y)) = \begin{cases} 2 \times (0.5 - \tilde{I}(x,y)), & \text{if } \tilde{I}(x,y) > 0.5 \\ 2 \times \tilde{I}(x,y), & \text{otherwise.} \end{cases} \tag{2}$$

$S(x,y)$ and $\max S((x,y))$ denote the saturation value and the maximum saturation value at the corresponding intensity

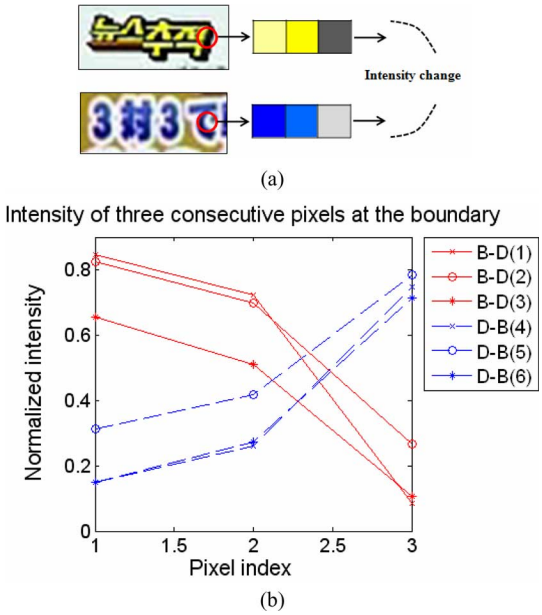


Fig. 3. (a) Change of intensities in the transition region. (b) Graphical illustration of intensity change in the transition region (red: bright to dark, blue: dark to bright).

level, respectively. $\tilde{I}(x,y)$ denotes the intensity at the (x,y), which is normalized to $[0,1]$. Based on the conical HSI color model [15], the maximum value of saturation is normalized in accordance with $\tilde{I}(x,y)$ compared to 0.5 in (2). The transition can thus be defined by combination of the change of intensity and the modified saturation as follows:

$$\begin{aligned} &D_L(x,y) \\ &\quad = (1 + dS_L(x,y)) \times |I(x-1,y) - I(x,y)| \\ &D_H(x,y) \\ &\quad = (1 + dS_H(x,y)) \times |I(x,y) - I(x+1,y)| \\ &\text{where } dS_L(x,y) \\ &\quad = |\tilde{S}(x-1,y) - \tilde{S}(x,y)| \text{ and} \\ &dS_H(x,y) \\ &\quad = |\tilde{S}(x,y) - \tilde{S}(x+1,y)|. \end{aligned} \tag{3}$$

Since the weight $dS_L(x,y)$ and $dS_H(x,y)$ can be zero by the achromatic overlay text and background, we add 1 to the weight in (3). If a pixel satisfies the logarithmical change constraint given in (4), three consecutive pixels centered by the current pixel are detected as the transition pixels and the transition map is generated

$$T(x,y) = \begin{cases} 1, & \text{if } D_H > D_L + TH \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The thresholding value *TH* is empirically set to 80 in consideration of the logarithmical change. Two transition maps are shown in Fig. 4(b) and (d), which correspond to the overlay text inserted images in Fig. 4(a) and (c), respectively. We can see that the transition maps are well generated even in the complex background.

Fig. 4. (a), (c) Original image. (b), (d) Transition map.

Fig. 5. (a) Transition map. (b) Linked map through connected components building. (c) Smoothed candidate regions based on rectangular bounding box fitting.

B. Candidate Region Extraction

The transition map can be utilized as a useful indicator for the overlay text region. To generate the connected components, we first generate a linked map as shown in Fig. 5(b). If a gap of consecutive pixels between two nonzero points in the same row is shorter than 5% of the image width, they are filled with 1s. If the connected components are smaller than the threshold value, they are removed. The threshold value is empirically selected by observing the minimum size of overlay text region. Then each connected component is reshaped to have smooth boundaries. Since it is reasonable to assume that the overlay text regions are generally in rectangular shapes, a rectangular bounding box is generated by linking four points, which correspond to $(\min\_x, \min\_y)$, $(\max\_x, \min\_y)$, $(\min\_x, \max\_y)$, $(\max\_x, \max\_y)$ taken from the link map shown in Fig. 5(b). The refined candidate regions are shown in Fig. 5(c).

C. Overlay Text Region Determination

The next step is to determine the real overlay text region among the boundary smoothed candidate regions by some useful clues, such as the aspect ratio of overlay text region. Since most of overlay texts are placed horizontally in the video, the vertically longer candidates can be easily eliminated. The density of transition pixels is a good criterion as well. Nevertheless, a more refined algorithm is needed to minimize the false detection due to the complex background. In this subsection, we introduce a texture-based approach for overlay text region determination.

Based on the observation that intensity variation around the transition pixel is big due to complex structure of the overlay text, we employ the local binary pattern (LBP) introduced in

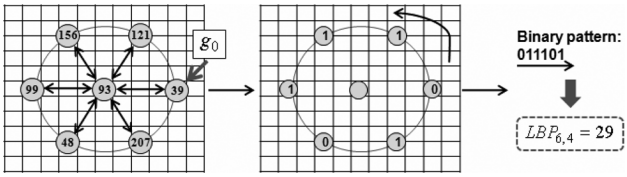


Fig. 6. Example of LBP computation.

TABLE I
NUMBER OF DIFFERENT LBP'S AND OVERLAY TEXT PIXELS IN EACH CANDIDATE SHOWN IN FIG. 5(C)

	Candidate 1 (C1)	Candidate 2 (C2)	Candidate 3 (C3)
# of different LBP's	74	55	78
# of transition pixels	404	125	381

[16] to describe the texture around the transition pixel. LBP is a very efficient and simple tool to represent the consistency of texture using only the intensity pattern. LBP forms the binary pattern using current pixel and its all circular neighbor pixels and can be converted into a decimal number as follows:

$$\mathrm{LBP}_{P,R} = \sum_{i=0}^{P-1} s(g_i - g_c)2^i, \text{ where } s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0. \end{cases} \tag{5}$$

P and R denote the user's chosen number of circular neighbor pixels of a specific pixel and the radius of circle, respectively. g_c and g_i denote the intensity of current pixel and circular neighbor pixels, respectively. We can obtain the binary pattern as shown in Fig. 6, and the resulting $\mathrm{LBP}_{6,4} = 29 (= 2^4 + 2^3 + 2^2 + 2^0)$.

Now we define the probability of overlay text (POT) using the operator as follows: The LBP operator is first applied to every transition pixel in each candidate region. Then, we compute the number of different LBPs to consider the intensity variation around the transition pixel. Since we use the 8 neighbor pixels to obtain the LBP value, the total number of potentially different LBPs is $2^8 = 256$. Although the number of different LBPs is generally increasing as the candidate region includes more transition pixels, it may not be guaranteed since transition pixels can have same local binary pattern. The number of different LBPs and transition pixels in each candidate region of Fig. 5(c) is shown in Table I.

Let ω_i denote the density of transition pixels in each candidate region and can be easily obtained from dividing the number of transition pixels by the size of each candidate region. POT is defined as follows:

$$\mathrm{POT}_i = \omega_i \times \mathrm{NOL}_i, \quad i = 1, \ldots, N \tag{6}$$

where N denotes the number of candidate regions as mentioned. NOL_i denotes the number of different LBPs, which is normalized by the maximum of the number of different LBPs (i.e., 256) in each candidate region. If POT of the candidate region is larger than a predefined value, the corresponding region is finally determined as the overlay text region. The detection result is shown in Fig. 7 and Table II. The thresholding value in POT is empirically set to 0.05 based on various experimental results. We can see that the overlay text region is well identified from other candidates.

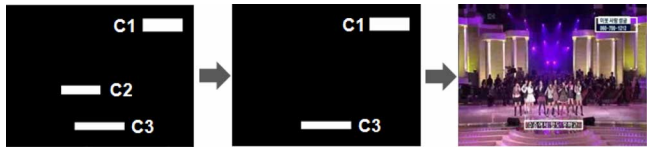


Fig. 7. Overlay text region determination.

TABLE II
POT IN EACH CANDIDATE REGION

Candidates	POT	Determination result
C1	0.078	Overlay text
C2	0.027	Background
C3	0.105	Overlay text

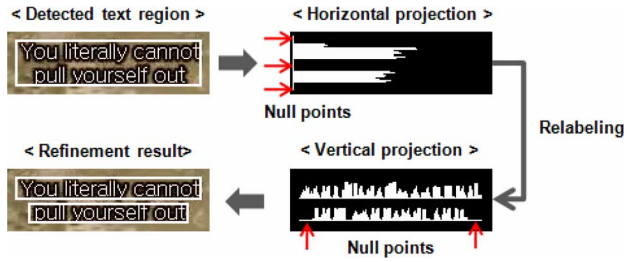


Fig. 8. Refinement process of detected overlay text region.

D. Overlay Text Region Refinement

The overlay text region or the bounding box obtained in the preceding subsection needs to be refined for better accurate text extraction, which will be addressed in Section III. In this subsection, we use a modified projection of transition pixels in the transition map [7] to perform the overlay text region refinement. First, the horizontal projection is performed to accumulate all the transition pixel counts in each row of the detected overlay text region to form a histogram of the number of transition pixels. Then the null points, which denote the pixel row without transition pixels, are removed and separated regions are re-labeled. The projection is conducted vertically and null points are removed once again. Compared to the coarse-to-fine projection proposed for edge-based scheme in [7], our projection method is applied to the detected overlay text regions only, making the process simpler. The result of refinement is shown in Fig. 8.

E. Overlay Text Region Update

Once the overlay text regions are detected in the current frame, it is reasonable to take advantage of continuity of overlay text between consecutive frames for the text region detection of the next frame. If the difference, which can be obtained by XOR of current transition map and previous transition map, is smaller than a predefined value, the overlay text

Fig. 9. Example of gradually appearing overlay text.

regions of previous frame are directly applied as the detection result without further refinement. A special situation that needs to be taken care of happens when an overlay text region is in gradual appearance as shown in Fig. 9.

In order to deal with such changes, we compare the current transition map with the transition map obtained 3 frames earlier and the dissimilarity measure between these maps is defined as follows:

$$d(T_n, T_{n-3}) = \sum_{(x,y)\in T} (T_n(x,y) \otimes T_{n-3}(x,y)) \quad (7)$$

$$\text{if}(d(T_n, T_{n-3}) < th) TR_n = TR_{n-3}$$
$$\text{otherwise, find new } TR_n \quad (8)$$

where T_n and T_{n-3} denote the transition map obtained from the nth frame and the $(n-3)$th frame, respectively. TR_n and TR_{n-3} denote the detected overlay text regions in the nth frame and the $(n-3)$th frame, respectively. $\otimes$ denotes the XOR operator. In other words, if the values on the nth frame and the $(n-3)$th frame transition map are same, the result of $\otimes$ between two values is set to be 0. Otherwise, the result of $\otimes$ between two values is set to be 1. The overlay text region update method can reduce the processing time efficiently.

III. Overlay Text Extraction

Before applying video OCR application, the refined overlay text regions need to be converted to a binary image, where all pixels belonging to overlay text are highlighted and others suppressed. Since the text color may be either brighter or darker than the background color, an efficient scheme is required to extract the overlay text dealing with complex backgrounds and various text appearances. Among several algorithms proposed to conduct such a task [7], [11], [17], there is one effective method, proposed by Lyu *et al.* [7], consists of color polarity classification and color inversion (if necessary), followed by adaptive thresholding, dam point labeling and inward filling. In this section, we propose a fast and efficient overlay text extraction technique, which is based on Lyu's approach with some modifications for better performance. The overall procedure of proposed extraction method is shown in Fig. 10.

A. Color Polarity Computation

Fig. 11 shows two opposite scenarios, in which either the overlay text is darker than the surrounding background

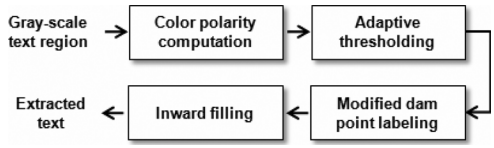


Fig. 10. Overall procedure of proposed extraction method.

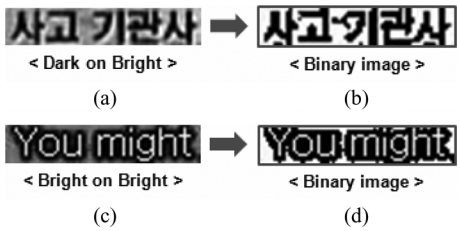


Fig. 11. Examples of binarized image by the average intensity of overlay text region.

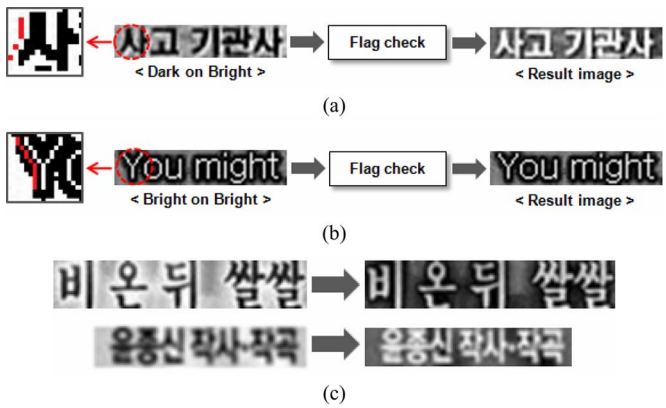


Fig. 12. Process of inversing image by the color polarity. (a) Dark text on bright background. (b) Bright text on bright background. (c) Examples of inverted overlay text by "bright_text_flag".

[Fig. 11(a)], or the text is brighter than its neighbors [Fig. 11(c)]. As shown in Fig. 11(b) and (d), the binarized images obtained by simple thresholding represent the overlay text as either 1 (or " *White*") or 0 (or "*Black*"), respectively. Such inconsistent results must complicate the following text extraction steps. Thus, our goal in this subsection is to check the color polarity and inverse the pixel intensities if needed so that the output text region of the module can always contain bright text compared to its surrounding pixels as shown in Fig. 12.

We observe that this goal can be simply attained owing to the transition map obtained in the preceding section. First of all, the binary image obtained by thresholding with average intensity value can be effectively utilized [see Fig. 11(b) and (d)]. Given the binarized text region, the boundary pixels, which belong to left, right, top, and bottom lines of the text region, are searched and the number of white pixels is counted. If the number of white boundary pixels is less than 50% of the number of boundary pixels, the text region is regarded as "bright text on dark background" scenario, which requires no polarity change. In other words, the overlay text is always bright in such scenarios. If the number of white pixels is greater than that of black pixels, we conduct a task to turn on or off the "bright_text_flag"

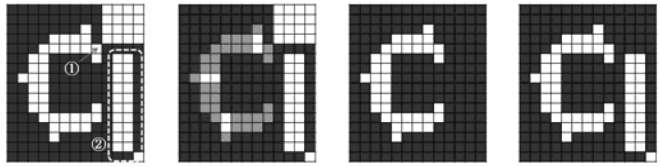
Fig. 13. Process of adaptive thresholding adopted from [7]. Horizontal adaptive thresholding (left). Vertical adaptive thresholding (right).

as expressed in (9), shown at the bottom of the page, where (x_F, y_F) denotes the position of the first encountered transition pixel in each row of the text region and I_B denotes the value on the binary image. As shown in Fig. 12, the flag is set to 1 for Fig. 12(a) since the first encountered transition pixel belongs to 1, whereas the pixel apart by two pixel distance belongs to 0. If such case happens at least once, the pixel values in the text region is inverted to make the overlay text brighter than the surrounding background. Note that the inversion is simply done by subtracting the pixel value from the maximum pixel value. The process of color polarity computation is shown in Fig. 12. The first transition pixels in each row on the binary image are represented by red color in Fig. 12. Examples with "bright_flag_text" are also shown in Fig. 12(c).

B. Overlay Text Extraction

Since it is confirmed that the overlay text is always bright in each text region, it is safe to employ Lyu's method to extract characters from each overlay text region. First, each overlay text region is expanded wider by two pixels to utilize the continuity of background. This expanded outer region is denoted as *ER*. Then, the pixels inside the text region are compared to the pixels in *ER* so that pixels connected to the expanded region can be excluded. We denote the text region as *TR* and the expanded text region as *ETR*, i.e., $ETR = TR \cup ER$. Next, sliding-window based adaptive thresholding is performed in the horizontal and the vertical directions with different window sizes, respectively. Compared to the Lyu's method, the height of expanded text region is not normalized in our method. Let $ETR(x, y)$ and $B(x, y)$ denote gray scale pixels on *ETR* and the resulting binary image, respectively. All $B(x, y)$ are initialized as "*White*". The window with the size of $16 \times ETR\_\text{height}$ is moving horizontally with the stepping size 8 and then the window with the size of $(ETR\_\text{width}) \times (ETR\_\text{height}/4)$ is moving vertically with the stepping size $\text{int}(ETR\_\text{height}/8)$. If the intensity of $ETR(x, y)$ is smaller than the local thresholding value computed by Otsu method in each window, the corresponding $B(x, y)$ is set to be "*Black*". The process of applying the sliding windows for adaptive thresholding is shown in Fig. 13.

Authors of [7] assume that the background pixels in *TR* are generally connected to *ER* in terms of intensity. They use filling from *ER* to the connected pixels in *TR* to remove the background pixels. However, since the text pixels might be connected to the background pixels in *TR*, the unwanted removal can occur when the filling task is conducted. Thus, the "dam points" inside *TR* is defined to prevent the filling from flooding into text pixels. We modify the dam point definition introduced in [7] by adding a simple constraint for transition pixels to improve extraction performance as follows:



(a) (b) (c) (d)

Fig. 14. Text extraction procedure. (a) Adaptive thresholding. (b) Modified dam points labeled in gray. (c) Inward filling. (d) Inward filling when the original Lyu's method is used.

$$\begin{aligned}\text{Dam points} = \{(x,y) | (B(x,y) = \text{“}White\text{”} \\ \wedge N_T(x,y) \neq 0) \\ \wedge (\text{MIN\_}W \leq \min[H_{\text{len}}(x,y) V_{\text{len}}(x,y)] \\ \leq \text{MAX\_}W)\}\end{aligned} \tag{10}$$

where $\text{MIN\_}W = 1$, $\text{MAX\_}W = \text{int}(ER\_\text{height}/8)$ and N_T denotes the number of transition pixels among the horizontally connected pixels with (x, y). H_{len} and V_{len} denote the length of the connectivity with horizontal and vertical direction at the (x, y), respectively. For example, H_{len} is 6 and V_{len} is 2 on the pixel ① in Fig. 14(a). Since the height of *ETR* is 16 in the figure, $\text{MAX\_}W$ is set to be 2 by (10). The minimum of these values is 2, which belongs to the range defined in (10). Therefore, the pixel marked as ① is labeled as a dam point, which is represented in gray in Fig. 14(b).

Finally, we can obtain characters correctly from each overlay text region by the inward filling as addressed in [7]. If a pixel is "*White*" during the scanning of binarized pixels in *ER*, all the connected "*White*" pixels including the pixel itself are filled with "*Black*". After the inward filling, all non-"*Black*" pixels are set to be "*White*". The result of inward filling is shown in Fig. 14(c). We see that the background of text is well removed. Note that the condition for the number of transition pixels is added in this paper. If the constraint using transition pixels is not added in the dam point labeling, background region ② is also labeled as dam points as shown in Fig. 14(d).

IV. Experimental Results

In this section, the proposed system is implemented and evaluated to show the efficiency and robustness of the proposed method. We first compare the detection performance of our proposed transition map with the Harris corner map which has been

$$\text{bright\_text\_flag} = \begin{cases} 1, & \text{if } I_B(x_F, y_F) = 1 \text{ and } I_B(x_F + 2, y_F) = 0 \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

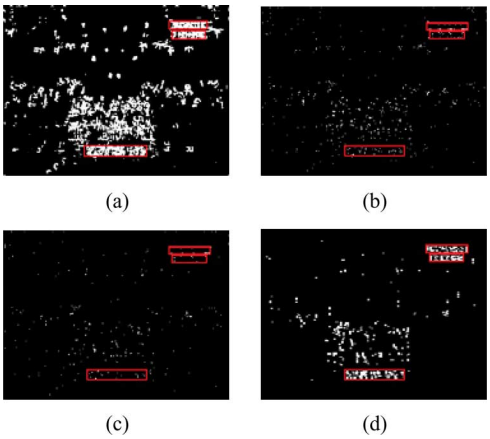


Fig. 15. Corner map and transition map of Fig. 4(a). Detected overlay text regions are represented by red rectangular box. (a) Corner map ($R = 0$). (b) Corner map ($R = 10$). (c) Corner map ($R = 20$). (d) Transition map.

TABLE III
COMPARISON PERFORMANCE OF CORNER MAP WITH TRANSITION MAP

	Corner map ($R = 0$)	Corner map ($R = 10$)	Corner map ($R = 20$)	Transition map
Recall	0.775	0.055	0.019	0.456
Precision	0.086	0.073	0.047	**0.193**
Total processing time (fps)	24.82	22.45	18.37	**32.05**

Fig. 16. (a) Overlay scene on the right-top. (b) Detected boundary of overlay scene. (c) Refined transition map.

frequently used for overlay text detection [18]. Then the detection performance of the overlay text region is evaluated by identifying the localized rectangular text regions and the average display time. Finally, we evaluate the overlay text extraction algorithm, which converts the gray scale overlay text regions to the corresponding binary images for further processing, such as video OCR.

A. Performance Comparison of the Transition Map With the Harris Corner Map

In this section, the Harris corner operator [18], which has been widely used in the literature, is used for comparison with the transition map. Here R denotes the reduction range. The pixel with the maximum corner value is only detected as a corner pixel within the range of $R \times R$ pixels.

In the case of the corner map with no reduction range (i.e., $R = 0$), many corner pixels are included in the detected overlay text region. However, since many corner pixels still belong to the background region, it is also highly likely to generate the false candidate region [see Fig. 15(a)]. Although the number of corner pixels on the background is decreasing as R increases, the number of corner pixels belonging to the overlay text decreases notably [see Fig. 15(b)–(c)]. In contrast to that, each overlay text region includes sufficient detected transition pixels in the transition map while a few detected transition pixels belong to the background [see Fig. 15(d)]. The proposed method makes possible to detect the overlay text region with higher efficiency. The recall and precision defined as below are used for evaluating detection rate

$$\text{Recall} = \frac{\text{Card}(P \cap T)}{\text{Card}(T)}, \quad \text{Precision} = \frac{\text{Card}(P \cap T)}{\text{Card}(P)} \tag{11}$$

where P denotes the set of detected pixels by each method and T denotes the number of pixels belonging to the overlay text. The performance evaluation of the Fig. 15 is shown in Table III.

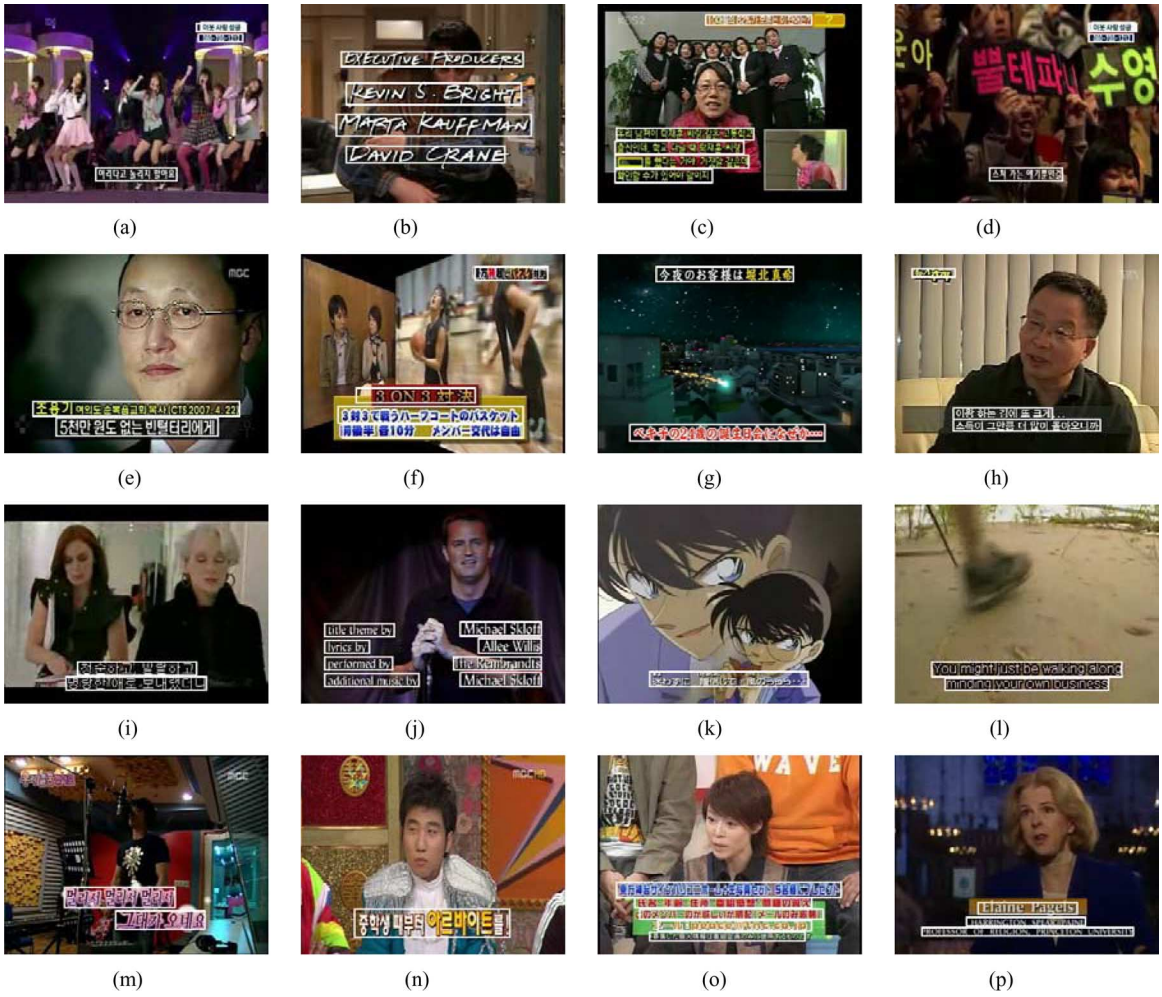
We can see that the transition map has higher *Precision* compared to the corner map. Since the following steps such as computing POT becomes simple with the small number of detected transition pixels, the high *Precision* is more desirable to detect the overlay text regions. On the other hand, since the empty space in the overlay text region of the transition map can be filled with the linked map as mentioned in Section II-B, it is not essential to have high *Recall* for an efficient detection. Moreover, the total processing (i.e., processing + decoding) speed for the transition map generation is much faster than that of the corner map. Since the search range increases as R increases, the total processing speed becomes slow as shown in Table III. Therefore, we conclude that the employment of transition map is highly desirable in positioning initial points for text extraction.

B. Performance Evaluation of Overlay Text Detection

Since the overlay scene can be inserted on the video scene like the overlay text is, the transition region is also observed at the boundary of the overlay scene. Moreover the boundary of overlay scene is vertically long in the transition map as shown in Fig. 16(b), we can easily remove the boundary of overlay scene from the transition map by the length of connectivity in the vertical direction.

The results of overlay text detection from complex video scenes are shown in Fig. 17. Overlay text regions are successfully detected regardless of color, size, position, style, and contrast. Although there exist different size of texts mixed in each image frame, the test results show the robustness of the proposed detection algorithm.

In the first row of Fig. 17 [i.e., Fig. 17(a)–(d)], the overlay text regions are successfully detected in the complex background. Although there are various colors and many edges in the background, the overlay text regions are correctly detected. In addition, the proposed method is robust to various text styles, as we can see, in particular, in Fig. 17(b). We can also see that the overlay texts embedded in the complex background are correctly located regardless of neighboring colors. For the overlay texts presented in multiple colors, as shown in Fig. 17(e), (f), (g), (n), and (o), they are extracted correctly using the proposed method. Our approach handles well the overlay texts independent of contents types, such as movie, drama, animation, and so on (see the third row of Fig. 17). In the fourth row of Fig. 17,



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m) (n) (o) (p)

Fig. 17. Experimental results of the overlay text detection.

TABLE IV
DETECTION ACCURACY WITH VARIATION OF *TH* VALUE

	TH = 60	*TH* = 70	*TH* = 80	*TH* = 90	*TH* = 100
FP	20.72 %	12.81 %	5.83 %	4.45 %	1.58 %
FN	3.23 %	2.63 %	2.45 %	10.36 %	23.64 %

TABLE V
PERFORMANCE EVALUATION OF THE PROPOSED METHOD

	Slowest case	Fastest case	Average case
Total processing time without update algorithm	18.32 fps	27.86 fps	23.23 fps
Total processing time with update algorithm	26.05 fps	31.77 fps	29.70 fps
Increasing rate of speed	42.19 %	14.03 %	27.85 %

the overlay texts with different sizes are correctly detected in the same video scene.

The framework for evaluating performance has been implemented by using Visual Studio 2003 $(C++)$ under FFMpeg library, which has been utilized for MPEG and DIVX decoding. Various videos are encoded with the image size of 320×240.

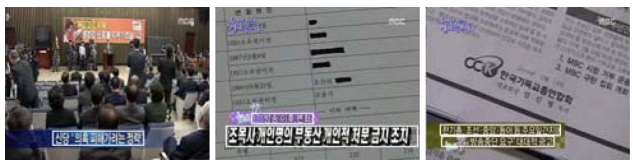


Fig. 18. Results of overlay text detection from scene text background.

Since the *TH* value in Section II-A plays an important role to generate a robust transition map, it is carefully set to 80 to minimize false positives (FP) and false negatives (FN) of overlay text detection as shown in Table IV. The minimum size of overlay text region used to remove small components in Section II-B is set to be 300. The threshold value for the overlay text region update is set to be 500. The parameters, such as window size for adaptive thresholding, the minimum size of overlay text region, and the threshold value for the overlay text region update, can be consistently set according to the image width or height. The experiments were performed on the low-end PC (Core2Duo 1.8 GHz). The comparison of total processing time is shown in Table V. To check the efficiency of text region update, we measure both the total processing time without update algorithm, which was addressed in Section II-E and with update algorithm separately using videos shown in Fig. 17.

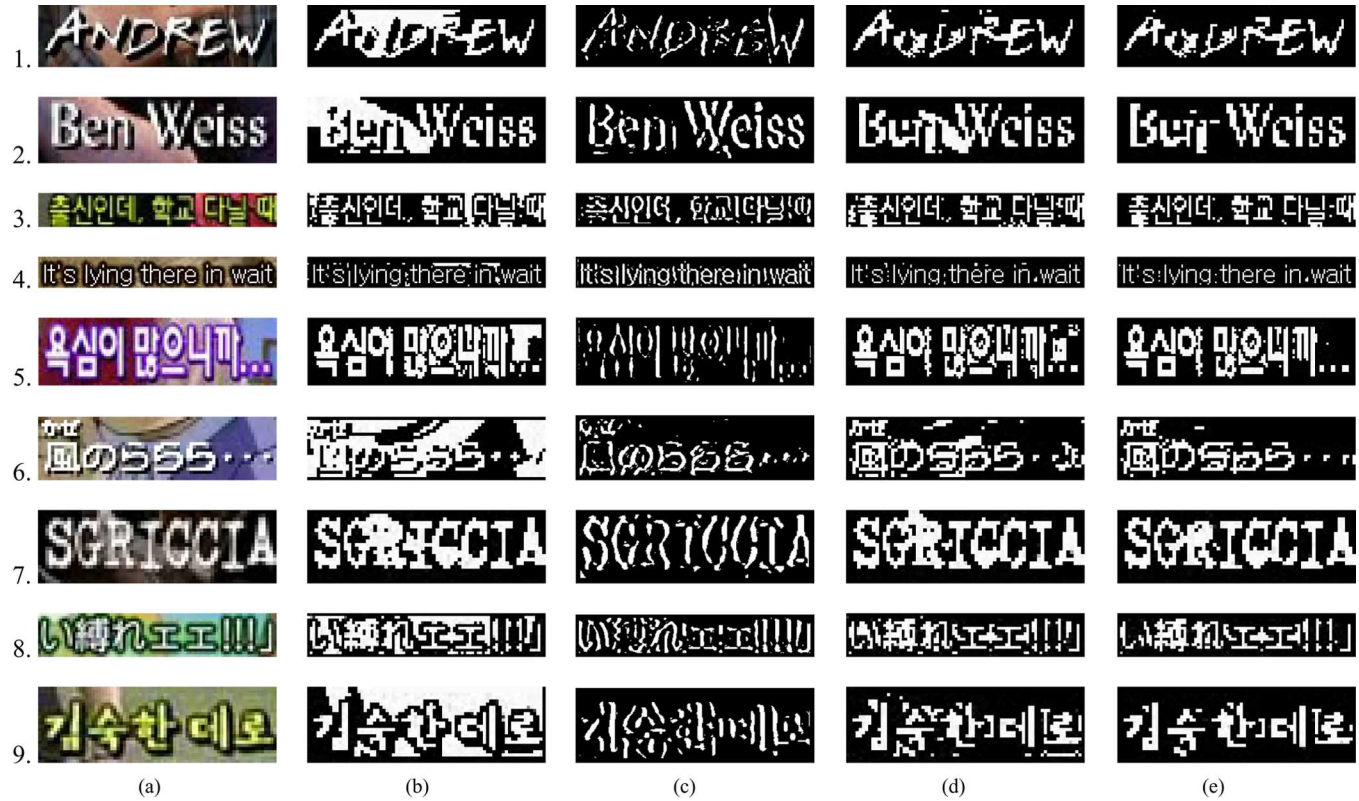


Fig. 19. Results of overlay text extraction (a) Original image. (b) Otsu result. (c) Sato filtering result. (d) Lyu result without the height normalization. (e) Our approach.

Fig. 20. Examples of dark overlay text extraction.

Since post processing after the transition map generation is omitted in the overlay text region update, the total processing time can be reduced. It makes our method much faster than other previous methods. Since the more candidate regions require the more total processing time, the total processing time is different with the number of detected candidate regions. Based on Table V, we can see that the proposed update algorithm is very efficient and makes possible real-time implementation because the total processing time with update algorithm is about 30 fps.

Moreover, since the proposed method focuses on the overlay text regions, it is essential to detect only the overlay text even though the scene text and overlay text are displayed in the same video scene. The results of detected overlay text from scene text background are shown in Fig. 18. We see that the proposed method is robust to the scene text background, such as banner, texts on the document and newspaper as shown in Fig. 18.

C. Performance Evaluation of Overlay Text Extraction

In order to confirm the superiority of our proposed text extraction method, we compare our approach with other methods; Otsu [14], Sato filtering method [11], and Lyu's method without the height normalization process [7]. All extraction methods are applied to the results of our overlay text region detection. The results of overlay text extraction are shown in Fig. 19. Dark overlay text is also extracted well by the color polarity computation. The result of dark overlay text extraction is shown in Fig. 20.

Since the Otsu method focuses on the global thresholding by using histogram-based analysis, it is not robust to background color variation. In Fig. 19(b), the background pixels, such as hand, clothes, and so on, are extracted as the overlay texts due to the similarity in intensity. The results obtained by using Sato filtering are shown in Fig. 19(c). Although the overlay texts are well extracted from videos with simple background, such as news, it fails for the complex background due to the various colors and font styles [see Fig. 19(c)]. Most overlay texts are well extracted by the Lyu's method even though the height normalization is omitted. However, as shown in Fig. 19(d) that some small noise pixels in *TR* cannot be removed by the inward filling, they are mistakenly extracted as the overlay texts. To solve this problem, we added a simple constraint based on

TABLE VI
PE FOR EACH EXTRACTION RESULT

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9
Otsu method	0.158	0.191	0.131	0.039	0.151	0.348	0.126	0.261	0.378
Sato method	0.110	0.105	0.160	0.091	0.153	0.118	0.172	0.209	0.161
Lyu method	0.086	0.111	0.085	0.011	0.108	0.114	0.108	0.165	0.107
Our method	**0.088**	**0.078**	**0.057**	**0.006**	**0.076**	**0.071**	**0.100**	**0.135**	**0.076**

TABLE VII
COMPARISON OF EFFICIENCY OF EXTRACTION

	Total processing time	Average PE
Otsu method	31.14 fps	0.198
Sato method	20.46 fps	0.142
Lyu method	29.33 fps	0.099
Our method	**29.35 fps**	**0.076**

detected transition pixels in the transition map. In Fig. 19(e), we can see that small noise pixels in *TR* are removed correctly.

The accuracy of overlay text extraction shown in Fig. 19 is evaluated using the probability of error (PE) [19] as follows:

$$PE = P(T)P(B \mid T) + P(B)P(T \mid B) \tag{12}$$

where $P(T)$ and $P(B)$ denote the probabilities of overlay text pixels and background pixels in the ground-truth image, respectively. $P(B \mid T)$ denotes the error probability to classify overlay text pixels as background pixels while $P(B \mid T)$ denotes the error probability to classify background pixels as overlay text pixels. Small PE means that the accuracy of overlay text extraction is high. The PE for each extraction result is shown in Table VI. Note that the increase of PE in our method compared to Lyu method is negligible in Image 1. We can see that the overlay texts are accurately extracted in our method.

We compare the efficiency of extraction using videos shown in Fig. 19. Based on Table VII, it is noted that our method is very efficient.

Although the experimental results in this section showed the efficiency and robustness of the proposed method, the behavior and limitations of the proposed method need to be briefly addressed.

First, we can consider the case that the proposed method is applied to high quality videos, where color bleeding may be reduced by the advanced compression techniques. In this case, the value D_L defined in (3) is also reduced while D_H increases. In other words, the constraint for logarithmical change in (4) is relaxed, and thus no problem is encountered in generating the transition map.

Secondly, we also consider the case that the proposed method is applied to the semitransparent overlay text. Content providers sometimes insert the semitransparent overlay text to reduce the loss of the original video scene. Since the logarithmical change of intensity may not be satisfied due to the background components on the back of the semitransparent overlay text, the transition map may not be correctly generated.

V. CONCLUSION

A novel method for overlay text detection and extraction from complex videos is proposed in this paper. Our detection method is based on the observation that there exist transient colors between inserted text and its adjacent background. The transition map is first generated based on logarithmical change of intensity and modified saturation. Linked maps are generated to make connected components for each candidate region and then each connected component is reshaped to have smooth boundaries. We compute the density of transition pixels and the consistency of texture around the transition pixels to distinguish the overlay text regions from other candidate regions. The local binary pattern is used for the intensity variation around the transition pixel in the proposed method. The boundaries of the detected overlay text regions are localized accurately using the projection of overlay text pixels in the transition map. Overlay text region update between frames is also exploited to reduce the processing time. Based on the results of overlay text detection, the overlay texts are extracted based on Lyu's extraction method. We add a simple constraint based on detected transition pixels in the transition map to improve the extraction performance. To validate the performance of our detection and extraction method, various videos have been tested. The proposed method is very useful for the real-time application. Our future work is to detect and extract the moving overlay text to extend the algorithm for more advanced and intelligent applications.

REFERENCES

[1] C. G. M. Snoek and M. Worring, "Time interval maximum entropy based event indexing in soccer video," in *Proc. Int. Conf. Multimedia and Expo*, Jul. 2003, vol. 3, pp. 481–484.

[2] J. Gllavata, R. Ewerth, and B. Freisleben, "Text detection in images based on unsupervised classification of high-frequency wavelet coefficients," in *Proc. Int. Conf. Pattern Recognition*, Aug. 2004, vol. 1, pp. 425–428.

[3] J. Cho, S. Jeong, and B. Choi, "News video retrieval using automatic indexing of korean closed-caption," *Lecture Notes in Computer Science*, vol. 2945, pp. 694–703, Aug. 2004.

[4] L. Agnihotri and N. Dimitrova, "Text detection for video analysis," in *Proc. IEEE Int. Workshop on Content-Based Access of Image and Video Libraries*, Jun. 1999, pp. 109–113.

[5] X. S. Hua, P. Yin, and H. J. Zhang, "Efficient video text recognition using multiple frame integration," in *Proc. Int. Conf. Image Processing*, Sep. 2004, vol. 2, pp. 22–25.

[6] K. C. K. Kim *et al.*, "Scene text extraction in natural scene images using hierarchical feature combining and verification," in *Proc. Int. Conf. Pattern Recognition*, Aug. 2004, vol. 2, pp. 679–682.

[7] M. R. Lyu, J. Song, and M. Cai, "A comprehensive method for multilingual video text detection, localization, and extraction," *IEEE Trans. Circuit and Systems for Video Technology*, vol. 15, no. 2, pp. 243–255, Feb. 2005.

[8] C. Liu, C. Wang, and R. Dai, "Text detection in images based on unsupervised classification of edge-based features," in *Proc. Int. Conf. Document Analysis and Recognition*, Sep. 2005, vol. 2, pp. 610–614.

[9] X. Liu and J. Samarabandu, "Multiscale edge-based text extraction from complex images," in *Proc. Int. Conf. Multimedia and Expo (ICME)*, Jul. 2006, pp. 1721–1724.
[10] M. Bertini, C. Colombo, and A. D. Bimbo, "Automatic caption localization in videos using salient points," in *Proc. Int. Conf. Multimedia and Expo*, Aug. 2001, pp. 68–71.
[11] T. Sato, T. Kanade, E. K. Hughes, and M. A. Smith, "Video OCR for digital news archive," in *Proc. IEEE International Workshop on Content-Based Access of Image and Video Libraries*, Jan. 1998, pp. 52–60.
[12] J. Wu, S. Qu, Q. Zhuo, and W. Wang, "Automatic text detection in complex color image," in *Proc. Int. Conf. Machine Learning and Cybernetics*, Nov. 2002, vol. 3, pp. 1167–1171.
[13] Y. Liu, H. Lu, X. Xue, and Y. P. Tan, "Effective video text detection using line features," in *Proc. Int. Conf. Control, Automation, Robotics and Vision*, Dec. 2004, vol. 2, pp. 1528–1532.
[14] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. 9, no. 1, pp. 62–66, Mar. 1979.
[15] R. C. Gonzalez and R. E. Woods*, Digital Image Processing*, 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2002.
[16] T. Ojala, M. Pierikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.
[17] S. Antani, D. Crandall, and R. Kasturi, "Robust extraction of text in video," in *Proc. Int. Conf. Pattern Recognition*, Sep. 2000, vol. 1, pp. 831–834.
[18] J. M. Pike and C. G. Harris, "A combined corner and edge detector," in *Proc. Alvey Vision Conf.*, 1988, pp. 147–151.
[19] S. U. Lee, S. Y. Chung, and R. H. Park, "A comparative performance study of several global thresholding techniques for segmentation," *Comput. Vis., Graph., Image Process.*, vol. 52, pp. 171–190, 1990.

Wonjun Kim received the B.S. degree in electronic engineering from the Sogang University, Seoul, Korea, and the M.S. degree in electronic engineering from Information and Communications University (ICU), Daejeon, Korea, in 2006 and 2008, respectively. He is currently pursuing the Ph.D. degree in electronic engineering at ICU.

His current research interests are object segmentation, image processing, pattern recognition, and visual quality measure.

Changick Kim (M'01) was born in Seoul, Korea. He received the B.S. degree in electrical engineering from Yonsei University, Seoul, the M.S. degree in electronics and electrical engineering from the Pohang University of Science and Technology (POSTECH), Pohang, Korea, and the Ph.D. degree in electrical engineering from the University of Washington, Seattle, in 1989, 1991, and 2000, respectively.

From 2000 to 2005, he was a Senior Member of Technical Staff at Epson Research and Development, Inc., Palo Alto, CA. Since February 2005, he has been with the School of Engineering, Information and Communications University (ICU), Daejeon, Korea, where he is currently an Associate Professor. His research interests include multimedia communication, 3-D video processing, image/video understanding, intelligent media processing, and video coding for IPTV.